H2020 **GV-8-2015**
Electric vehicles' enhanced performance and integration into the transport system and the grid

electrific

Enabling seamless electromobility through smart vehicle-grid integration

Project Nº 713864

Electrific

D3.1 - Initial architecture and integration framework

Responsible:	FM
Contributors:	CVUT, GFI, THD, UNIMA, UNIPASSAU
Document Reference:	D3.1- Initial architecture and integration framework
Dissemination Level:	Public
Version:	1.0
Date:	28/02/2017

Executive Summary

This deliverable is the output of the analysis done in the context of WP3 and describes the initial high-level architecture of the ELECTRIFIC Solution that enables a collaboration between the electro-mobility actors to improve the EV infrastructure utilization, to optimize the grid and the use of renewable energy resources.

The ELECTRIFIC Solution is designed as a multi-agent system with three type of agents:

- The ADAS (Advanced EV Driver Assistance Services) agent that interacts with the ELECTRIFIC solution, on behalf of an Electric Vehicle (EV) user, to propose optimized routes for his/her trip. The optimization criteria that are considered in this iteration are the cost (minimization of the charging cost), the green aspect of the trip (maximization of the renewable intake) and the time.
- The Charging Service Provider (CSP) agent is responsible to provide to the ELECTRIFIC Solution the predictions in terms of available power, charging capacity and prices at the different charging stations (CS) that it manages. In addition, CSP agent monitors the power quality in the grid and adapts, if necessary, the current charging processes.
- The EV Fleet Operator (EFO) agent is responsible to interact with its internal systems to provide to the ELECTRIFIC solution the EVs that are available for booking and the status of these EVs (e.g. State of Charge, geolocation).

To enable the interactions between these agents, a middleware (Common Information Model) is foreseen to share the information about the agents and the assets that are managed by these agents (EVs or CSs).

The ELECTRIFIC solution is based on a micro-services architecture where each agent is developed as a suite of independent services communicating with others by a JSON-over-HTTP API.

Contributors Table

DOCUMENT SECTION	AUTHOR(S)	REVIEWER(S)
I. Introduction	Jean-Luc Sonnet (FM)	Benedikt Kirpes (UNIMA)
II Actors and interactions	Benedikt Kirpes (UNIMA), Jean-Luc Sonnet (FM)	Markus Eider (THD)
III High level architecture	Jean-Luc Sonnet (FM)	Markus Eider (THD), Hermann De Meer (UNIPASSAU), Matej Matejicek (GFI)
IV.1 Components Overview	Jean-Luc Sonnet (FM)	Gunther Verhemeldonck (GFI)
IV.2 User interfaces and associated services	Thomas Schulze (UNIMA), Filipe Calo (GFI), Seppe Dijkmans (GFI), Chen Chen (GFI)	Antonin Komenda (CVUT)
IV.3 ADAS Agent	Antonin Komenda (CVUT), Michal Štolba (CVUT), Markus Eider (THD)	Robert Basmadjian (UNIPASSAU)
IV.4 CSP Agent	Ammar Alyousef (UNIPASSAU), Dominik Danner (UNIPASSAU)	Thomas Schulze (UNIMA)
IV.5 EFO Agent	Markus Eider (THD), Florian Schweiger (THD), Andreas Berl (THD)	Antonin Komenda (CVUT)
IV.6 Common Information Model	Jean-Luc Sonnet (FM)	Ammar Alyousef (UNIPASSAU)
V Development and testing	Gunther Verhemeldonck (GFI), Filipe Calo (GFI), Seppe Dijkmans (GFI), Chen Chen (GFI)	Robert Basmadjian (UNIPASSAU), Adelfio D'Angio (FM)

Table of Contents

Table of Figures

List of Tables

Table of Acronyms

Acronyms	Meaning
AC	Alternating Current
ADAS	Advanced EV Driver Assistance Services
API	Application Programming Interface
ASL	Android Support Library

CCP	Charging Capacity Predictor
CD	Continuous Delivery
CI	Continuous Integration
CIM	Common Information Model
CS	Charging Station
CSMS	Charging Station Management System
CVRMSE	Coefficient of Variation of the Root Mean Square Error
CSP	Charging Service Provider
CStO	Charging Station Owner
DC	Direct Current
DSO	Distribution System Operator
DoA	Description of Action
EFMS	EV Fleet Management System
EFO	EV Fleet Operator
EFU	EV Fleet User
ELSA	ELectric vehicle Smart Algorithm
ESoH	Electrific State Of Health
EV	Electric Vehicle
EVD	Electric Vehicle Driver
EVO	Electric Vehicle Owner
EVU	Electric Vehicle User
GIS	Geographic Information System
GUI	Graphical User Interface
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
MBE	Mean Bias Error
PoI	Point of Interest
QoS	Quality of Service
R^2	Coefficient of Determination
REST	Representational State Transfer
RMS	Root Mean Square
SDK	Software Development Kit

SoH	Battery State of Health
SoC	Battery State of Charge
ToU	Time of Using
TSO	Transmission System Operator
PQ	Power Quality
WP	Work Package

Table of Definitions

Definitions	Meaning
Charging profile	It is a time depending profile containing information about the maximum charging capacity which can be used by the CS during a certain time slot.
Charging capacity	The Charging capacity is the maximum available power available in a certain timeslot
Charging options	A time depending list of the possible power capacities at a charging station with pricing. These capacities must be compatible with the maximum capacity of this charging station and the technical specifications of its connectors.
Charging model	The charging model is a computational service in form of a remotely or locally called function which for a particular EV (described by EV type, SoH and SoC) computes how SoH and SoC will change in time, provided that the EV is connected to a charging station and is charged using a charging rate.
Discharging model	The discharging model is a computational service in form of a remotely or locally called function which for a particular EV (described by EV type, SoH and SoC) computes how SoH and SoC will change in time, provided that the EV goes by a particular route (including the vertical profile) with a particular speed profile.
Forecasting model	A model to Predict the Future using (time series related or other) data we have at hand.
Service proposal	A service proposal from CSP or EFO components to ADAS consists of a description of the charging or EV renting service respectively. The description contains time of charging or rental, charging rate (for CSP), EV type (for EFO), price and fees of cancelation from both the CSP/EFO and the user.

I. INTRODUCTION

I.1. Purpose and organization of the document

This document is the first deliverable (D3.1) of WP3 and describes the ELECTRIFIC technical solution. It presents the overall architecture, functional description of the technical components and the scope for the first iteration of the ELECTRIFIC Solution (Preliminary prototype in project milestones of the DoA). The deliverable document is structured as follows:

The first chapter introduces purpose, organization, scope, audience, and context of this document. The second chapter describes the different actors and their interactions with the ELECTRIFIC Solution. In the third chapter, the high-level architecture of the complete ELECTRIFIC Solution, which will address the needs of the different actors, is presented, and explained. The fourth section provides technical details of the different ELECTRIFIC Solution components and how they are going to support the specified functionalities. The fifth chapter deals with development and testing tools which will be used within the ELECTRIFIC project.

I.2. Scope and audience

This deliverable describes the high-level architecture of the ELECTRIFIC solution. It will serve as technical basis and provide guidelines to the technical WPs involved in the development of the ELECTRIFIC software components. The design of the different components of the ELECTRIFIC Solution will be covered in more detail by the following deliverables: D3.2, D4.1, D5.1, D6.1 and D7.1.

The concepts of this solution are based on the business requirements and use cases defined in WP2. The first deliverable concerning the business requirements and use cases (D2.2) is planned chronologically after this deliverable at hand. Therefore, this initial architecture considers a sub-set of the actors and use cases, as specified by the final version of D2.2. The second chapter of this document introduces the actors and related interactions that will be considered for this deliverable.

The architecture will be validated with simulated and real-life data (WP8). It will then be refined to incorporate additional requirements as defined by WP2 and considering the results from the experimental phase.

The final architecture of the ELECTRIFIC Solution will be described in D3.4.

In this iteration, three different actors are considered: The EV user (EVU) that uses ELECTRIFIC solution to plan his/her trips; the EV Fleet Operator (EFO) that manages a fleet of EVs and the Charging Service Provider (CSP) that manages a set of charging stations.

The focus of this first iteration is set on the main use case for the EVU which plans a trip and receives optimized routing suggestions for his/her trip from the ELECTRIFIC Solution. This requires the collection and computation of data coming from the grid, the charging stations, the EV fleet and the EV.

Use cases related to the EFO for the charging scheduling of unused EVs from a fleet and the re-allocation of EVs based on the battery information are not covered in this iteration (except booking process partially). The incentives schemes and models to be supported by the software solution will also be integrated in the next iteration of this deliverable (based on D2.2 requirements).

This deliverable is primarily intended for EC Officers and Project-appointed reviewers. Further, the document is relevant for project members involved in the development within the different technical Work Packages (WP3, WP4, WP5, WP6 and WP7) as it describes the basic architectural choices and guidelines. The document is also intended for the trial partners within the project (WP8) as it depicts the architecture of the software solutions, that will be used and tested during the trials and experiments.

II. ACTORS AND INTERACTIONS

The ELECTRIFIC Users (human actors) interacting with the ELECTRIFIC Solution are defined and described in detail in D2.2 (subject to change for final deliverable). This chapter provides an overview of the actors as they are defined at the time of the writing of this document. The following diagram shows a re-cap of the ELECTRIFIC User terminology:

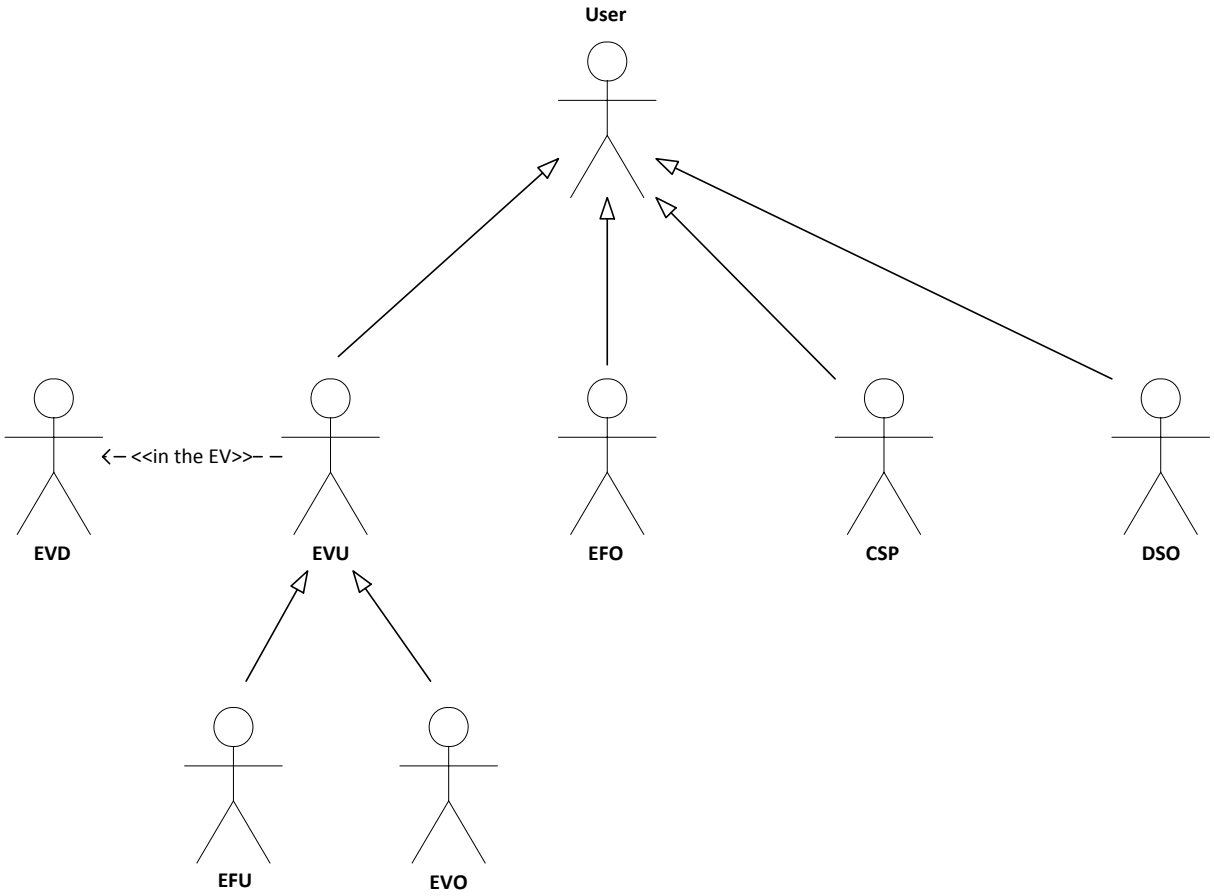

Figure 1. User Terminology.

Besides the Users, several other actors (systems) interact with ELECTRIFIC. These systems are not described with technical details in D2.2, so they will be defined in this deliverable.

Table 1. Actors description.

Actor	Description
Electric Vehicle (EV)	System, which can be an electric car, bus, scooter, etc. including its battery management system.
EV Owner (EVO)	Human actor, owning a private EV.
EV Fleet User (EFU)	Human actor, using an EV from a fleet.
EV User (EVU)	Human actor, using an EV. Can be either EFU or EVO.
EV Driver (EVD)	Human actor, which is an EVU inside/driving an EV.

EV Fleet Operator (EFO)	Human actor, managing an internal or external fleet of EVs. *External*: long-term rental, short-term rental *Internal*: EVU and EV belong to the same organization, e.g. public transport (bus, taxi), delivery services
EV Fleet Management System (EFMS)	System, used by the EFO to manage an EV Fleet.
Charging Station (CS)	System, where EVs can be charged and its control logic.
Charging Station Owner (CStO)	Human actor, who owns CS (public, private, commercial).
Charging Service Provider (CSP)	Human actor, who has a contract with the CStO to operate his CS.
Charging Station Management System (CSMS)	System, used by the CSP to manage the CSs.

Although some of the actors are included on business use case level in D2.2, here the following actors are not considered for technical interactions in the first iteration: DSO, TSO, ADAS provider, EV manufacturer, battery manufacturer, energy supplier and provider and utility services (traffic, weather, …).

II.1. User interactions

Based on the business use cases and requirements (D2.2) at their current state of this deliverable, in this section, the different technical user interactions with ELECTRIFIC are specified.

II.1.1. EV User

The table and diagram below show the interactions between EVU and the ELECTRIFIC Solution (the column “preliminary prototype” indicates if it will be part of the first preliminary prototype).

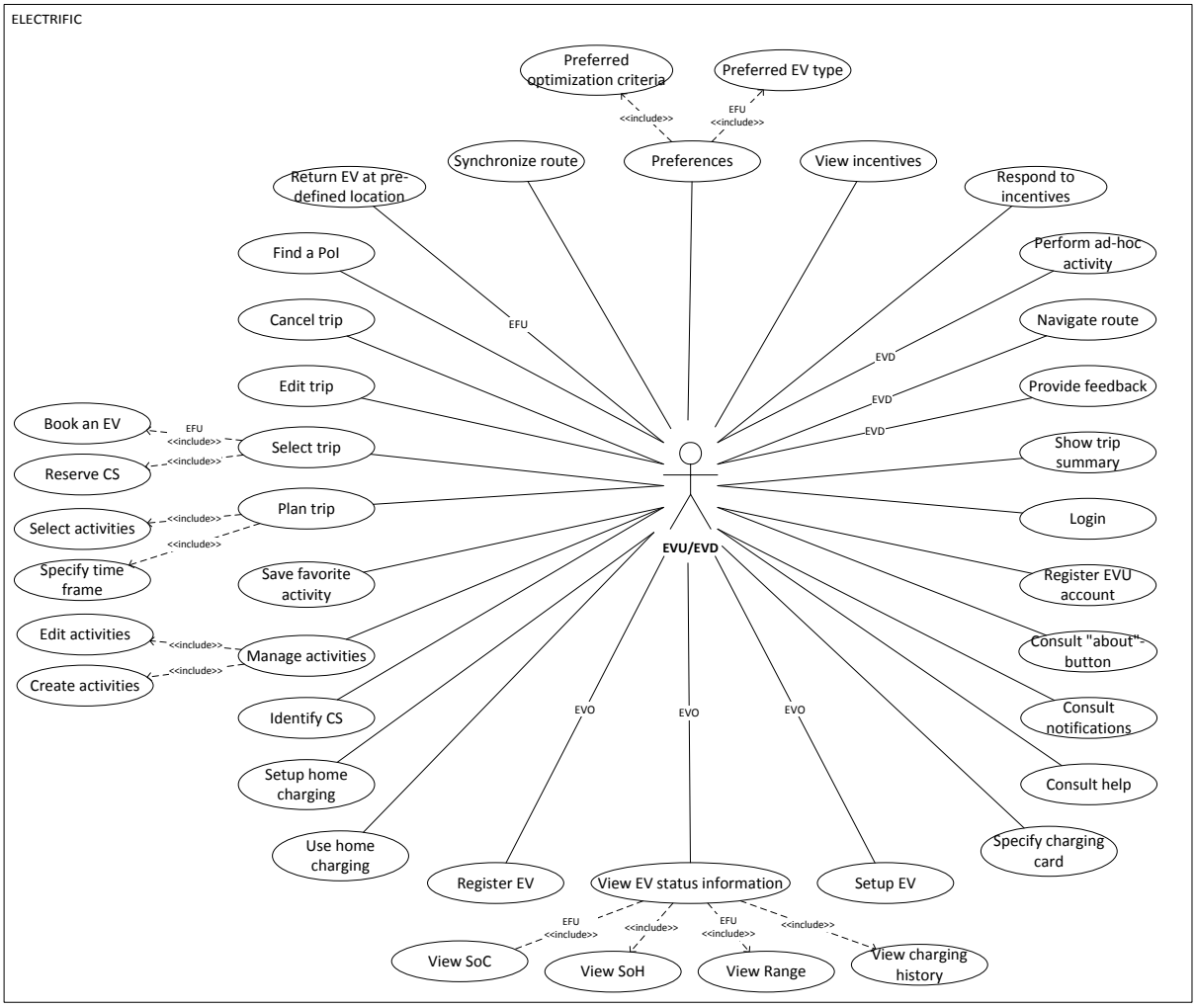

Figure 2. EVU Interactions.

Table 2. EVU interactions.

Actor	Interaction	Description	Preliminary prototype (Yes/No/Partial)
EVU	Register EVU account	Registration of an EVU user account.	**Partial**. Not via UI nor via API. Users are preregistered.
EVO	Register EV	Registration of the EV (type) and ownership (fleet, private, enterprise, leasing company).	**Partial**. Not via UI. EVs are preregistered.
EVO	Setup EV	Setup EV for data collection (battery, EV measurements) e.g. connect via Bluetooth, USB, CAN bus.	**Partial**. ADAS UI component will be manually configured to collect data of EV.

EVU	Login	Login to ELECTRIFIC using the credentials obtained from (pre-) registration.	Yes.
EVU	Preferences	Set general user preferences such as: ● Language. ● Colour codes. ● Minimum SoC (charge EV in a way that the battery level is never below a certain %). Preferred optimization criteria: cost, greenness, time **Preferred EV type:** The user specifies which type of EV he/she wants to use by selecting one or more EV manufacturer and model. [EFU]	No.
EVU	Manage activities	Create activities manually. An *activity* is defined by: ● Location. ● Time window (optional). ● Duration (optional). **Edit activities** that are already in the system.	Yes
EVU	Plan trip	**Select activities** (location + time window + duration) for a trip today or tomorrow by selecting from: ● History. ● list of favourite activities. ● list of abstracted terms (e.g. laundry, hairdresser). ● by importing a calendar; or ● from nearby activities. **Specify period** for the trip including start and end time.	**Partial.** Import from calendar not. Abstracted terms not. Favourite activities and nearby not.
EVU	Save favourite activity	A set of activities can be saved as an activity under a certain name to allow the EV user to select it later in a faster way.	No.
EVU	Select trip	Select one trip from a selection of trips, taking planned activities, future grid status and one optimization criterion (cost, time, or greenness) into account. Different optimization criteria are:	Yes.

		• cost (charge EV to minimise the charging cost). • time (charge EV to perform activities in shortest time). • greenness (charge EV with max renewable). Select trip is extended by **Book an EV** (only EFU) and **Reserve CS**.	
EFU	Book an EV	Together with the trips, a proposed EV from the EFO is shown to the EFU. This EV can be booked.	**Yes.** But no integration with booking system.
EVU	Reserve CS	The CS proposed in the selected trip can be reserved.	Yes.
EVU	Synchronize route	Synchronize the route of the selected trip with external navigation.	Yes.
EVD	Navigate route	Use the app's internal navigation functionality.	No.
EVD	Perform ad-hoc activity	Perform an ad-hoc activity (preferably selected from a list) when in the EV.	Yes.
EVU	Cancel trip	Cancel a trip (and all possible reservations done).	Yes.
EVU	Edit trip	Modify a selected trip before starting the trip.	No.
EVU	Identify CS	Identify nearest or most suitable CS (considering the current battery SoC, optimal recharge location).	No.
EVU	Consult notifications	Receive and consult notifications from ELECTRIFIC about charging prices (discounts), green electricity, promotions.	No.
EVU	Consult "about"-button	About button: explanation about which data (EV, user, CS, grid) is used and how it is used to provide the ELECTRIFIC assistance service. View information about green points.	Yes.
EVU	Consult help	Consult the help functionality within the ELECTRIFIC Solution.	No.

EVU	Show trip summary	Show trip summary based on data automatically collected by ELECTRIFIC from the EV (speed, battery level, …).	No.
EVD	Provide Feedback	Ask driver to provide feedback about ELECTRIFIC service, driving and charging experience.	Yes.
EVU	View incentives	User can view different types of incentives at different places in the app, e.g. green points, financial, coupons.	**TBD.** Analysis based on WP6 requirements
EVU	Respond to incentives	Respond (accept, decline) to proposed incentives by the ELECTRIFIC Solution.	No.
EVU	View EV status information	View status information about the EV: ● View SoC (EVO + EFU). ● View Range (EVO + EFU). ● View SoH (EVO). ● View charging history (EVO).	No.
EFU	Return EV at pre-defined location	Different pre-defined locations (hotspots), where the EV can be returned, are shown to the EFU. This interaction may include incentives.	No.
EVU	Find a PoI	Find any Point-of-Interest (tourist attraction, super market, etc.). Can also be used to display coupons, discounts at local places.	No.
EVU	Setup home charging	Include home charging by setting up a private CS at home location.	No.
EVO	Use home charging	Charge an EV with private CS at home location.	**No.** EV is assumed to be fully charged when used by the EV owner.
EVU	Specify charging card	Different charging cards are used for different charging services, offered by CSPs.	No.

II.1.2. EV Fleet Operator

The table and diagram below show the different interactions between EFO and the ELECTRIFIC Solution.

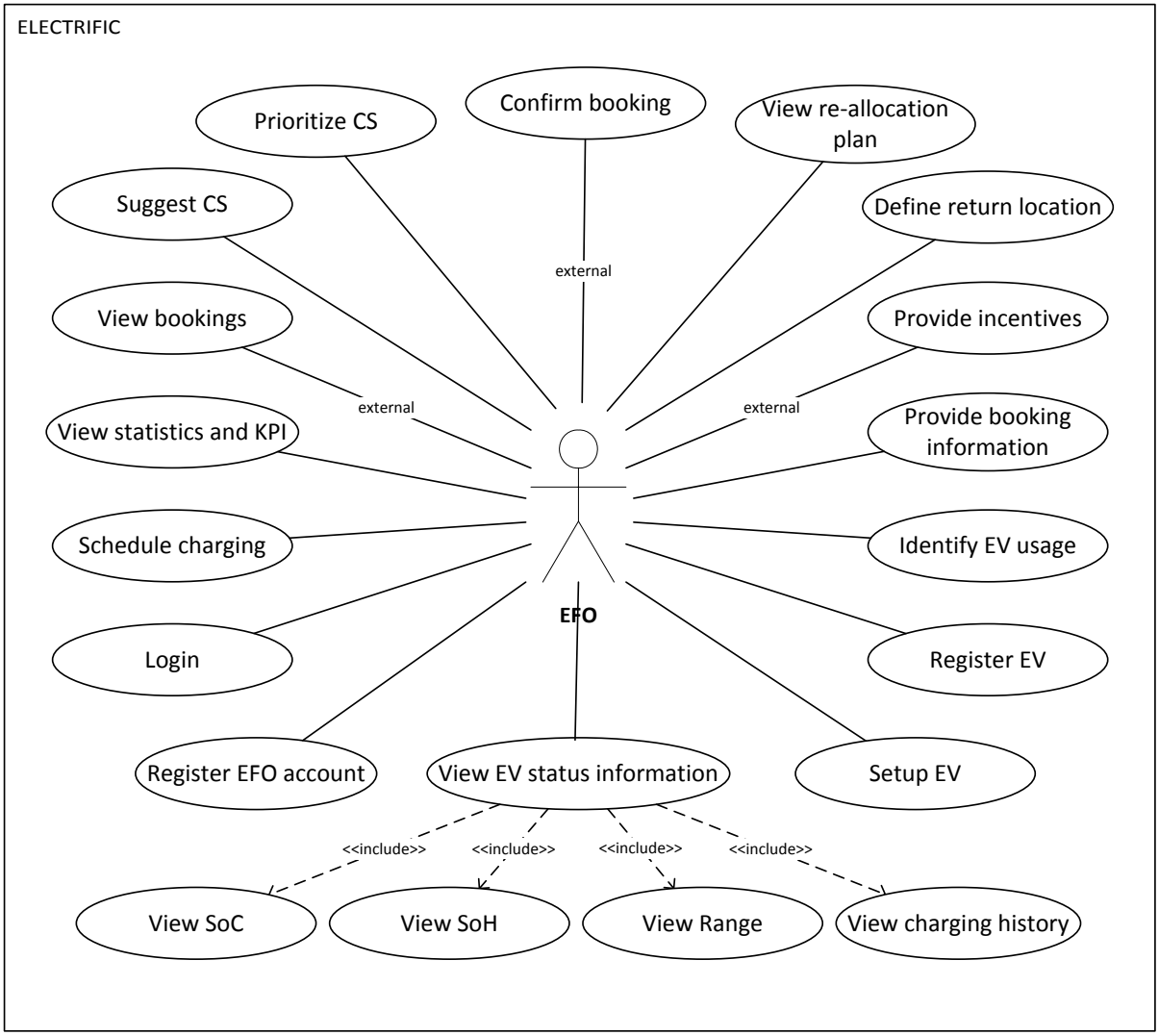

Figure 3. EFO Interactions.

Table 3. EFO interactions.

Actor	Interaction	Description	Preliminary prototype (Yes/No/Partial)
EFO (both[1])	Register EFO account	Registration of an EFO user account.	No.

[1] Both means internal and external EFO

EFO (both)	Login	Login to ELECTRIFIC using the credentials obtained from (pre-) registration.	No.
EFO (both)	Setup EV	Setup EV for data collection (battery, EV measurements) e.g. connect via Bluetooth, USB, CAN bus.	No.
EFO (both)	Register EV	Registration of the EV (type) and ownership (fleet, private, enterprise, leasing company).	**Partial**. Not via UI nor via agent API. EVs are preregistered.
EFO (both)	View EV status information	View status information about the EV: ● View SoC. ● View Range. ● View SoH. ● View charging history.	**Partial.** EV data is collected.
EFO (ext.)	Confirm booking	Confirm booking either manually or by the EFMS.	**No.** EVs are booked when selecting a trip.
EFO (both)	Schedule charging	Schedule charging of: 1. unbooked [external] / unused [internal] EVs of the fleet at company location. 2. including CS near company location (~5-10 km). 3. considering Smart Charger. in an optimal way, according to: ● Fleet EV booking schedule. ● SoH. ● Grid aspects. ● Energy price.	**No.** EVs are assumed to be fully charged when made available to the EV user.
EFO (both)	Identify EV usage	Identify EV usage patterns and associated cost.	**Partial.** EV data is collected.
EFO (both)	View statistics and KPI	View some statistics and KPI, e.g. increase in profitability/reliability due to ELECTRIFIC service.	No.
EFO (both)	Provide booking information	Provide booking information (schedule, location, etc.) to ELECTRIFIC for all participating EVs from the fleet.	Yes.
EFO (ext.)	View bookings	View bookings of EVs, which were done via the ELECTRIFIC service/app.	No.

EFO (ext.)	Provide incentives	Provide different types of incentives to the EFU.	No.
EFO (both)	Suggest CS	Suggest certain CS to the EFU via user app.	No.
EFO (both)	Define return location	Define one or multiple locations where the EV should be returned after the rental or internal use.	No.
EFO (both)	Prioritize CS	Suggest prioritized CSs to the ELECTRIFIC Solution.	No.
EFO (both)	View re-allocation plan	View the proposed re-allocation plan for the EVs from ELECTRIFIC.	No.

II.1.3. Charging Service Provider

The table and diagram below show the different interactions between CSP and the ELECTRIFIC Solution.

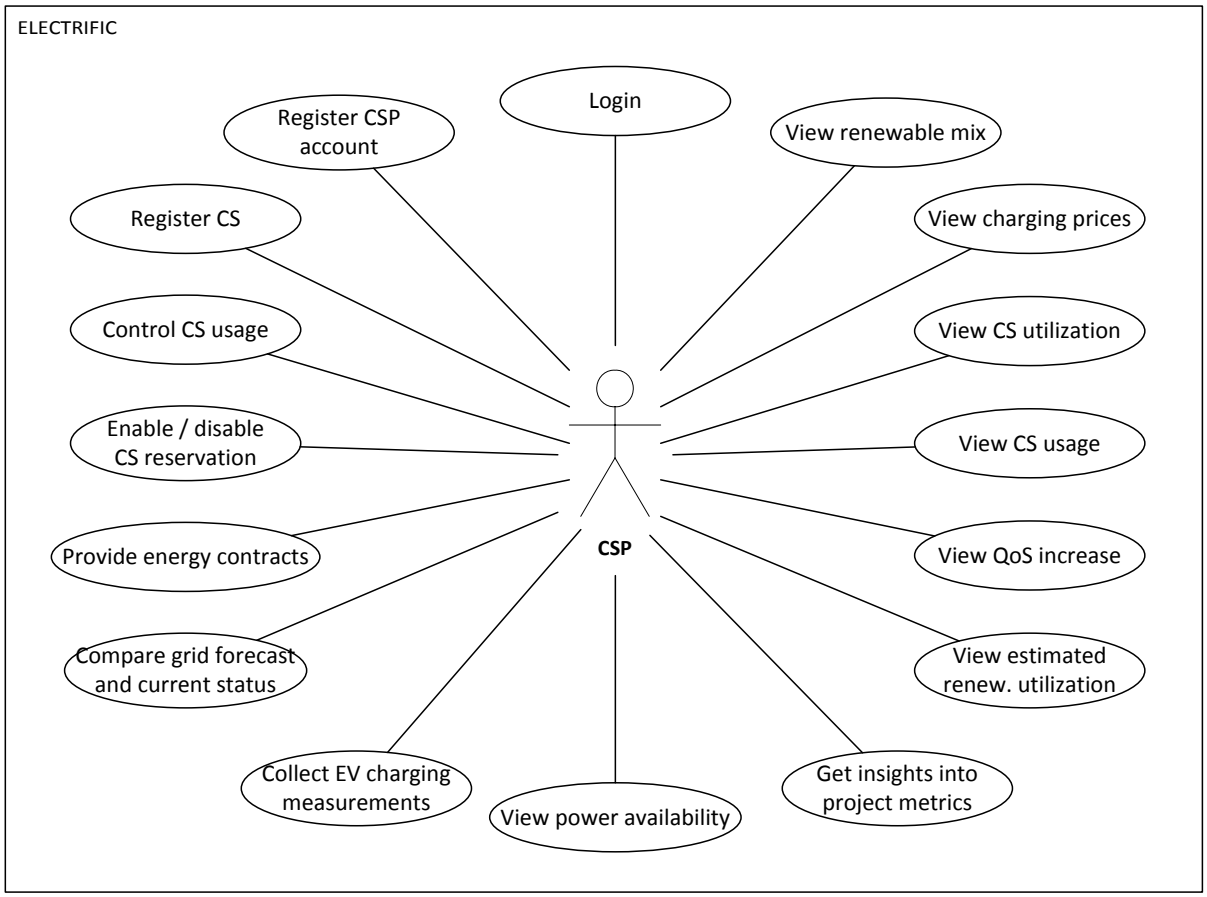

Figure 4. CSP interactions.

Table 4. CSP interactions.

Actor	Interaction	Description	Preliminary prototype (Yes/No/Partial)
CSP	Register CSP account	Registration of an CSP user account.	No.
CSP	Login	Login to ELECTRIFIC using the credentials obtained from (pre-)registration.	No.
CSP	Register CS	Registration of CS to be used in ELECTRIFIC solution.	**Partial**. Not via UI nor via agent API. CSs are pre-registered.
CSP	View power availability	View short-term power availability (contracts) from DSO.	**Partial.** CS data is collected and stored in the CIM.
CSP	View renewable mix	View renewable mix at the different CSs (from DSO).	**Partial.** CS data is collected and stored in the CIM.
CSP	View CS usage	View CS usage (booked and not booked) forecast from Electrific EVs (aggregated view).	**Partial.** CS data is collected and stored in the CIM.
CSP	Control CS usage	Control the usage of all CSs, e.g. if certain CSs should be used less or more, depending on different situations, e.g. technical issues.	No.
CSP	View CS utilization	View utilization (current, historic, forecast) of the CSs.	No.
CSP	Enable / disable CS reservation	Enable or disable certain CSs for reservation by ELECTRIFIC.	No.
CSP	View QoS increase	View increase in QoS due to using the ELECTRIFIC solution.	No.
CSP	View charging prices	View the prices of charging cycles (current and historic prices).	**Partial.** CS data is collected and stored in the CIM.

CSP	View estimated renew. utilization	View estimated renewable energy utilization for each CS (+ aggregated).	No.
CSP	Compare grid forecast and current status	Compare the forecasted grid status with the actual current grid status.	**No.** Not collecting data of grid but using simulated data.
CSP	Collect EV charging measurements	Collect EV charging measurements (charging type, duration, used power).	**Partial.** CS data is collected and stored in the CIM.
CSP	Get insights into project metrics	Insights into project-related metrics (# of users, EVs, attractiveness, power grid, incentives).	No.
CSP	Provide energy contracts	Provide (parameters of) energy contracts (max power) between CSP and Energy provider/DSO.	No.

III. HIGH LEVEL ARCHITECTURE

III.1. High Level view

One of the objectives of ELECTRIFIC project is to enable seamless electromobility through a smart EV-grid integration and coordination among the actors of this ecosystem. An expected deliverable, among others in the project, is the ELECTRIFIC Solution which will incorporate the project goals.

The Agent modelling paradigm is used to capture the interactions and dynamics of the different actors of the ELECTRIFIC Solution. This means, that the ELECTRIFIC Solution is modelled as a multi-agent system. Such a multi-agent system consists of several agents, which are entities capable of autonomous decisions, in a shared environment. An agent is typically defined by its internal state, actions it can perform in the environment, possible interactions with other agents and its goals and intentions.

Three types of agents are modelled:

- Advanced Driver Assistance Services (ADAS) agent, is the agent responsible for communicating with the ELECTRIFIC Solution on behalf of the EV user.
- Charging Service Provider (CSP) agent, is the agent interacting with the ELECTRIFIC Solution on behalf of the CSP.
- EV Fleet Operator (EFO) agent, is the agent interacting with the ELECTRIFIC Solution on behalf of the EFO.

The Common Information Model (CIM) is the middleware which supports the interactions between these agents (agent routing, interfaces, central data storage and analysis).

Figure 5 depicts the main modules of ELECTRIFIC Solution and external systems that interact (EFO, CSP) with it. External systems are represented by blue (far left) and green (far right) rectangles outside main rectangle in the middle (representing main ELECTRIFIC Solution).

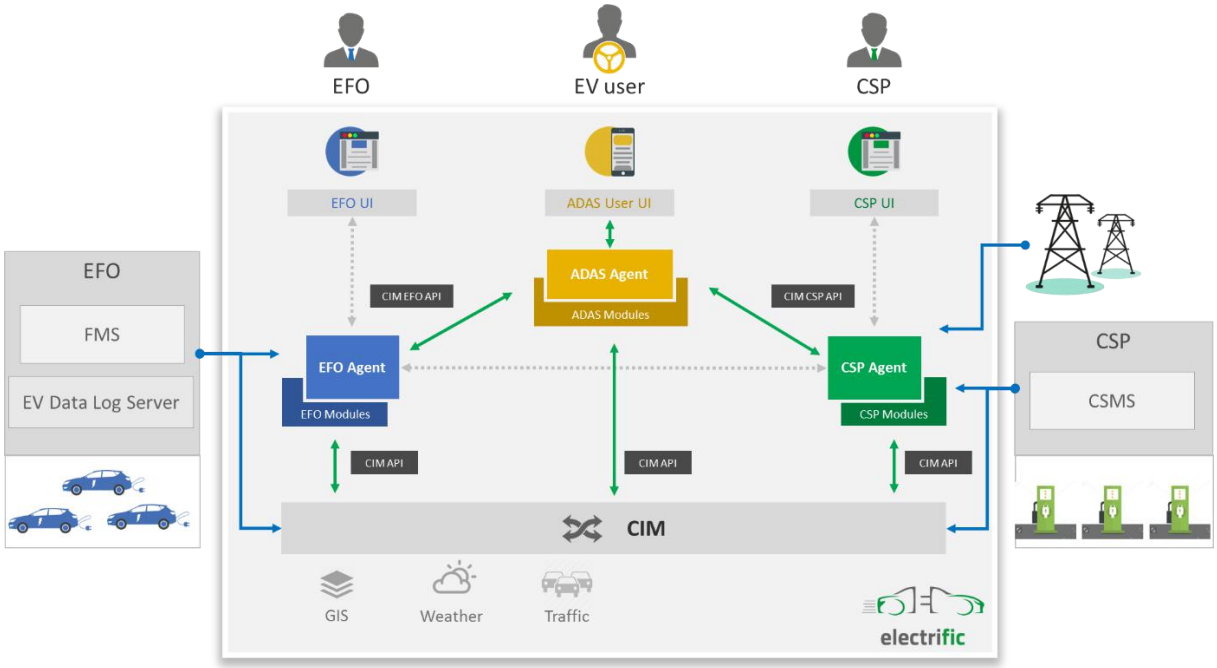

Figure 5. ELECTRIFIC Solution High level view.

On the EFO side, the ELECTRIFIC Solution interacts with the Fleet Management system (FMS) used by an EFO to manage an EV Fleet. Similarly, ELECTRIFIC communicates with the EV Data Log server used by the EFO to collect EV data (SoC, …) of the different EVs.

On the CSP side, the ELECTRIFIC Solution interacts with the Charging Management System (CSMS) used by the CSP to manage its charging stations; It interacts also with the Grid Management System to retrieve grid information.

The EV user interacts with ELECTRIFIC Solution through a mobile application (**ADAS User UI**) used by the user at all times, not only while being in an EV but also from home, at the office or outside. As it will be interacting with human users, it will be designed to give the optimal user experience. Permissions to access certain operations depend on the user's role. Using ELECTRIFIC, the EV user can plan his/her activities and select trips and EV that fulfil his/her plans. The user has also the possibility to reserve the charging slots that have been identified for his/her trip. The EV user can also define his/her preferences that are considered in the trip selections.

The **ADAS agent** identifies the optimal route and EV selection for the user, taking the grid and EV into account.

It communicates with the different EFO agents to retrieve the price, SoC, and additional properties of their EVs. It communicates with the different CSP agents to retrieve information about availability, price, power quality and percentage of renewables at the different charging stations.

Based on the activities and preferences (optimisation criteria) of the EV user, the ADAS agent will then propose to the EVU the best trip and EV to fulfil his/her activities. It will do the reservation of EV and charging slots if requested by the EV user.

CIM supports the ADAS agent to identify the CSP and EFO agents that should be contacted.

The **CSP agent** is responsible to provide all the grid information required by the ADAS agent to perform the optimal trip selection for the EV user. This information is the prediction of available power, renewable percentage, charging profiles and prices at the different charging stations managed by the CSP.

To build this information, the CSP agent retrieves the power and renewable forecasting from the Grid Management System.

The CSP agent is also responsible to interact with the CSMS to make the reservation of a charging slot once initiated by the EV user in his/her trip selection.

Finally, the CSP agent is responsible to collect real-time data from the grid to estimate the current power quality in the grid: In case of degraded grid status, the CSP agent will send new charging profiles to the charging stations for the current charging processes.

The **EFO agent** is responsible to provide all information concerning the vehicles required by the ADAS agent to perform its optimisation. This information is mainly the SoC, geolocation and in case of an EV from a fleet, the price and availability.

Each EFO agent interacts with their relevant internal systems (Fleet management system (FMS), EV Data Log server) to collect this data and make it available to the ADAS agent.

The **CIM** is used to share between the different agents the necessary information for these agents to interact with each other. It also defines the generic interfaces between these agents:

- CIM CSP API is the generic API implemented by the CSP Agent to communicate with ADAS agent.
- CIM EFO API is the generic API implemented by the EFO Agent to communicate with ADAS agent.

In the scope of the project, CIM is also used by the EFO and CSP agents to store relevant data (CIM API) to allow computation of the different metrics defined in the ELECTRIFIC project and shown in the CSP/EFO dashboards.

The dashed line rectangles and interconnecting lines inside the ELECTRIFIC Solution represent the EFO and CSP user interfaces, the API between the CSP and EFO agents (this will be required for the development of the Fleet assistance services described in section IV.5.5. Some external services useful for ELECTRIFIC Solution (traffic, weather…) are not part of the first iteration.

III.2. Registration

For this first iteration of the deliverable, the registration aspects have not been designed. All entities (EVUs, CSs, Vehicles, Agents) will be predefined in the system.

In the next iteration, the corresponding agent will do the registration of the different entities. Only the identification of the relevant entities of the EFO and CSP agents will be centralized in the CIM.

The registration of users will be done in a dedicated database.

IV. ELECTRIFIC SOLUTION COMPONENTS

IV.1. Components Overview

Figure 6 shows all the components of the ELECTRIFIC solution (in yellow) and the module they belong to: ADAS User UI (in orange), ADAS Agent (in brown), CSP Agent (in green), EFO Agent (in blue), CIM (in grey). These components are not necessarily collocated on a same server.

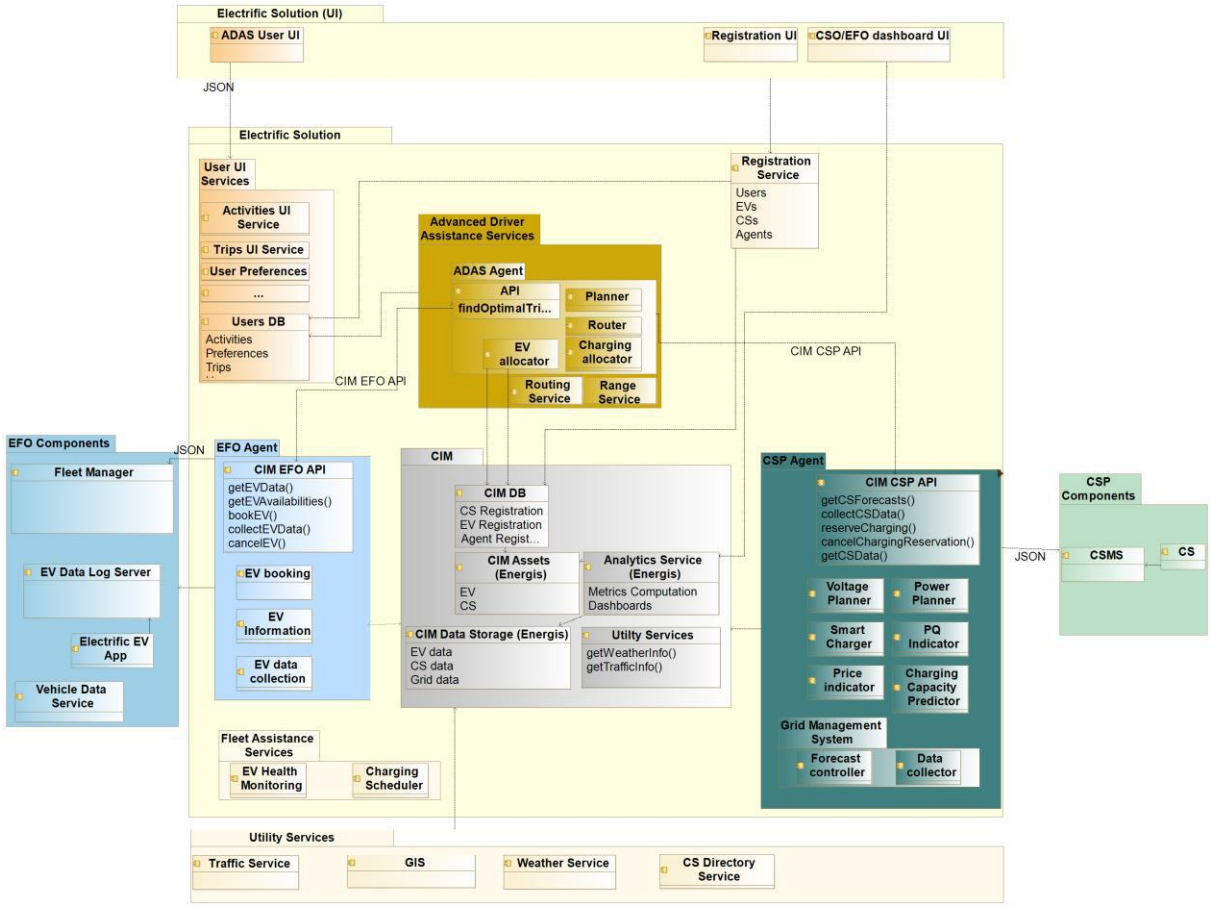

Figure 6. ELECTRIFIC Solution components.

In addition, this diagram shows modules that are not designed in this first iteration: Registration Service, Fleet Assistance Service, and Utility Services (in beige).

These different modules and components are described in the next sections.

IV.2. User interfaces and associated services

IV.2.1. User interface

The Human Machine Interface (HMI) is concerned with how users can interact with the ELECTRIFIC solution. Three different UIs are defined, namely the ADAS User UI, the EFO UI, and the CSP UI. While the first is part of the first iteration of the Electrific project, the other User Interfaces will be developed in later phases of the project.

IV.2.1.a. ADAS User UI

The ADAS User UI is exclusively used by EV users. As described earlier, these can be EV owners and EV fleet users.

As described in D2.2, the EV User is connected with two main use cases, namely ”Plan and navigate trip” and ”View EV Status Information”. The latter is not included in the first iteration of the project whereas the second one is majorly considered.

An EV user can login, enter activities, specify an EV type (for fleet users), plan trips, reserve cars as well as charging stations, and choose the best trip and EV suiting his/her needs from a list of suggestions provided by ELECTRIFIC. Additionally, the user can check planned trips, start navigation, and get help. Other interactions will be included in later phases of the project (see Section II.1.1.).

The ADAS User UI is primarily developed as an Android application.

The material design guidelines have been followed for designing mockups. The goals for creating the material design guidelines have been to ”create a visual language that synthesizes classic principles of good design with the innovation and possibility of technology and science” and to ”develop a single underlying system that allows for a unified experience across platforms and device sizes.”[2] Following these guidelines therefore leads to a unified user experience on all devices and an overall good usability by making use of layouts and components which are well known to the user.

A more detailed description on how this is considered in the ADAS UI will be illustrated in D6.1.

IV.2.1.b. EFO UI

As the name indicates, the EFO UI is used by the EV Fleet Operator (EFO). It offers the possibility to interact with the system as described in Section II.1.2. As the EFO UI is not developed in the first iteration of the project, mockups and more detailed descriptions will follow in later deliverables of WP3.

IV.2.1.c. CSP UI

Besides the EV User and EFO the charging service provider (CSP) needs to interact with ELECTRIFIC in the way described in SectionII.1.3. However, equivalently to the EFO UI, the CSP UI is not developed in the first iteration of the project and therefore mockups and more detailed descriptions will follow in later deliverables of WP3.

IV.2.2. User Services

The ADAS UI app is an Android application which supports every version of the SDK, starting from Android SDK 4.3 (API level 18, Jelly Bean) and up. Support for this specific version is needed to support the E-WALD TomTom hardware devices (which are in use today) during the trials.

The application offers support to later versions of the Android SDK (4.3+) by using the **Android Support Library (ASL)**. This library supports multiple API versions and offers a standard way to introduce new features on earlier versions of the SDK.

All business-related logic of ADAS UI app resides on the server and is made available by a microservices backend system. The goal is to keep the UI layer as thin as possible.

Everything related to user authentication and preferences will be handled by dedicated services. To support the functionality of the first prototype, 4 different microservices will be developed. These microservices all have a separated API which will be called by the ADAS UI app:

- User Service: Handles authentication, preferences, and possible user’s Green Score that could be used to score the greenness behaviour of the EV user (exact definition

[2] https://material.io/guidelines

will be done in D6.1). For the first iteration, users and their preferences will be predefined, no private data will be used.

- Trip planning Service: Returns a list of possible plans calculated by ADAS AI, based on the users' inputs. For research purposes, information on how the user navigated through the application, to make the trip plan request and other choices, can be included to the service on request.
- Trip Service: Handles all operations regarding trips chosen by a user from the suggested ADAS AI alternatives. The main ones being trip selection, cancelling, and listing of upcoming trips.
- Activity Service: Handles all operations regarding the managing of user defined activities.

IV.3. ADAS Agent

The ADAS Agent consists of the ADAS AI, consisting of the four components explained in the following sections, and two utility components explained in Section ADAS Agent UtilitiesIV.3.6. One ADAS Agent represents one EV driver and behaves on behalf of her or him in communication with the CSPs and EFOs either represented by their CSP and EFO Agents or directly using the user's accounts (esp. when the CSPs and EFOs are outside of ELECTRIFIC).

IV.3.1. ADAS AI

The ADAS AI is the core component of the ADAS Agent (see Figure 7). It generates EV routes with charging intervals together with a day plan for the EV user. The ADAS AI takes into the account the preferences the user has in form of optimization criteria. On behalf of the EV user, it communicates with the CSP Agents and the EFO Agents to get various service proposals of the charging slots and EV bookings respectively. These proposals are analysed and based on them, the ADAS AI prepares the best solution day plans, routes, and charging. These are proposed to the user via the ADAS User UI. The list of CSP and EFO Agents is retrieved from the Common Information Model (CIM).

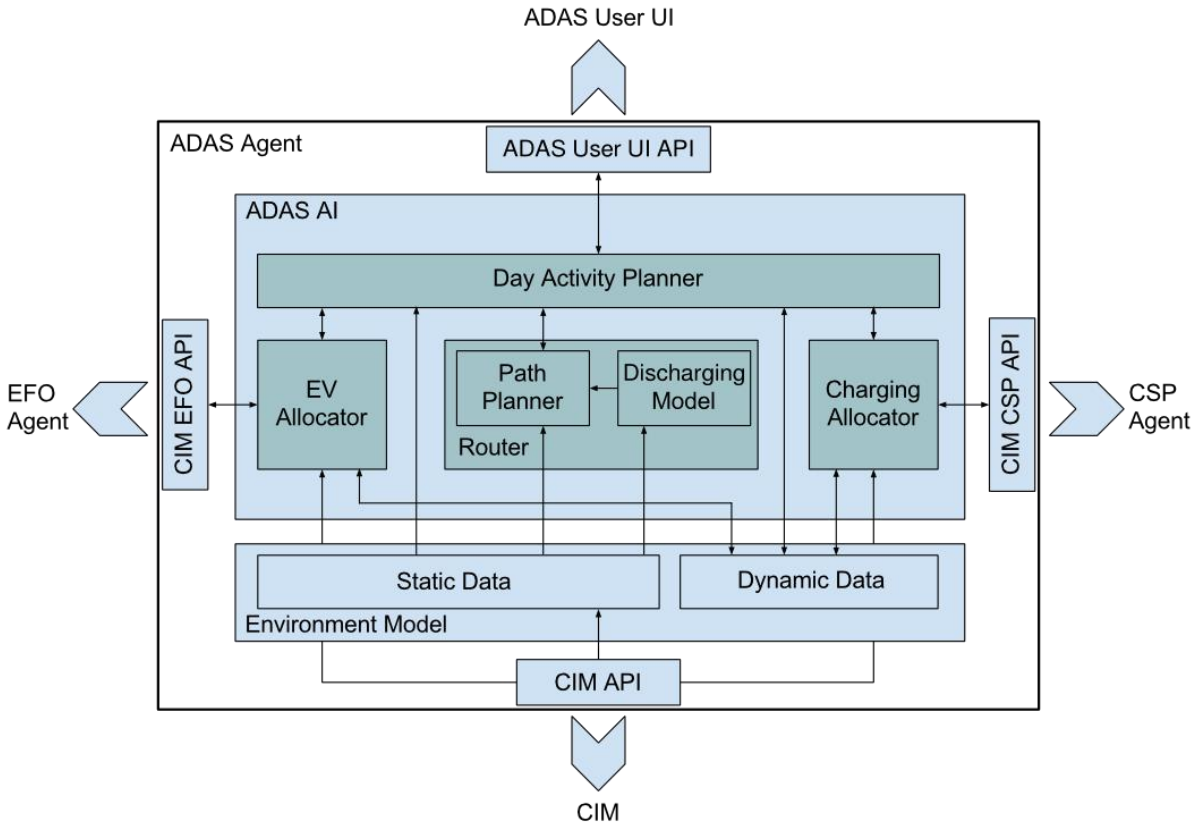

Figure 7. ADAS Agent.

The ADAS AI component consists of four building blocks. The Day Activity Planner provides the day planning functionality based on the set of user activities requested by EVU and orchestrates the other three blocks. The Router, using the Path Planner, generates possible paths between different locations and evaluates them using an internal Discharging Model (see Section IV.3.3. . The Charging and EV Allocators request the CSP and EFO service proposals respectively and provides them to the Day Activity Planner such that it can consider combinations of them. The Charging and EV Allocators communicate with the other agents and components of the ELECTRIFIC Solution via the CSP, EFO and ADAS AI APIs.

The ADAS AI uses a model of the environment which represents the current knowledge of the ADAS Agent. The Static Data contains rigid parts as the road graph, locations of the charging stations, parameters of the Discharging Model and the map, and hierarchy of the points of interest. These data are retrieved from CIM or imported directly using the ADAS Agent Importer (see Section IV.3.6.a.). The Dynamic Data contains current view of the environment the agent currently knows and the day planner uses during planning, e.g. the prices at charging stations, availability of EVs for renting and others.

In the following sections, we will describe the four ADAS AI blocks in more detail.

IV.3.2. Day Activity Planner

To prepare the day plans, routes and charging intervals, the Day Activity Planner searches through a space of possible orderings and time slots of the activities requested by the user. The input of the user is processed by the ADAS User UI and send to the ADAS AI via the ADAS User UI API (see Section IV.3.7.). The planned activities must comply to the user requirements and their ordering and timing must allow moving between them, be it by an EV or by walking. Usage of an EV is possible only if the EV is booked before and its battery is appropriately charged during the day, such that the SoC does not go below the predefined threshold. The charging slots at the charging stations must be reserved during the plan. The planning algorithm will be based on forward-chaining search in the space of possible future states. A state describes possible future status of the EV (its SoC and location), location of the

user and reservation of charging intervals and EV bookings. Each of these parameters acts as an optimization criteria (duration of driving, price of charging, and price of EV rental). Together these criteria will be optimized by the search in form of multi-criteria optimization. The search will heuristically select a combination of the criteria minimizing the overall value of the plan reflecting the user's preference. This principle will be described in detail in the deliverables D7.1-D7.3 in following months, incl. technical details of the search algorithm, heuristics, and optimization criteria.

IV.3.3. Router

Assessing change in SoC on routes between the activities ordered by the Day Activity Planner goes hand in hand with generation of best such routes. The Router will use dynamic programming and search techniques in space of possible locations in the road graph. The Discharge Model supplements the search algorithm with foreseen decrease of the state of charge of the battery. The Day Activity Planner will query the Router to verify whether moving between two activities is possible, for what change in the SoC and over what sequence of locations in the map.

The range prediction included in commercial EV's drivers assistance services mostly considers the energy demand within the previously driven distance. The Discharging Model as part of the Router, in contrary, uses topological information provided by the Geographical Information Service (GIS) to predict the drivable range with regards to the current SoC. This range information will be on the one hand displayed to the user through the ADAS User UI to get an overview of the current driving situation and on the other hand, it is used for the routing service.

Based on the Discharging Model's predictions and the driver's priority chosen in the reservation process for an EV, the Router will determine the optimal route to the destination. Hence, the Router needs to optimize regarding speed, cheapest route, and greenness. Therefore, the service will consider a large variety of input data, such as geographical data from GIS to process route properties (e.g. distance), traffic and grid information as well as forecasts.

IV.3.4. EV Allocator

The allocator blocks are responsible for requesting available service proposals from CSP and EFO considered by the Day Activity Planner to be used in the built day plans. The EV Allocator deals with getting available EV offers from different EFOs, their price, SoC, and additional properties as defined by the user (EV brand, engine parameters, etc.). As the Day Activity Planner is considering more day plans simultaneously, the EV Allocator builds for it a view of the currently proposed offers in the Dynamic Data in the Environment Model, which can be efficiently used by the Day Activity Planner during its search. Although the final process of EV booking goes outside of the ADAS AI, the user will be provided with a booking identifier to easily finalize the booking process in the EFO's booking system.

IV.3.5. Charging Allocator

The Charging Allocator provides similar functionality as the EV Allocator, but with the charging slots as the requested service from various CSPs. The parameters requested are price, power quality and percentage of renewables. Based on the user preferences these are considered by the Day Activity Planner to provide plans combining them as requested. Additionally, the parameters are later used to visualise the qualities of the plans to the user via the ADAS User UI. Similarly, as with the EV booking, the charging slots are reserved outside of the ADAS AI by the user via a provided charging slot reservation identifier.

IV.3.6. ADAS Agent Utilities

IV.3.6.a. ADAS Agent Importer

The agents use both dynamic and static information about the environment. The ADAS Agent Importer takes care of loading the static part of the environment to the ADAS Agent from the database or cached local files, especially for debugging purposes, when the retrieval from CIM is not appropriate. The particular data are (i) a directed road graph in form of a set of locations describing junctions and a set of roads between the junctions; (ii) the Discharging Model (subsuming the Range Service) as a part of the Router in form of a function giving discharge rate per hour for an EV type, road parameters and speed profile (e.g. fast, normal, slow); (iii) a charging model as a part of the Day Activity Planner in form of a function giving a charge rate per hour for an EV type and charging speed; and (iv) a set of points of interest, their locations and hierarchy. The Points of Interest (PoIs) are possible geographical locations of interest of the users, e.g., markets, cinemas, kindergartens, etc. The PoIs are naturally described by a hierarchy, e.g. a PoI “market” subsumes all particular markets at concrete locations in the target area of the ELECTRIFIC Solution.

IV.3.6.b. ADAS Agent Visual Inspector (AAvis)

The AAvis component will be a part of the ADAS Agent. The main goal of AAvis is to provide easy to use visualization tool for debugging the ADAS Agent and ADAS AI. The component will visualize plans (output of the ADAS AI) in space and time together with input planning problem(s). In the future, this component will allow to visualize more plans at the same time and analyse their interactions at the charging stations and possibly in the route graph.

IV.3.7. ADAS AI in Context

The ADAS AI component interacts with other parts of the ELECTRIFIC Solution (see Figure 8), as we outlined in the previous sections. The following scheme shows those interactions in context, particularly as bidding of the service proposals between the ADAS Agent and CSP and EFO Agents. The CSP and EFO Agents contain components paired with the Charging and EV Allocator respectively in the ADAS AI. These components prepare the service proposals, which are used by the Day Activity Planner in the end.

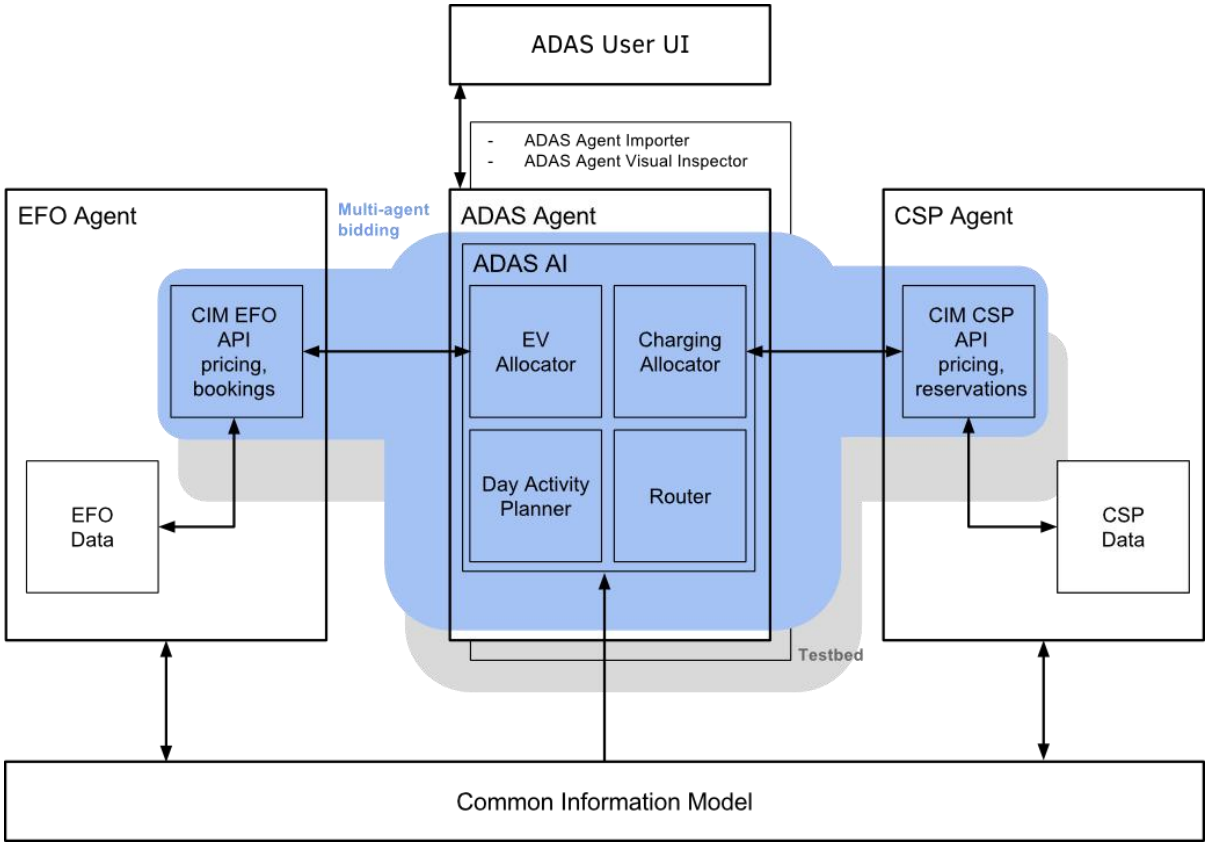

Figure 8. ADAS AI in Context.

The ADAS AI API *planning request* describes the input from the user. The API call is represented by findOptimalTrips. The parameters of the call are (i) an initial position of the user/driver; (ii) preferences of the user/driver (price, time, percentage of renewables); (iii) set of activities the user wants to carry out. Each activity is defined (i) by a set of intervals when the activity can be fulfilled, (ii) by a duration how long the activity will take; (iii) whether the activity can be splitted to separate intervals; and (iv) what poi categories (incl. subcategories based on the PoI hierarchy) are eligible for the activity.

The ADAS AI API *planning response* contains the planned day plans, including routes, chargings and the other actions the user should take. Each response is a sequence of actions. An action is defined (i) by its starting time and (ii) by its duration. The possible types of actions are walk, claim, disclaim, drive, charge, and doActivity. Walking is described by a sequence of locations over which the user should go by foot or by other transportation mode.

Based on an identifier of an EV and reservation with an EFO, the user can claim the reserved EV. A claimed EV is such that no one else can use it before it is disclaimed, but the user does not have to be physically in the car when it is claimed (e.g., it can be charging, or the user is doing an activity in parallel). Disclaiming the currently claimed EV lefts the car for another user to claim it and use it. The claimed EV is driven over a several locations, technically describing the route for the user. The claimed EV is charged at a reserved charging slot (based on a charging slot and charging reservation identifiers). Finally, the previously requested activity is done by the used via the doActivity action at the predefined PoI in the planning request.

The CSP and EFO APIs are described in their respective sections.

IV.4. CSP Agent

The Charging Service Provider (CSP) agent is the connecting point between different external systems. It exposes its service by an API (for example to the ADAS) and uses the services of others (Grid Management System and CS Management System). The main goal of the CSP

agent is responding correctly to the different constraints from the grid (e.g. available power) to help the DSO in case of a degraded grid status.

Making or calculating forecasts about the low voltage grid is a fundamental assumption of proving the correctness of the functionality of CSP agent. Hence, the feasibility of this assumption and all other technical details of this component will be included in D4.1.

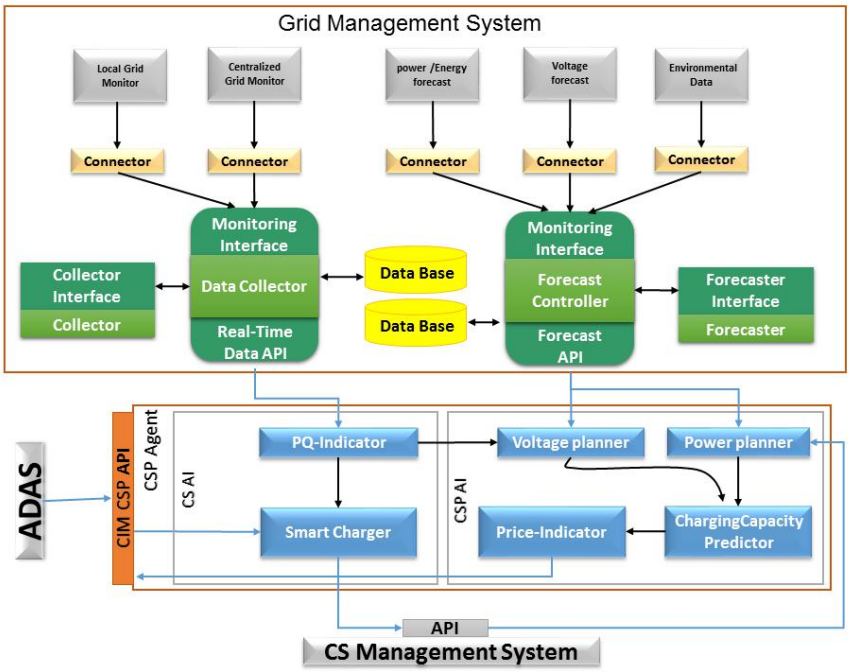

Figure 9. CSP Agent.

Furthermore, the CSP is targeting to increase the renewable intake in the charging stations' consumption which in turn can increase its profit. The energy price (kWh) is very cheap from local source in compare to the energy from the grid. In the future, Time of Using (ToU) pricing scheme can encourage consuming during renewables peaks.

The following sections describe all components of the architecture shown in Figure 9.

IV.4.1. Grid Management System

The Grid Management System contains two main parts, the Forecast Controller, and real-time Data Collector.

The Forecast Controller is required to collect forecasts about the different power sources (local source or grid) in a certain time period. On the one hand, this forecast contains information about the renewable percentage in the power mix from the grid and the available cable (feeder line) capacity connecting a charging station to the transformer substation. The predicted nominal voltage value in the low voltage grid where the charging station (CS) is located can also be part of the forecast. On the other hand, the Forecast Controller should provide predictions about the available renewable energy from the local renewable sources. These predictions may be requested via the Forecast API. To serve these requests, the Forecast Controller needs information about future power availability, which may either be delivered directly by a third party (e.g., a power supplier) or be forecasted locally based on environmental data. In case the forecasts are not available directly, but environmental data is required to calculate them (e.g., about locally installed solar panels), the Forecast Controller will exchange this data with the Forecaster via a push or pull mechanism. The algorithm inside the Forecaster will use this data (maybe also additional available data) to provide a forecast of the availability of renewable power using a forecast model. This forecast will be stored in the database. The Monitoring Interface at the Forecast Controller offers a unified interface to the different parties

to submit their data in the future. For legacy party, a connector architecture will adapt the specific implementations to the interface.

The second part of the Grid Management System is the real-time Data Collector. This component is connected to the measuring devices, which are either installed locally at the CSs or centrally at the transformer substations (by the DSO). The Data Collector introduces an API to serve requests from the CSP agent about the real-time PQ-measurements, which are used by the Smart Charger. The Data Collector should be designed in a way, such that collecting data from differently manufactured measuring devices is supported and the data collection from a significant high number of measuring devices is supported. This corresponds to Objective 2 of the ELECTRIFIC project which aims to improve interoperability (cf. Deliverable D2.1).

IV.4.2. Charging Station Management System (CSMS)

The Charging Station Management System (CSMS) allows the CSP agent to keep an eye on all tasks and processes concerning the charging stations. It communicates with the CSs using some standardized protocols (e.g. OCPP protocol) and can retrieve the current CS status (e.g. available or unavailable). The CSMS manages the actual CS reservations and keeps track of all reservations that will be handled by a certain CS. Furthermore, the opportunity to remotely start or stop a charging process and information about ongoing charging processes are provided. All the needed information can be requested by the Charging Capacity Predictor to build the forecast of the available capacity.

Additionally, the Smart Charger can change the charging profile of a CS during the charging process in case the PQ-indicator detects grid instabilities (e.g. increasing harmonics, flickers). The CS Management System must provide an API method for that request.

IV.4.3. CSP AI

The CSP AI represents the logic of the CSP agent. On the one hand, it is responsible for providing forecasts about the availability of CSs during a certain time window to other components and proposing the suitable pricing scheme. It contains multiple modules which contribute to achieving the end goals of the CSP agent.

IV.4.3.a. Power Planner

The Power Planner is responsible for giving an overview about the aggregated available power at a certain CS with a renewable percentage. It uses the power forecast from the Forecaster (coming from Forecast interface: Grid, Micro Grid, Local Plants). The available capacity from the grid is limited to the maximum available cable (feeder line) capacity between the CS and the transformer substation during a certain time slot. The percentage of renewable energy refers to the percentage value drained from the grid by the CS and is the composition of any kind of renewable source mix (from Wind, PV, etc.). The available capacity from the local renewable energy plant (e.g. local photovoltaic plant) is limited by the maximal power produced by the plant and the percentage of renewable is equal to 100%.

Furthermore, the Power Planner should distribute the available capacity between the CSs sharing the same feeder line (i.e. located at one charging spot). This distribution should be fair and respect the maximum capacity of each charging station and the already booked capacity by each charging station.

IV.4.3.b. Voltage Planner

The Voltage Planner is responsible for estimating the grid status in the next time period depending on the prediction obtained from the Forecast Controller via the API. In the first iteration of ELECTRIFIC, we will use forecasts about the nominal voltage value in the low voltage grid which is currently the only possible PQ-measurement that can be predicted. The reason behind that is, voltage is directly affected to the load, but the other PQ-parameters not necessarily. In other words, what we know about a low voltage network is the demand or feed

in, but we have no specific information about which concrete electrical devices are in this low voltage grid, at which times they are used and for how long.

The output of the Voltage Planner is a value between 0 and 1. The outcome 1 means no problem and the maximum available capacity can be used, whereas 0 means no possibility for charging at all. We can estimate the power quality in many manners. For example, if $PQ_{Voltage}$ is between 0.8 and 1.0, then the power quality is noted as Excellent, if it is between 0.6 and 0.8 it is Good, if it is between 0.4 and 0.6 it is Medium, and if it is smaller than 0.4 the corresponding power quality is considered as Bad.

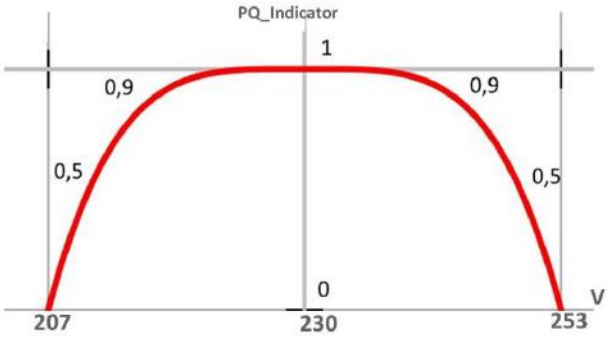

Figure 10. PQ indication.

Figure 10 depicts calculation of PQ_indicator using only Root Mean Square (RMS) voltage value. Small deviation from the normal value (230 V) does not have a larger effect on the grid than a higher deviation. For that case, some exponential adjustment will be used, which can be different depending on where the power quality value is measured.

IV.4.3.c. Charging Capacity Predictor

The main task of the Charing Capacity Predictor is to provide information about the availability of the different connectors at the charging stations in conjunction with a list of charging options based on the available power capacity from the Power Planner and the PQ forecast from the Voltage Planner. The charging options can have different charging capacities with different prices and percentage of renewables. In example, the available power capacity from the grid (retrieved using the Power Planner) is decreased in case the power quality (retrieved using Voltage Planner) is at a bad level.

IV.4.3.d. Price Indicator

The Price Indicator is a very important part of the CSP AI. It will help encouraging the end users to choose the most suitable charging option from the perspective of the CSP. The output of the Charging Capacity Predictor will be used as an input to determine the best price of each proposed charging option depending on three factors: renewable percentage, the accuracy of the forecast (near to the reservation time, more accurate), and the grid status. The price will determine how much the user must pay for a given time slot with the chosen charging option.

IV.4.4. CS AI

The CS AI is responsible for ongoing charging operations. Each CS is represented by a Smart Charger and the corresponding PQ Indicator. Both components are needed for the so-called Smart Charging Process.

IV.4.4.a. PQ-Indicator

The PQ-Indicator represents a component that tries to find a conclusion about the current power grid state in the certain area around the CS. Based on the results of the PQ Indicator, the Smart Charger will decide whether the individual desired charging process can be done

without restrictions or not. For the determination of the current power grid state, PQ-data of the installed measuring devices (transformers, charging stations) are recorded periodically (e.g. every 30/60 seconds) in high resolution (e.g., 3 seconds average values). For various PQ parameters, e.g. voltage, frequency, or harmonics, values between 1 and 0 are determined. Afterwards, these results will be consolidated into a further calculation step. The final value "PQ_Total" represents the current power grid status in a certain area.

The value 1 corresponds to no problems and the loading process can be performed without any restrictions.

The value 0 reflects the worst case in the grid and indicates that a charging operation is currently not tolerated/possible.

All other values between 0 and 1 indicate how far the PQ parameters move away from the ideal state and give information about how much of the available capacity can be consumed at the moment.

E.g. PQ_total = 0.9 * available capacity = 0.9*max_available_capacity

Figure 11 shows the idea of how the PQ Indicator should work. First, all needed measured data is retrieved from the database of the Data Collector and then KPIs are applied to them. The result is a value between 0 and 1 for each KPI (e.g. voltage, frequency, harmonics, …). The overall "PQ_Total" value is then computed out of the different outcomes of the KPIs.

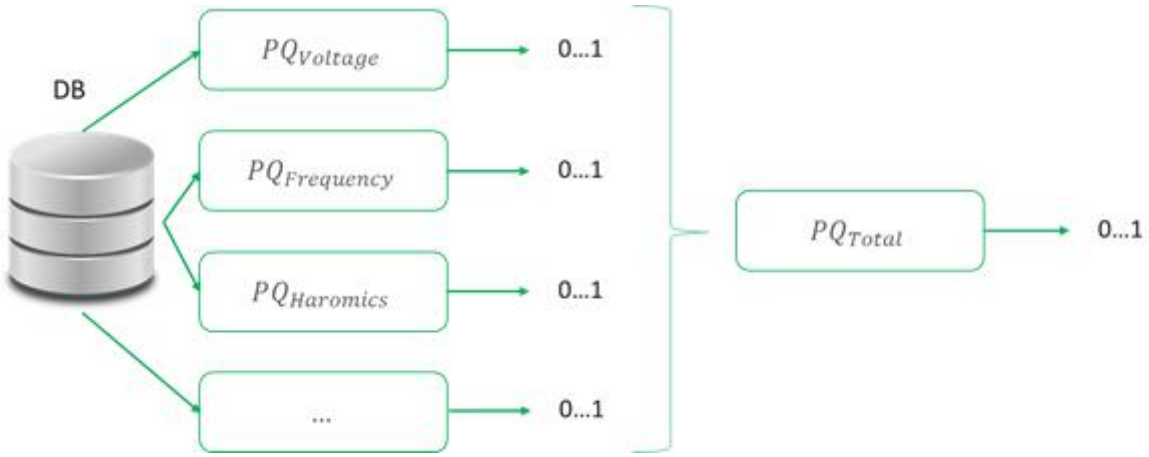

Figure 11. Power Quality.

IV.4.4.b. Smart Charger

After receiving the final request of a reservation from ADAS which includes the chosen charging option, the required energy, the CS of the reservation, and the time period of the reservation, the Smart Charger will create a charging profile containing information about the maximum amount of power that can be used during each time slot, the price and validation time of this profile. This charging profile is then submitted to the CSMS. Additionally, the actual reservation is being performed.

The second part of the Smart Charger is based on responding to the notifications from the PQ-Indicator concerning differences between the predicted PQ-parameters and the current ones. Based on the output of the PQ-Indicator, the charging capacities (used power) should be changed and a new charging profile should be sent to the CS.

As a third part, the Smart Charger tries to fulfill one of the users' concerns:

- Booked Energy: The Smart Charger tries in any way to provide the booked energy until the end of the charging process. With that option, the Smart Charger has the freedom

to allocate different charging rates freely during the charging process however respecting the overall amount of booked energy.

- State of Health: The Smart Charger does not exceed the maximum booked capacity (for example 40 kW for 15 min) during the whole charging process. In this example, the maximum booked capacity can only be shifted in time, but only for a timeframe of 15 minute 40 kW can be used.
- Greenness: Maximize Ren% in the booked time with the help of real time data and updated forecasts before and during the charging process.

IV.5. EFO Agent

In the following, the structure and functionalities of the EFO agent and components are explained. The EFO agent provides a link between existing components of THD, the EFOs and the ELECTRIFIC backend. The main use cases of this link are the EV booking, supply of EV properties and the collection of driving data. These functionalities are used by the ADAS and CIM via the CIM EFO API. For the supply of the described data, the EFO is connecting to the EV Data Log Server and the specific fleet manager software of the EFO.

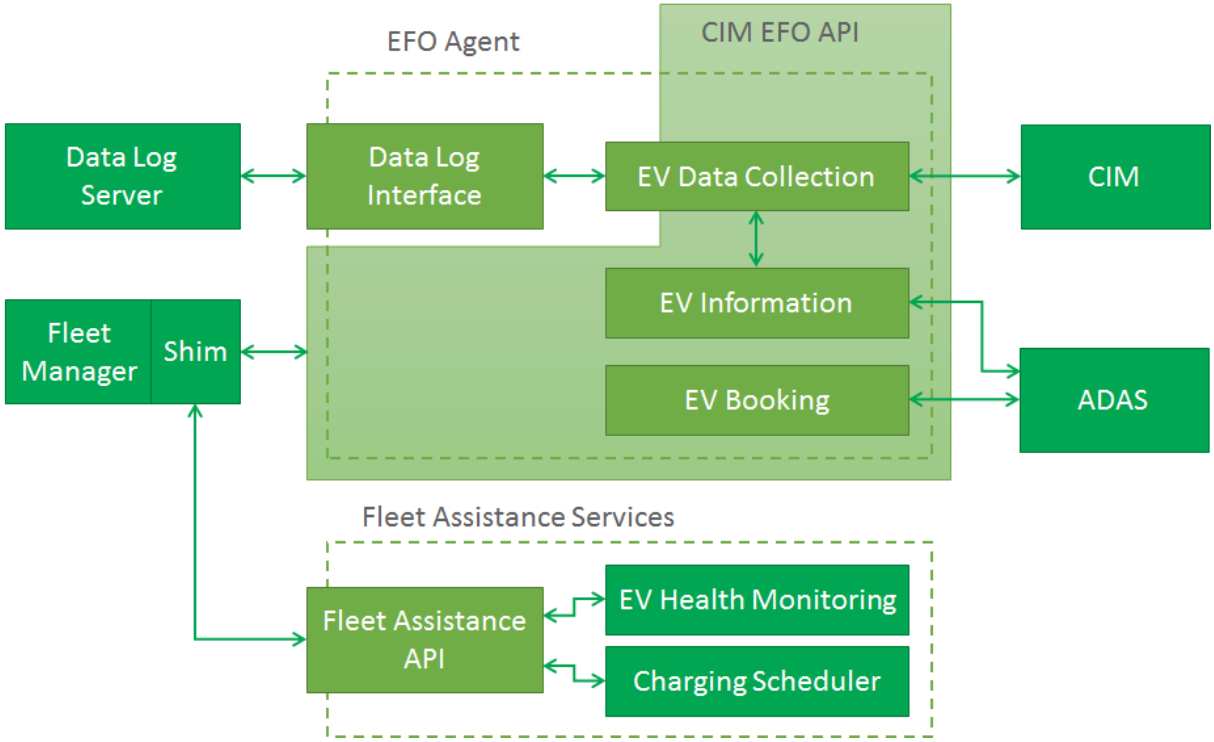

Figure 12. EFO Agent and simultaneous data flow to the Fleet Assistance Services.

The interface to the Data Log Server is realized by a TCP bound MySQL connection, which will query the Data Log Server for new data regarding EVs that shall be observed. Therefore, the CIM EFO API provides functions to start and stop the data collection for certain EVs. The fleet managers of the trial partners are interfaced through the CIM EFO API, which is a handler component of the EFO Agent. Figure 12 depicts the CIM EFO API as a rectangular shape containing the EV Data Collection, Information, and Booking. To access the fleet manager specific APIs, shims (explained further) need to be implemented by the trial partners. The shims fulfil the requirements of the CIM EFO API. Additionally, the shims need to support the Fleet Assistance API, which enables the link to the EV Health Monitoring and Charging Scheduler.

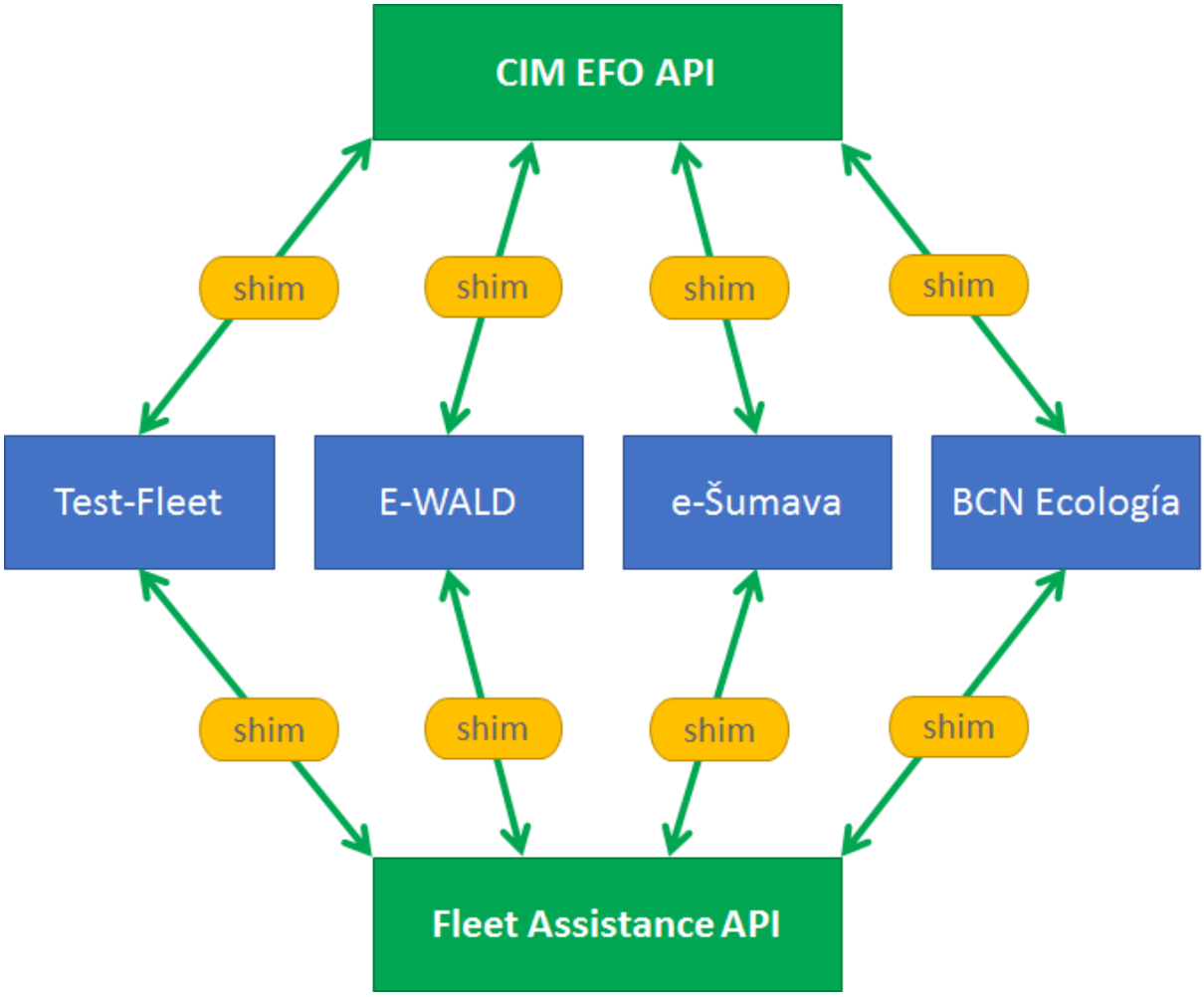

Figure 13. Shims between the Fleet Manager systems and the EFO Components.

A shim is a small component, that connects two different APIs, without logic in the best case, but for some use cases a small logic is needed to fulfil the functionality. It handles the operation by itself and gives the possibility to connect an existing Booking System to the EFO Agent. Shims are typically developed by the participants to fulfil the connection to the API of the host system, which is in this case ELECTRIFIC. These shims are depicted in Figure 13.

Since ADAS and CIM will need information (e.g. SoC, SoH) from the Data Log Server and the separate fleet manager implementations based on specific situations, the information from the data sources need to be routed in a certain order. The CIM needs EV properties and driving information that is collected by the ELECTRIFIC EV App. Therefore, the EV Data Collection within the CIM EFO API is needed. To prevent the CIM to receive driving information from every EV, only the information of EVs, for which the data collection is activated, will be filtered out and sent to CIM.

ADAS requires information about EV properties and access to EV booking. For this task, information from the Data Log Server and the specific fleet manager is needed. EV availability data, SoC level, etc. is put together by EV information functions, taking the EV Data Collection as the source. The actual EV booking management is done in the EV Booking functions.

IV.5.1. Fleet Management System

A Fleet Management System is needed to organise the fleet. It provides different functionalities on booking of EVs, billing, and viewing EV status information. The Fleet Management System will be independent from the ELECTRIFIC Solution itself, must be provided by the EFO and give only the access to use all needed actions. Depending on the actual fleet management

system, maps showing the available EVs at their current location or other functionalities are included.

Some extensions to the common used types of Fleet Management Systems must be implemented to reach the goal of minimization of the battery health degradation. Therefore, Fleet Assistance Services will be developed and they need functionalities like reserve or blocking an EV for charging or send a request for reallocation of the EVs home location based to the battery health and the usage by EV users.

IV.5.2. EV Datalog Server

The current EV driving data is being stored on the EV Data Log Server. This way, physical models, such as a battery health metric, can be derived from data of multiple, independent EVs and trips. By storing current driving data in the EV Data Log Server, the data can be simultaneously used by ELECTRIFIC backend services, such as CIM, enabling constant system operation. The data storage is needed to calculate some parameters of the unique EV, e.g., the battery health, the energy consumption for trip planning, or state of charge (SoC) for charging prediction.

IV.5.3. ELECTRIFIC EV App

Driving data is read by the Vehicle Data Service from the EV and then handed to the ELECTRIFIC EV App. From there, the data is then transferred to the Data Log Server. Also, the current data is used by the ADAS UI to display the SoC value, and other information.

IV.5.4. Vehicle Data Service

The Vehicle Data Service is a software abstraction for the Bluetooth OBD Adapter. It handles the communication setup and teardown, as well as the data transmission. This way, the EV data will be usable in the tablet software and the ELECTRIFIC backend services.

IV.5.5. Fleet Assistance Services

In this section, the purpose and the functionalities of the fleet assistance services are explained. For a clearer understanding of these explanations, Figure 14 depicts the components and the data transmission.

Information from the current and historical fleet data from the Data Log Server, as well as the grid forecasts from the CSP Agent are taken as parameters for further optimization processes. In case of the grid information from the CSP Agent, the Fleet Assistance Services optimize the utilization of renewable energies and reduce load peaks due to charging. The fleet data is taken by the EV Health Monitoring to classify the health states of the EV batteries and then create suggestions for the Charging Scheduler to consider batteries in bad health state for slow charging. Also, the battery health states shall be displayed to the EFO, which is done in the EFO dashboard. The communication between EV Health Monitoring and Charging Scheduler is kept within the EFO Components.

The charging scheduler takes the current booking schedule of the Fleet Manager system to make decisions on a short, mid, and long-term basis. This influences the type and schedule how the EVs are charged. Thus, the Fleet Manager will receive a new schedule containing bookings that are used for EV charging. The long-term decisions of the Charging Scheduler are mainly aimed to influence the location of the fleet EVs. This means that frequently booked EVs of one location are suggested to be swapped with EVs of other locations, which are mostly unused. These suggestions are displayed to the EFO in the EFO dashboard, enabling the EFO to instruct his employees to swap the EVs.

To enable the communication between the Fleet Assistance Services, CIM EFO API and Fleet Manager, a shim needs to be developed by the EFO, which adapts to the specific Fleet Manager implementations to provide data and functional access. This shim does not need to

be developed separately for each connection as depicted in the following diagram. Instead, the Fleet Manager can be expanded by a single shim which fulfils the connection requirements. As shown in Figure 14, the shims need to provide and accept different data structures. Here, x contains information about the booking as well as charging schedule. x' is a substitute terminology for the information, the ELECTRIFIC backend services will need. This information contains static and current EV properties and functionality to book an EV.

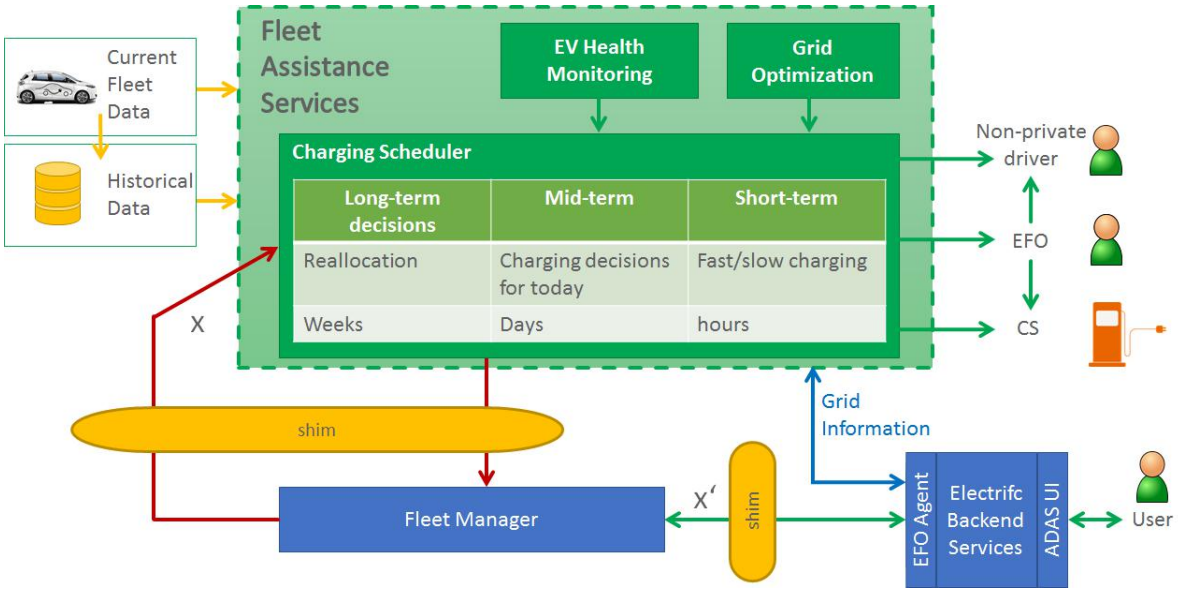

Figure 14. Fleet Assistance Services.

The requirements for the shim include the supply of EV information, i.e. manufacturer and model information, EV availability and pricing, and access to the booking system. The latter contains functionality to retrieve the bookings from the booking system, book or block EVs for charging through the charging scheduler, and to book EVs for actual trips planned in the ADAS UI. To guarantee more details in the ADAS UI, the pricing information of a booking shall be provided, so that users can request the billing for their bookings.

IV.5.5.a. Charging Scheduler

One main component of the fleet assistance services is the charging scheduler for the EVs of a vehicle fleet. It uses the Electric Vehicle Smart Algorithm (ELSA), which is to be developed in the further project course. Its purpose is the scheduling of the charging of multiple unbooked EVs at the home location. Also, it shall control the charging process. These requirements serve the goals to avoid load peaks in the grid and the minimization of battery ageing.

ELSA must consider different constraints to allow smart charging in a real-world application. These are the available power of the grid, the available charging stations at site, and respectively, the SoC levels of the EVs in the fleet. Additionally, the battery health of the EVs, which is classified by a EV Health Monitoring system, is taken into account to realize the battery ageing minimization. Therefore, the EV Health Monitoring will make suggestions to the Charging Scheduler in order to minimize battery ageing, which the Charging Scheduler will take as input in order to make the optimal charging decisions.

As a subcomponent, an EV Reallocator will order an EFO to reallocate certain EVs to specific home locations. This is based on a battery health metric, which allows the classification of EV batteries. Depending on the battery state and the EV usage, the EVs will be swapped at different home locations over time, guaranteeing a homogenized usage of the EV fleet.

IV.5.5.b. EV Health Monitoring

Another functionality of the fleet assistance services shall be the EV health monitoring. It uses a battery health metric, which is used to classify the health state of an EV battery. This

classification leads to a result of five different health state classes, as shown in Figure 15, making it easier for EV users and EFOs to determine the EV battery health state. To monitor a whole EV fleet, this functionality shall be expanded over multiple, different EVs.

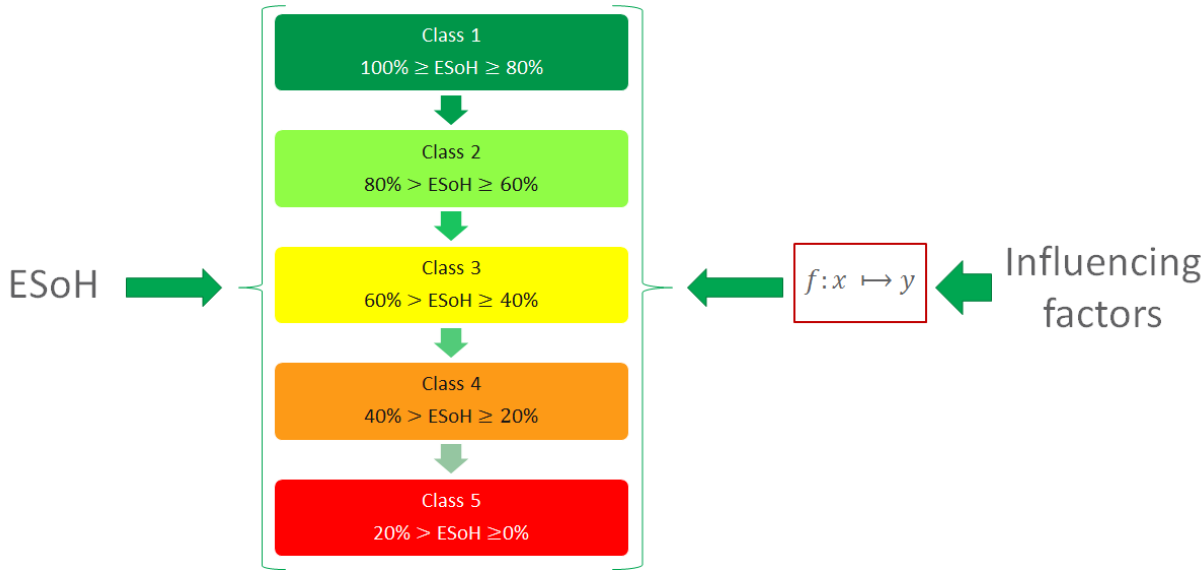

Figure 15. ELECTRIFIC SoH (ESoH).

The battery health metric is based on battery ageing models. Therefore, research must be done in the following iterations to determine, which battery influencing factors correlate with the ageing of batteries. Instead of the manufacturer SoH, which is not publicly defined by EV manufacturers, a universal parameter, the ELECTRIFIC SoH (ESoH) is defined. This parameter shall be applicable with every, commercially available EV.

IV.6. Common Information Model

The CIM consists of the CIM database (CIM DB), the CIM data storage, the Analytics service, and the Utility service. CIM defines the different APIs between the agents interacting in the ELECTRIFIC solution.

The CIM Database is used to store the information of the assets and agents during the registration process (user registration data are not stored in the CIM).

The CIM is also used to store and compute the metrics concerning the assets. The storage (CIM Data storage) and analytics tools (Dashboarding and analytics service) will be provided by the Energis software of Freemind.

The next section presents the assets managed in the ELECTRIFIC Solution; then the CIM APIs are explained followed by the Energis software. The last section mentions additional usage of the CIM (utility services).

IV.6.1. CIM Assets

This section descibes all the assets and their main static attributes managed in the ELECTRIFIC Solution:

- Charging area (spot), associated charging stations and charging connectors.
- Feeder line (cable), where a CS is connected to the transformer substation.
- measuring device at transformer substation.
- Local voltage measuring device at the CS.
- Electric vehicle.

- Battery.

All these assets will be registered in the CIM with a unique identifier (asset code), a reference to the agent managing it, and the type of the asset.

A **charging area (spot)** is an asset type that is defined by its coordinates, address and the list of charging stations. A **charging station** is the physical system where an electric vehicle can be charged; it is defined by its coordinates, manufacturer and model information, the parking places, and by its charging connectors. A **charging connector** is characterized by the following static attributes:

- Type of current (AC, DC).
- Charging mode (Mode1... Mode 4, as defined in[3]).
- Charging plug (plugs to provide electrical power for the charging modes it specifies).
- Maximum power for this connector (kW).

An **Electric Vehicle** is an asset type with the following information: Manufacturer, model information and reference to a battery

A **battery** is an asset type with following static attributes:

- Battery Type (NiMH, Li-Ion..).
- Nominal capacity (Wh).
- Nominal voltage (V).

IV.6.2. Metrics catalog

This section details some of the metrics that will be captured in the ELECTRIFIC solution for the different assets (input metrics) and the metrics that can be computed from these input metrics (derived metrics).

IV.6.2.a. Input metrics

Transformer and Local voltage and measuring device

These assets collect the actual voltage (V), frequency (Hz), Harmonics, Flicker([0 .. 10]) at the transformer or local voltage measuring device.

Feeder line (Cable)

For these assets, the following metrics are retrieved via the Forecasting interface (controller):

- Forecast of maximum available power capacity in a time slot (W).
- Forecast of voltage value in the low voltage grid (V).
- Forecast of percentage of power produced from renewable sources (%) (this can also be computed with a forecasting model in case the forecasts are not provided directly).

Charging station

- EV charging measurements (charging type, duration, used power); collected from the CSMS.

Electric Vehicle

- Booking availability of the vehicle.

[3] IEC 62196 – IEC web site (www.iec.ch)

- Expected range of the vehicle (km).
- Price information for the booking (EUR/time).

Battery

- SoH (%), that represents the remaining battery capacity in relation to the nominal battery capacity.
- SoC (Wh).

IV.6.2.b. Computed Metrics

Transformer and Local voltage and measuring device

- Actual power quality (computed in PQ-indicator).

Feeder line (Cable)

- Forecasted percentage of power produced from renewable sources (%) when computed with a forecasting model.

Charging Station

- Forecasted available power capacity (computed in Power Planner).
- Forecasted power quality (computed in Voltage Planner).
- Forecasted renewable percentage (computed in Power Planner).
- Forecasted prices (computed in Price indicator).
- Actual availability of the different connectors (CSMS).
- Actual available power capacity (computed in PQ-indicator).

Computed metrics for ELECTRIFIC

- Accuracy for the predictions of power capacity, renewable percentage.
- Accuracy for the predictions of power quality.
- Accuracy for the predictions of the forecasting models.
- Percentage of renewable used in charging processes.

IV.6.3. CIM API

The CIM CSP API and CIM EFO API are used to define the operations that can be performed with these agents. These interfaces homogenize the data sources of the agents and could be standardized as output of the project.

These different APIs will be implemented as RESTful API using JSON data-intechange format.

CIM CSP API

This API is used to perform the following operations with a CSP agent:

- Search the charging stations located in a certain area.
- Retrieve the static parameters of a charging station.
- Retrieve the forecasted charging stations availabilities.
- Reserve a charging station within a certain timeslot and capacity.
- Cancel a reservation of a charging station.
- Get the current forecasted available energy for the whole reservation.

CIM EFO API

This API is used to perform the following operations with an EFO agent:

- Retrieve available EVs in a certain area and time window, which can be booked.
- Book an EV.
- Cancel a booking of an EV.
- Retrieve the dynamic data of an EV.
- Activate the data collection for a specific EV from the EV Data Log Server to ELECTRIFIC , while enabling a whitelist to only receive parameters, that are needed in further processing.
- De-activate a data collection of a specific EV.
- Retrieve the static information of an EV.

CIM API

This API is used by the different agents to store, retrieve, aggregate and compute time-values series metrics relevant for ELECTRIFIC:

- Store metrics: for each asset, multiple different time-values series can be stored in the CIM. The couple asset-metric identifies a single time-values series.
- Query the different time-values series of an asset.

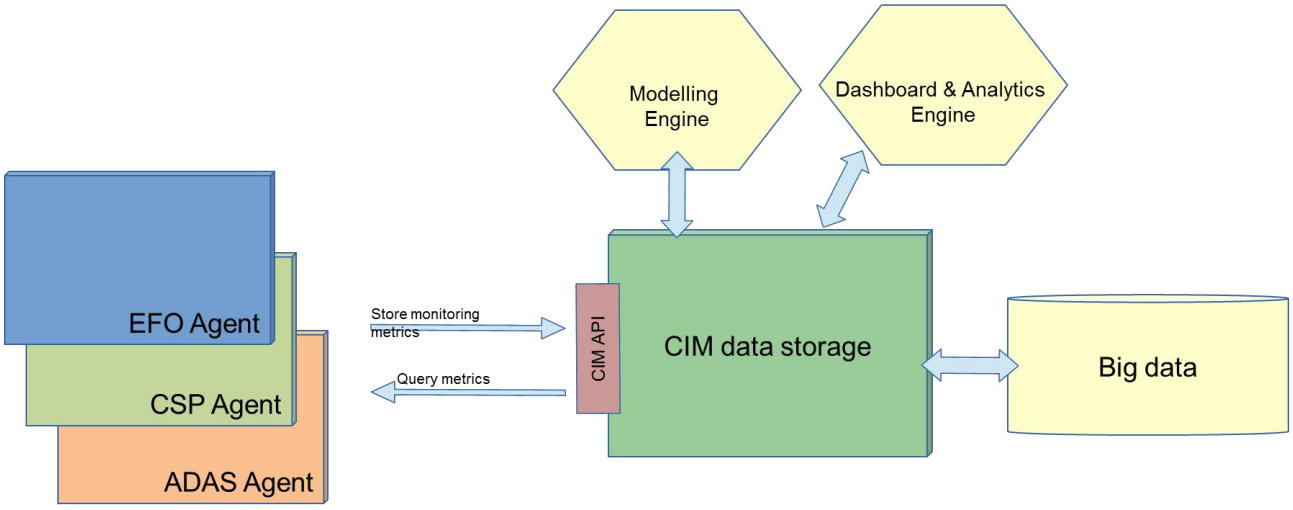

Figure 16. CIM data storage.

IV.6.4. Energis

The Energis software will be used for the The CIM data storage, Modelling Engine and Dashboard & Analytics engine.

CIM Data storage

This CIM Data storage is the component in which all historical, monitored and forecasted metrics used by the ELECTRIFIC solution will be stored.

The Open Source KairosDB/Cassandra is used as Big data repository. It is a very fast and scalable time series database. It allows storing data that can be labelled with tags. Values can be of type long or float.

A metric value is stored as follows in the CIM Data storage:

timeStamp metricName metricValue tagName1=tagValue ... tagNamen=tagValue

The meaning of each different field is described below:

1. timeStamp

The timestamp of a metric can be in the past, present or future. Timestamps are in UTC time.

2. metricName

The value must be unique and must follow a certain naming convention.

3. metricValue

A metric can only have a numerical value.

4. tags

A tag is a keyword that can be stored together with a metric to ease its filtering.

When the metric is stored, the following tags are added

- **assetCode: code to identify uniquely the asset**
- **assetType: cable | measuring device | CS | EV | battery**
- **reference: actual | forecasted**
- **granularity: none|1 minute |15 minute| 1 hour**

5. AssetCode

This code allows to uniquely identify an asset.

6. AssetType

In ELECTRIFIC, the following types will be used: Feeder line (cable), measuring device, CS, EV, battery.

7. Reference

The same metric can be stored with different references that indicate what the metric is used for.

The possibilities are: ‘actual’ for the measured values, ‘forecasted’ for the predicted values.

8. Granularity

An optional tag. If it’s present, it is expressed as "number of timeunit". Timeunit can be "second", "minute", "hour", "day", "week", "month", and "year". When a granularity is specified it means the metric value refers to the time period after the timestamp.

Modelling Engine

Energis identifies models from historical data; the mathematical formula is one element of a model. A model also provides performance criteria to assess the quality of the generated formula. The formula is stored in different formats (Latex, MathML, Java) inside the Energis database. Performance criteria are for example the accuracy, Coefficient of Variation of the Root Mean Square Error (CVRMSE), Mean Bias Error (MBE), R^2.

This modelling engine will be used in ELECTRIFIC for the forecasting models.

Dashboarding & Analytics Engine

Metrics as the ones defined in deliverable 2.1, will be imported or computed by the Dashboarding & Analytics Engine. This engine allows aggregation of metrics and creation of derived metrics based on mathematical functions.

These imported and computed metrics of assets can then be visualized in many display styles (bar, line , scatter plot, pie chart...).

IV.6.5. Utility Services

The CIM is also used as a acces point to some external services that could be required in the scope of the project (traffic information services, weather information services, charging station directories, local renewable forecasting...).

V. DEVELOPMENT AND TESTING

V.1. Methodology

The project combines **waterfall** and **agile methodologies** to organize development activities that will finally release iterative versions of the platform.

As defined in DoA, Section 3.1.1, the need to speed up the availability of results is especially critical such as to have early working elements usable in testing and evaluations. On the other hand, ELECTRIFIC partners are also aware of the challenges of collaborative and distributed projects, which indeed raise the bar of cooperation complexities in comparison with single organization settings. ELECTRIFIC will apply Agile Development methods although that Agile Development for large distributed collaborative projects is indeed a challenge. The Consortium will rely on an adapted version of Agile, in which development methods and techniques are combined with the structure necessary for a large and distributed collaboration effort.

Moreover, the ELECTRIFIC methodology also incorporates the overarching principle of having final users (real-life trials scenarios) as active, participating actors during all the project duration. This is reflected in a project planning structured in 6 different development-implementation-testing-evaluation iterations. Given the size and the distributed nature of the project, the duration of each iteration cycle has been fixed at six months average (within each cycle the Agile methodology will be applied with short term –few weeks based –Sprints).

V.2. Code organisation

The source control management system used in this project is Mercurial SCM[4]. The ELECTRIFIC source code is structured in following parts:

Table 5. Code organisation.

Repository	Description
/electrific/api	Holds CIM API related code (except Energis software)
/electrific/adas-service	Holds code for the ADAS Agent
/electrific/csp-service	Holds code for the CSP Agent
/electrific/efo-service	Holds code for the EFO Agent
/electrific/shared	Contains code shared across domains
/electrific/test-ui	Contains the source code for the debugging Web Application

The components build cycle is managed by the Gradle[5] build tool.

Individual teams can organize themselves using their own set of SCM tooling. However, the ELECTRIFIC project standardizes around Mercurial SCM. As a result, each team that uses a different SCM tooling will need to merge their source code in the designated Mercurial repository. Because CI tooling is setup to work with the Mercurial system, these merges need to occur on a frequent basis.

[4] https://www.mercurial-scm.org/

[5] https://gradle.org/

If changes are made to a Mercurial repository that needs to be backported to the team-managed SCM repository somehow, this will be the responsability of the team that manages the non-Mercurial repository.

V.3. Microservice architectural style

ELECTRIFIC adopts the microservice architectural style. With this approach a single application is developed as a suite of independent services, each running in its own process and communicating with lightweight mechanisms. For ELECTRIFIC the lightweight communication mechanism is the JSON-over-HTTP resource API. Every microservice is independent from others in terms of involved software technologies. Each service owns and manages its own data. However, the Java development framework Dropwizard[6] is the default choice in ELECTRIFIC because it provides all the functions needed to build a Microservices-based application:

- Embedded Jetty: every application is packaged as a jar instead of war file and starts its own embedded jetty container. There is no WAR file and no external servlet container.
- JAX-RS: Jersey (the reference implementation for JAX-RS) is used to write RESTful web services.
- JSON: The REST services speak JSON. Jackson library is used to do all the JSON processing.
- Logging: It is done using Logback and SLF4J.
- Hibernate Validator: Dropwizard uses Hibernate Validator API for declarative validation.
- Metrics: Dropwizard has support for monitoring using the metrics library. It provides unparalleled insight into what our code does in production.

The Jersey framework comes with a Server-Sent Events (SSE) Support. This can be useful to implement event-driven communication with events sent out from the ELECTRIFIC Backend.

For the design of the RESTful API, we use Swagger. Via the Cloud-based tooling ”SwaggerHub” we edit and build our API’s.

[6] http://www.dropwizard.io/

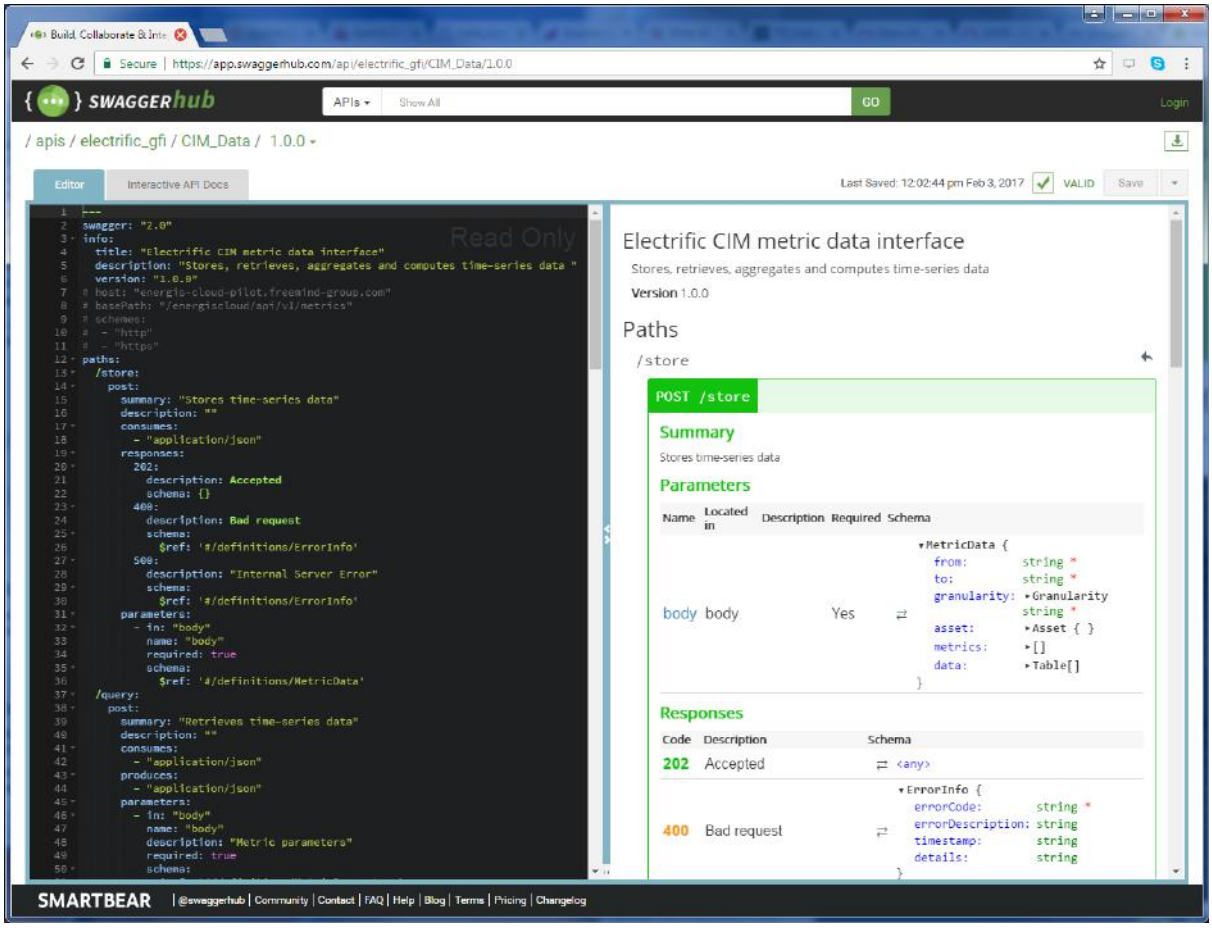

Figure 17. RESTful API design.

Using Swagger Editor we can design, describe, and document our API's. The Editor is great for quickly getting started with the Swagger specification. It's clean, efficient, and armed with a number of features to help you design and document your RESTful interfaces, straight out of the box. The source is a YAML document.

The solution allows for document and code generation.

V.4. Versioning

ELECTRIFIC applications follow the recommendations from the Semantic Versioning 2.0.0[7], a set of rules and requirements to manage software version in a suitable manner.

Versions must be defined exactly with three numbers: major.minor.maintenance (e.g. 1.4.2: 1=major, 4=minor and 2=maintenance).

Once a versioned package has been released, the contents of that version must not be modified. The source files constituting the version must be tagged with the name of the version. For each source control management repository, there will be a tag with the same name. Any new modifications must be released as a new version.

Each application will be versioned independently from the others and the version of the ELECTRIFIC solution will be defined adding a unified Electrific-version tag on every application repository.

[7] http://semver.org/

V.5. Artifact repository

A central artifact repository (Nexus) will be put in place and will act as a proxy towards the Internet. This repository will contain all released versions of the ELECTRIFIC modules and/or any 3rd-party library. A convenient authorization schema will be enforced to protect private artifacts.

V.6. Database versioning

The database schema will be maintained and versioned by Liquibase. This tool applies incremental changes to the schema and the database and keeps track of them. It allows for schema consistency and rollback.

Schema maintenance is organized using change logs. A change log is a sequence of change sets and each change set MUST contains one change to the database (e.g.: create table, add column, ...).

V.7. Static analysis

The use of static code analysis tools is a freedom of choice but is highly recommended. Each sub-project is free to integrate static analysis tools such as: Findbugs, checkstyle and PMD/CPD. If used, the proper plugins must also be enabled in the build scripts in order to follow-up on code quality in the CI environment.

V.8. Coding Style guidelines

Per sub-project coding style guidelines need to be agreed upon and applied by each developer. Such rules should be enforced by the IDE (Eclipse, IntelliJ) as much as possible (e.g. by using “save actions” in Eclipse).

The main driver here is to make sure that everybody on the team uses the same coding style to ease the process of merging code branches.

V.9. Automated tests using Jenkins

Continuous integration is performed using Jenkins[8].

The integration with Mercurial SCM is provided by Mercurial plugin of Jenkins[9]

V.10. Testing

During development unit tests will be developed to gradually build a suite of automated tests. These kind of tests allow for regression detection at the code level very quickly. Typically these these tests are: small, fast and numerous. During the development cycle these test are executed (very) often by the developers. Where possible the test-first principle should be applied. Related techniques and frameworks for mocking, calculating code coverage etcetera are also applied (e.g. Mockito, JaCoCo, ...) as a best practice in the testing methodology.

For the Android App, a suite of functional tests will de developed in the Espresso framework. These tests should cover at least the most important use cases.

For simulating traffic, load testing tool such as JMeter or Grinder are used. A set of tests will be developed to measure performance and validate main SLA’s. The outcome (metrics) of these tests can be used to compare previous simulations.

[8] https://jenkins.io/

[9] https://wiki.jenkins-ci.org/display/JENKINS/Mercurial+Plugin

Because most Service components are based on DropWizard, all RESTful API's will be unit tested (and integration tested) using the built-in testing capabilities of the framework.

Unit tests that have no runtime dependencies are executed during CI processes.

V.11. Continuous Integration and Delivery (CI/CD)

The ELECTRIFIC Solution will a use a continuous integration methodology, to integrate the multiple components developed by the different consortium partners. Developers and project management will receive reports on the success/failure of each of the integration phases.

The chosen tool, Jenkins will be used for:

- **Component builds and testing:** Build component by specifying their code repository branch.Run unit tests and deploy artifacts to a repository.
- **Integration Testing:** Run the Integration Test cases by specifying the project consortium components artefact version or code repository branch, as well as these components interaction with external APIs.
- **Integration Environment deployment:** Start containers and install the system components by specifying their artifact version or code repository branch.
- **System Testing:** Run the System tests cases.
- **Testing Environment deployment:** Start containers and install the system components by specifying their artifact version or code repository branch.
- **Tools Required:** Jenkins, Mercurial Code Repository, Nexus Artifact Repository, container deployment tool (e.g. Docker).

V.12. Branching strategy

The branching strategy will be based on Vincent Driessen git branching model[10].

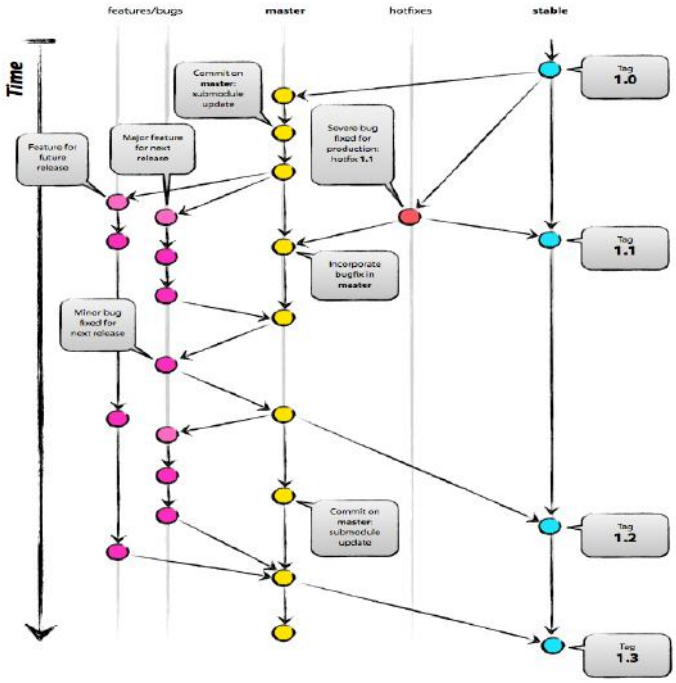

Figure 18. Branching strategy.

The master branch (head) of each software module typically has a CI job in Jenkins. For other branches like feature branches or hotfix branches, similar CI jobs might also be set up.

Continuous Delivery (CD) jobs can exist for the branches used to deliver to the test environment and can be scheduled on a daily basis (or triggered manually). For delivery jobs to the production environment we will most likely trigger them manually.

[10] http://nvie.com/posts/a-successful-git-branching-model/


electrific