*Smart system of renewable energy storage based on **IN**tegrated E**V**s and b**A**tteries to empower mobile, **D**istributed and centralised **E**nergy storage in the distribution grid*

Deliverable nº:	**D4.3**
Deliverable name:	**Overall INVADE architecture final**
Version:	**1.0**
Release date:	**18/12/2018**
Dissemination level:	**Public**
Status:	**Submitted**
Author:	Pau Lloret, Pol Olivella – UPC

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement No 731148.

Document history:

Version	Date of issue	Content and changes	Edited by
0.1	29/11/2018	Preliminary draft version	UPC
0.2	10/12/2018	Review of pilot owners	Lyse, Estabanell, Elaad, Greenflux, Albena, badenova
0.3	12/12/2018	Draft version, with updated sequence diagrams	UPC
1.0	18/12/2018	Peer-reviewed and submitted version	UPC, Terje Lundby, Stig Ødegaard Ottesen and Malte Thoma

Peer reviewed by:

Partner	Reviewer
eSmart	Terje Lundby and Stig Ødegaard Ottesen
badenova	Malte Thoma

Deliverable beneficiaries:

WP / Task
WP5
WP7
WP8
WP10

Table of contents

Abbreviations and Acronyms

Acronym	Description
AHES	AMI Head End system
API	Application programming interface
BMS	Battery Management System
BRP	Balance Responsible Party
BS	Balance Scheduling
CES	Centralised Energy Storage
CMO	Capacity Management Operator
CMS	Capacity Management System
CSMS	Charging Station Management System
CSO	Charging Station Operator
DER	Distributed Energy Resources
DES	Distributed Energy Storage
DMS	Distribution management system
DSO	Distribution System Operator
EMS	Energy Management System
EMSP	e-Mobility Service Provider
EV	Electric Vehicle
EVSE	Electric Vehicle Supply Equipment
FD	Flexible Device
FEP	Front End Processor
FMS	Flexibility Management System
FO	Flexibility Operator
FR	Flexibility Request
IBEX	Independent Bulgarian Energy eXchange
IEC	International Electrotechnical Commission
IED	Intelligent Electronic Device
IIP	Integrated INVADE Platform
MDC	Meter Data Concentrator
MDM	Meter data management
OCMP	Open Capacity Management Protocol

Acronym	Description
OCPP	Open Charge Point Protocol
OFPF	Optimal Flexibility Power Flow
OM	Operation meter
OSCP	Open Smart Charging Protocol
PRIME	PoweRline Intelligent Metering Evolution
PV	Photovoltaic
RTU	Remote Terminal Unit
SCADA	Supervisory control and data acquisition
SDC	Smart device controller
SGAM	Smart Grid Architecture Model
SM	Smart Meter
SOC	State of Charge
ToU	Time-of-Use
TSO	Transmission System Operator
V2G	Vehicle to Grid
WP	Work Package

Executive summary

This document is part of the WP4 (Overall INVADE architecture), which main objectives are to design the general INVADE system architecture and the software architecture of the Flexibility Cloud as a base for its implementation. In addition, the interface, communication and physical connections of the users to the Flexibility Cloud platform will also be specified.

In particular, this deliverable describes the outcomes of task T4.6, which reviewed the architecture of pilots and further developed the INVADE concept design to include the aggregated services in the second implementation phase.

The second section summarizes all changes and updates that have been identified in all pilots with respect to the second version of D4.2. The updated identification of the flexibility services that are going to be offered by the FO in each pilot are reflected in Table 1.

Table 1: Flexibility services to be used in each pilot (Y: yes; N: no).

Flexibility customer	Flexibility services INVADE	Norwegian pilot	Dutch pilots	Bulgarian pilot	German pilot	Spanish pilot
DSO	Congestion management	N	Y	N	Y	Y
	Voltage / Reactive power control	N	Y	N	Y	Y
	Controlled islanding	N	N	N	N	Y
BRP	Day–ahead portfolio optimization	N	N	N	N	N
	Intraday portfolio optimization	N	N	N	N	N
	Self-balancing portfolio optimization	N	N	Y	N	Y
Prosumer	ToU optimization	Y	Y	Y	N	N
	kWmax control	Y	Y	Y	N	N
	Self-balancing	Y	Y	Y	Y[1]	N
	Controlled islanding	N	N	N	N	N

Next section describes the aggregated flexibility services. These services are based on the flexibility request that the DSO or BRP have to send to the FO to specify the requested difference between the aggregation of individually optimized baseline and the desired load profile. In addition, the flexibility request issues are addressed to solve the

[1] Prosumer self-balancing service is going to be supplied by SMA outside the scope of the INVADE project.

conflicts that can arise between the individual prosumer optimization and the requests from the DSO/BRP or between the flexibility requests of the DSO and BRP.

Section 4 defines the sequence diagrams, focused on the interactions and messages that the FO needs to send for offering flexibility services to DSOs, BRPs and prosumers. It first describes the generic case that covers all the situations and possible interactions without constraining them to the available services and infrastructures in pilots, and then, it is adapted to the reality of each pilot.

Finally, this document describes the extensions of existing standards and protocols reviewed in T4.4. The proposed extensions include the new OCPP v2.0 protocol for integrated smart charging and V2G and the new OCMP 1.0 for information between EMSP and the DSO (OSCP 2.0 pre-version).

The outcomes of this deliverable are an input for WP7 as an input to specify the API functions, for WP8 to define the functionalities and architecture of the INVADE platform, and for WP10 to progress with the pilot implementation.

1 Introduction

In D4.1 [1], the general INVADE architecture is described. Using the description of the pilots in task T10.1, D4.2 [2] adapted the INVADE architecture to the different pilot sites. Then, this document D4.3 includes the review of the INVADE concept and architecture according to the feedback from pilots examined in T4.6.

The following sections describe the updates of the concept design and architecture of the pilots focused on the second implementation phase, which covers the aggregated services for the DSOs and BRPs.

2 Pilot's architecture update

This section summarizes all changes and updates that have been identified in all pilots with respect to the last version of D4.2 [2]. In D4.2 there were several uncertainties with the aggregated services for the DSO and/or BRP, which has been clarified and reflected in Table 2.

Table 2: Flexibility services to be used in each pilot (Y: yes; N: no).

Flexibility customer	Flexibility services INVADE	Norwegian pilot	Dutch pilots	Bulgarian pilot	German pilot	Spanish pilot
DSO	Congestion management	N	Y	N	Y	Y
	Voltage / Reactive power control	N	Y	N	Y	Y
	Controlled islanding	N	N	N	N	Y
BRP	Day–ahead portfolio optimization	N	N	N	N	N
	Intraday portfolio optimization	N	N	N	N	N
	Self-balancing portfolio optimization	N	N	Y	N	Y
Prosumer	ToU optimization	Y	Y	Y	N	N
	kWmax control	Y	Y	Y	N	N
	Self-balancing	Y	Y	Y	Y[2]	N
	Controlled islanding	N	N	N	N	N

If there are variations in the architecture of the pilot, the following sections describe the changes applied in each of them.

2.1 Norwegian pilot

No changes have been identified in the Norwegian pilot in relation to D4.2 (version 2.0). This pilot is focused on the standalone prosumer optimization without aggregated services.

2.2 Dutch pilot

The BRP services are not going to be implemented in the Dutch pilot. This pilot is then focused on the interaction of internal prosumer optimization based on EVs and DSO services.

[2] Prosumer self-balancing service is going to be supplied by SMA outside the scope of the INVADE project.

In addition, some updates have been done in the communication SGAM layer regarding the protocols used between the DSO and the FO, and between PV panels, batteries and their controlling systems. In the first case, the protocol used is OCMP, while in the second case, the protocols used are proprietary, as there are no broadly accepted industry standards for these devices yet. Updated communication layers can be seen at section 8.1 of the Annex.

2.3 Bulgarian pilot

The Bulgarian energy exchange is a young market platform that is entirely based on Nordpool. Recently, market regulation has changed and now the energy for the whole industry needs to be traded on the spot markets. As a result, the tariff used by Albena was changed by the energy supplier. Since the first of December 2018, Albena will be billed for electrical energy based on the day-ahead prices on the IBEX. This fact presents an excellent opportunity to show the benefits of the ToU optimization service offered by the FO.

In addition, the BRP can request external flexibility requests to improve its self-balancing portfolio to avoid imbalances. Day-ahead and intraday portfolio optimization are not going to be implemented in this pilot. Taking this into account, the Bulgarian pilot deals with the interaction of internal prosumer optimization and BRP services.

2.4 Spanish pilot

Prosumer services has been removed from this pilot due to the Spanish regulation, that does not allow the installation of batteries at household level and discouraged the installation of PV panels for installations of more than 10 kW of contracted power. In addition, no controlled EV nor PV panels are included in the pilot site. Updated component and communication layers can be seen at section 8.2 of the Annex.

The focus of this pilot is the interaction of DSO and BRP flexibility requests, especially when they are contradictory. Within the BRP services, only self-balancing portfolio optimizations will be applied. The lack of specific bidding products for batteries in the day-ahead Spanish market, like “linked block orders”, makes day-ahead bidding strategies very complicated, risky and therefore less profitable in this pilot. Due to the limited benefits it can bring, the day-ahead portfolio optimization is not going to be implemented in this pilot. Intraday market optimization is a scenario of great interest due to the possibility of exploiting the intraday market options in parallel with flexibility

provided by the battery. However, this scenario is not going to be considered due to the lack of participation of Mercator in this market and the inexistence of available tools to forecast it.

2.5 German pilot

No changes have been identified in the German pilot in relation to D4.2 (version 2.0). This pilot is focused on the interaction of external prosumer optimized baseline and DSO services.

Significant progress has been made in the field of communication issues. Updated communication layers can be seen at section 8.3 of the Annex. Regarding the centralised energy storage (CES), the battery manufacturer uses a standardised Profibus communication protocol. A protocol converter will be programmed by the manufacturer to enable communication between the CES and the INVADE-API.

For distributed energy storages (DES), the INVADE platform receives actual values as well as forecasts for the residual load of each connected household as well as separately for the most important devices in each household (e.g. PV-system, EV-charger, battery) via the SMA platform. A large variety of data is available in high resolution and can be aggregated to user defined average values, if necessary. Further the nominal values of all relevant devices in the household are deposited. The data can be read out via the REST-API of the SMA platform. The use of smart meters is not necessary, as all needed values can be provided by the SMA platform. This helps to fit in the German pilot in the design of INVADE, as due to regulatory issues, smart meters are still not available in the German market. A protocol converter between the REST-API of the SMA platform on one hand and the INVADE-API on the other hand has to be programmed, as both platforms use proprietary communication protocols.

3 DSO and BRP flexibility services

Under the context of smart grids and flexibility services in place, balance responsible parties (BRPs) and distribution system operators (DSOs) could benefit by activating flexibility in distribution grids. Electricity prices are increasing in Europe due to a combination of factors that can differ from country to country. For example, the NordPool experienced twice costly electricity prices during summer 2018 due to no precipitation in combination with high CO2 and coal prices. Also, intraday and balancing markets could experience price volatility during scarcity periods of flexible generation. In that scenario, electric vehicles (EVs) may become price-reactive and, as a consequence, distribution grids could experience overloads during cheap periods with all EVs charging at the same time.

The solution offered by the INVADE project is a platform-based system capable of activating flexibility behind the meter during constrained periods. For instance, during high prices, BRPs could balance their portfolio by activating flexibility instead of paying penalties or high Intraday market costs. Likewise, DSOs could compensate grid congestions during high consumption or production periods.

Therefore, BRPs could use flexibility to reduce their electricity costs and DSOs could reduce their networks stress. At the same time, DSO can increase their renewable generation hosting capacity by using behind-the-meter flexibility during peak production periods.

3.1 Aggregated services

The main relevant assumptions made to be able to offer the BRP/DSO flexibility services are:

- The flexibility operator (FO), also known as an aggregator in other contexts, is in charge of scheduling all flexible assets according to different objective functions and flexibility contracts.
- End-users define their flexibility costs for all their Flexible Devices (FDs).
- The central platform sends the control signals to the flexible devices or to another broker platform which executes these signals. It can be aggregated control signals if there is local intelligence in the local EMS to distribute it between its FDs.

- The BRPs and DSOs define their flexibility needs, and they send flexibility requests (FR) to the FO.

Table 3 shows all the aggregated services that can be offered by the FO to the DSOs and BRPs. They are further described in the Deliverables D4.1 [1] and D5.1 [3] of the INVADE project.

Table 3: Aggregated flexibility services to be offered by the FO.

Flexibility customer	Flexibility services	Description (Flexibility usage)
BRP	Day-ahead portfolio optimization	Shifting loads from a high-price time interval to a low-price time interval before the day-ahead market closure. It enables the BRP to reduce its overall electricity purchase costs
	Intraday portfolio optimization	Enabling value creation on the intraday market, equivalent to the day-ahead market
	Self-balancing portfolio optimization	Reducing imbalance by the BRP within its portfolio to avoid imbalance charges. The BRP does not actively bid on the imbalance market using its load flexibility but uses it within its own portfolio.
DSO	Congestion management	Avoiding the thermal overload of system components by reducing peak loads where failure due to overloading may occur.
	Voltage / Reactive power control	Using load flexibility by increasing the load or decreasing generation is an option to avoid exceeding the voltage limits, typically, when PV systems generate significant amounts of electricity.

The flow diagram of the flexibility algorithm with aggregated services can be seen in Figure 1. The first step is to optimize for each prosumer all its flexibility devices. The result obtained is the 'optimized baseline'. This baseline is used to apply the flexibility request. The zone level optimization considers the external flexibility requests and create new control signals for the following period. This process is repeated for every period.

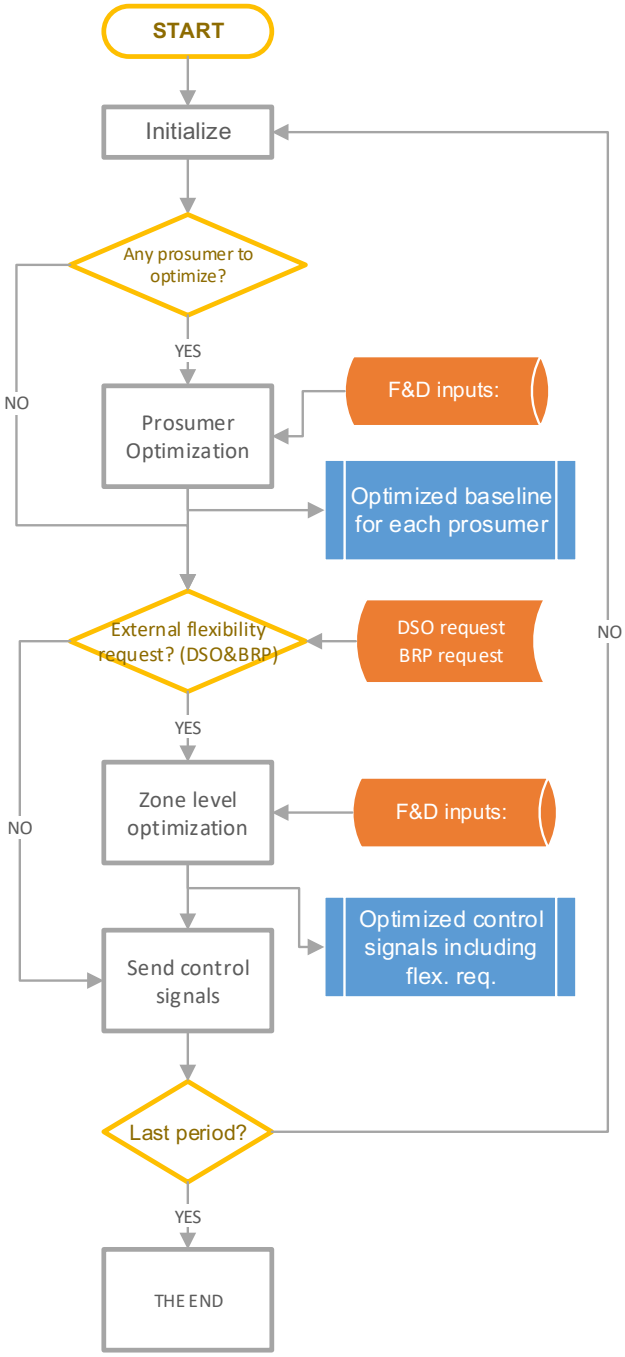


Figure 1: Flow diagram of the flexibility algorithm with aggregated services.

3.2 Flexibility request and reference baseline

The fundamental concept for creating flexibility exchanges between energy agents is the flexibility request which is very similar to the TSO's requests in tertiary reserve markets. It is defined as the difference between a baseline and the desired load profile. It can be for up or down regulation and can be measured in energy per programming time unit (PTU). Typically, an hour or a quarter.

Therefore, it is necessary to agree in a common baseline as a reference. The baseline in tertiary reserve markets is the previous settled markets. However, the application of FRs at distribution level is necessarily different as there are no previous market settlements at this level.

The agent in charge of defining the baseline can be different from case to case. For instance, it can be an independent entity in charge of energy data management or regulated energy agents. In case of no separated entity for it, the different agents involved need to agree on a common reference.

In cases with no flexibility requests, the FO schedules its flexibility devices (FDs) to minimize the electricity bill of each consumer individually. This is the FO optimized baseline and it is the reference of the FO in case of receiving a FR. The FO baseline would reflect the optimized prosumer cost without the aggregated flexibility services.

DSOs, BRPs and FO have different portfolios and they forecast their own portfolio load for the following hours. This means that they can have different expectations but they must agree on a common baseline. For simplicity, in case of a single aggregator, DSOs and BRPs accepts FO's baseline as the baseline reference.

Based on forecasting, DSOs and BRPs can detect flexibility needs. For example, DSO can potentially exceed a capacity limitation based on its own forecast. However, the aggregator does not necessarily control all end-users that the DSO is forecasting. Therefore, the DSO sends a flexibility request based on its own forecast, the FO applies this FR based on its own baseline and the DSO agrees on the FO's baseline. The same scheme can be applied for BRPs. If the BRP forecasts a deviation, it produces a FR based on its own baseline and the FO schedules its portfolio of FDs in order to meet the BRP FR.

In case of multiple aggregators competing for providing flexibility this could be adapted. However, it is out of the scope of the project.

3.2.1 Spare capacity

The difference between a baseline and the desired load profile is not the only necessary information that a flexibility request should include. When processing a flexibility request, the FO needs to know in which periods it is possible to modify the forecasted baseline in order to avoid creating worst problems than the solved by the flexibility request. These spare capacity are the periods used to allocate the energy needed to provide the

flexibility request due to charging/discharging necessities, rebound effects or shifting allocation.

In the example represented in Figure 2, the DSO sends a flexibility request to reduce the consumption in period 4, as the forecasted consumption surpass the limit of the grid (20 kWh). In addition, it also indicates the spare capacity for the rest of periods, which is the difference between the forecasted consumption and the limit of the grid.

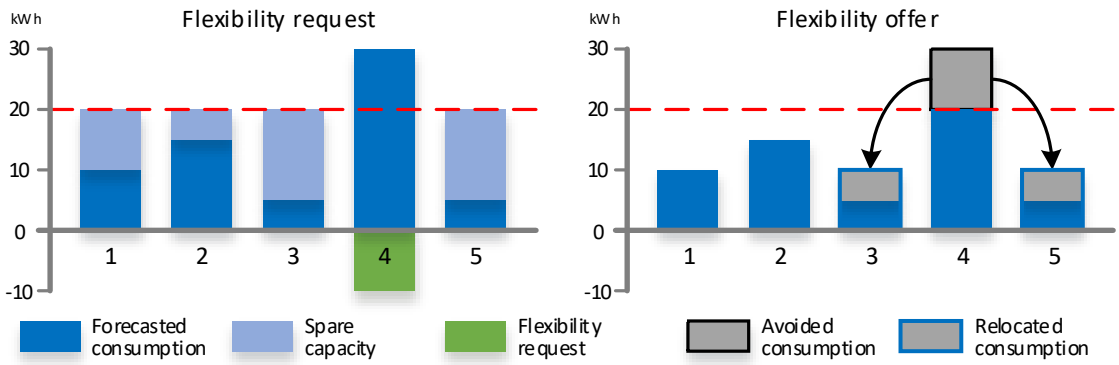


Figure 2: Example of usage of the spare capacity.

Then, when the flexibility request is accepted, the avoided consumption in period 4 can be relocated in other periods. The flexibility offer contains both the periods where the consumption is reduced to accomplish the flexibility request and the spare capacity used to relocate the avoided consumption, if needed. In this example, the request is fully accomplished in period 4 shifting the reduced consumption to periods 3 and 5.

3.3 Flexibility request prioritization

3.3.1 Prosumer vs BRP/DSO services

By definition, prosumer and BRP/DSO decisions will differ. FO schedules FDs to minimize prosumer's electricity bill individually. This situation is considered the baseline and BRPs and DSOs use them as a reference. If it is necessary, the FO can send this information to BRPs and DSOs. As a result, it is possible that the optimal solution for each prosumer individually causes congestion in the distribution grid because, for example, all EVs are planned to charge simultaneously.

Additionally, BRP/DSO flexibility needs are known only by themselves. The BRP portfolio balance or the grid status is not public information. Therefore, BRPs and DSOs have to send their flexibility requests to the FO, and they have to specify the quantity of flexibility needed during each period. In case of a single FO, BRP/DSO and FO can agree on the

flexibility cost in advance by contract. In the case of multiple aggregators, it can be more complicated, and it is out of the scope of this project.

In case that a flexible end-user has to change its energy consumption from the optimal baseline case, it should be compensated economically. Therefore, the FO can pay an economic compensation to the end-users that are deviating from their baseline energy profile. Moreover, the FO aims to find the cheapest flexibility offers that comply with the BRP/DSO request.

3.3.2 BRP vs DSO conflict resolution

In addition to the conflict between the prosumer and BRP/DSO necessities, BRP and DSO flexibility requests can differ and some kind of request priority is needed. The following scenarios aim to show the resolution of flexibility request conflicts under different situations.

In this case, the aggregator provides flexibility services to the DSO for congestion management and to the BRP for self-balancing portfolio optimization. The case study and all input parameters to the optimization problem are the same as in [4]. Figure 3 shows the flexibility requests applied to the same case study and they constitute three comparative scenarios:

- Scenario 1: The DSO and BRP request flexibility in different periods; the DSO requests down-regulation during midday; and BRP requests down- and up-regulation during the night.
- Scenario 2: The DSO and BRP request flexibility in the same periods and in the same direction. During midday and midnight, both need down- and up-regulation, respectively.
- Scenario 3: The DSO and BRP request flexibility in the same periods but in opposite directions. The DSO and BRP request down- and up-regulation, respectively.

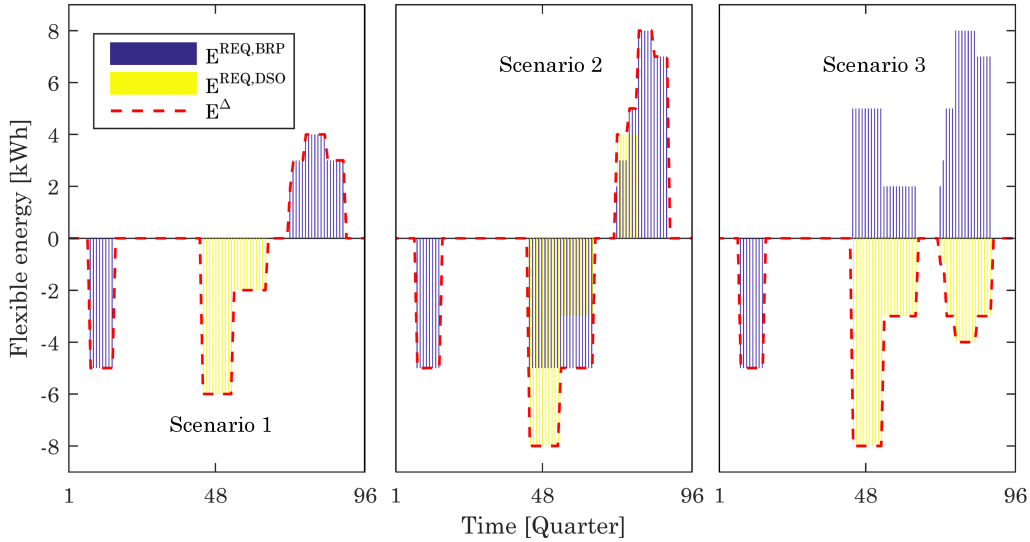


Figure 3: The DSO and BRP flexibility requests in the three scenarios [5].

Applying the optimization problem presented in [4] in the same case study to the previous flexibility requests, Figure 4 shows how the loads, batteries and generators contribute to fulfil the requirement. Disconnectable loads contribute to up-regulation during the evenings; generators can be disconnected during mid-day to provide down-regulation; and batteries and shiftable loads can contribute in both directions.

All scenarios could be profitable for the aggregator depending on the flexibility contracts and the DSO and BRP flexibility fees. The analysis of the aggregator profitability is out of the scope and could depend on the case study.

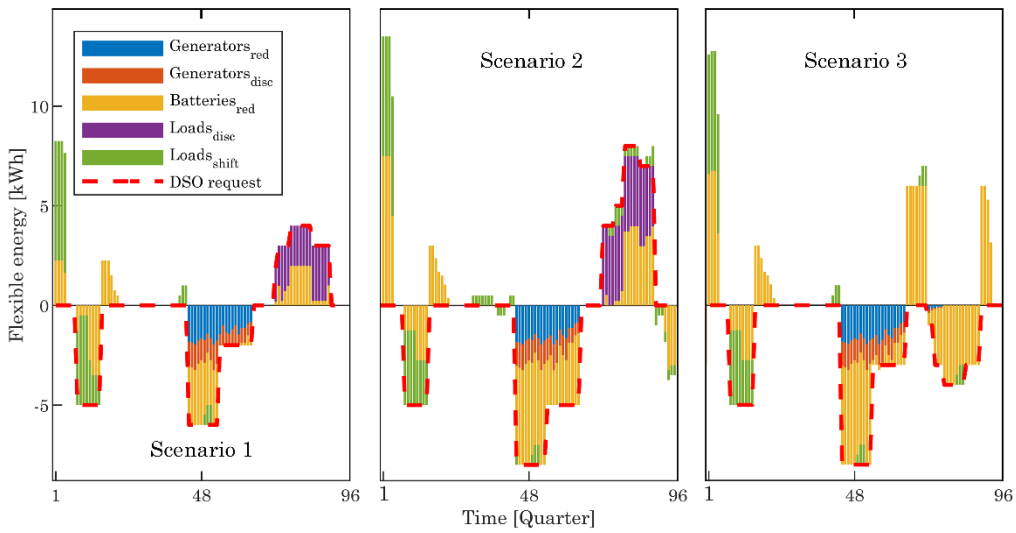


Figure 4: Test case results from the three scenarios [5].

3.4 Flexibility request examples

3.4.1 Flexibility request example with EVs

The following case shows an example with two houses offering flexibility services to external entities. In this case, there are two houses with one EV each one and they optimized the EV charging to be as cheap as possible. Therefore, the EVs charge simultaneously between periods 6 and 8 (Figure 5).

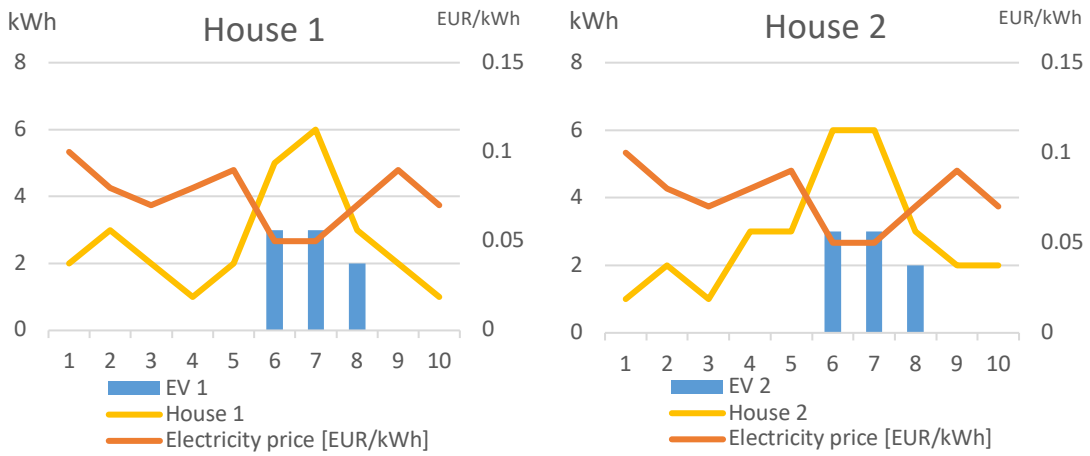


Figure 5: Example 1: baseline.

Figure 6 shows the aggregated baseline, which is a combination of these two houses. Due to this, the DSO needs to send a flexibility request to the FO to reduce its consumption during periods from 6 to 8.

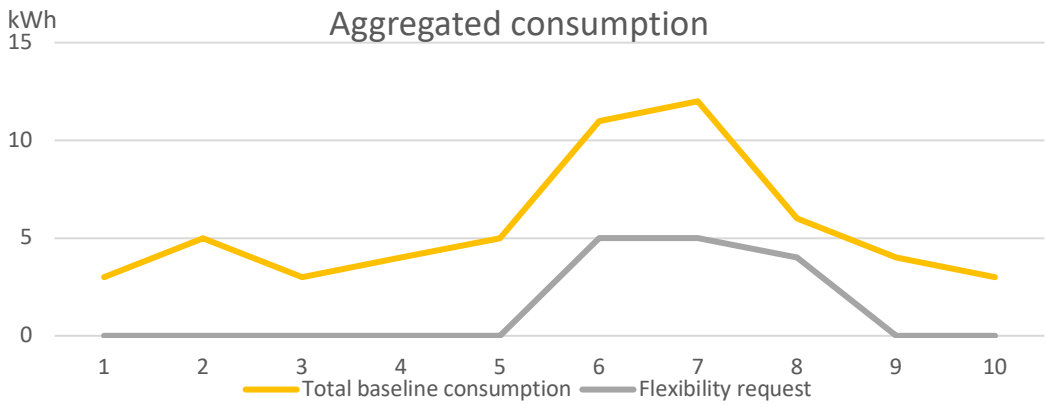


Figure 6: Example 1: aggregated baseline and flexibility request.

Therefore, the FO reschedules the EVs to be charged later (Figure 7). The EV charging station only supplies 1 kWh to charge the EV between periods 6 and 7, and 6 kWh between periods 9 and 10. The aggregated rescheduled consumption can be seen in Figure 8.

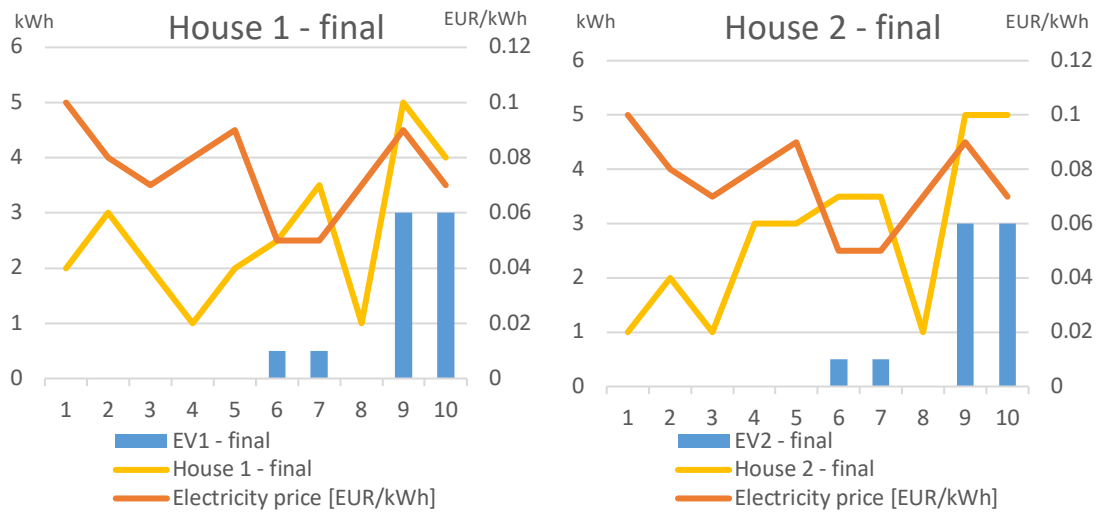


Figure 7: Example 1: individual schedule accepting the flexibility request.

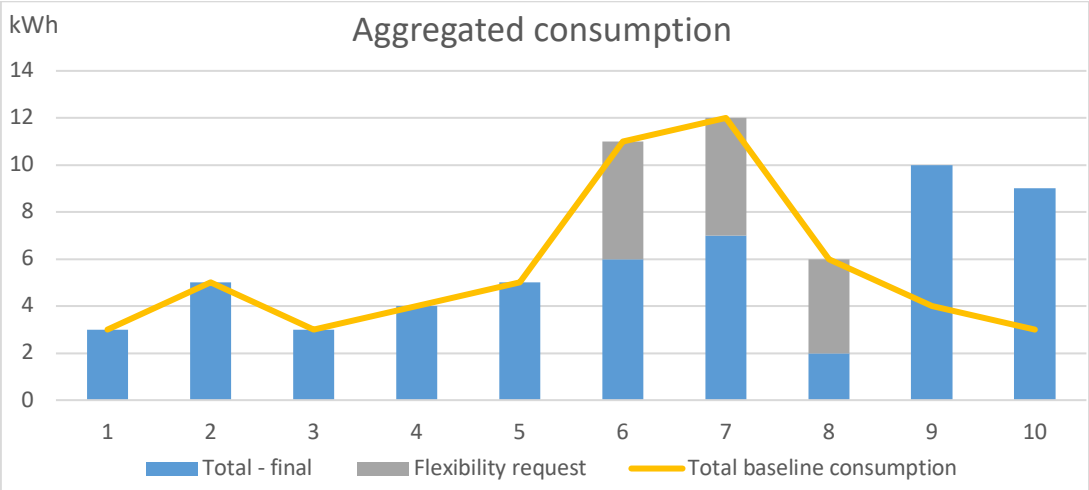


Figure 8: Example 1: aggregated schedule accepting the flexibility request.

It is possible to see how the aggregated consumption is reduced in periods 6 to 8 according to the external flexibility request. Therefore, the flexibility request and the final aggregated consumption sum the same than the previous aggregated consumption.

The table of electricity costs for each house is attached below. Each house increases 9 c€ their electricity bill and the FO has to pay at least this amount to each house for providing help to the DSO. Additionally, the FO could include additional rewards on top of that.

Table 4: Aggregated.

	House 1	House 2
Without request	1.85 €	1.97 €
With request	1.94 €	2.06 €

Difference	0.09 €	0.09 €

3.4.2 Flexibility request example with batteries

This case is the same than the first example, but the houses have batteries instead of EVs. According to this baseline (Figure 9), both batteries try to charge during cheap hours and discharge at high cost hours. This is just an example and the results of the individual optimization baselines could be suboptimal.

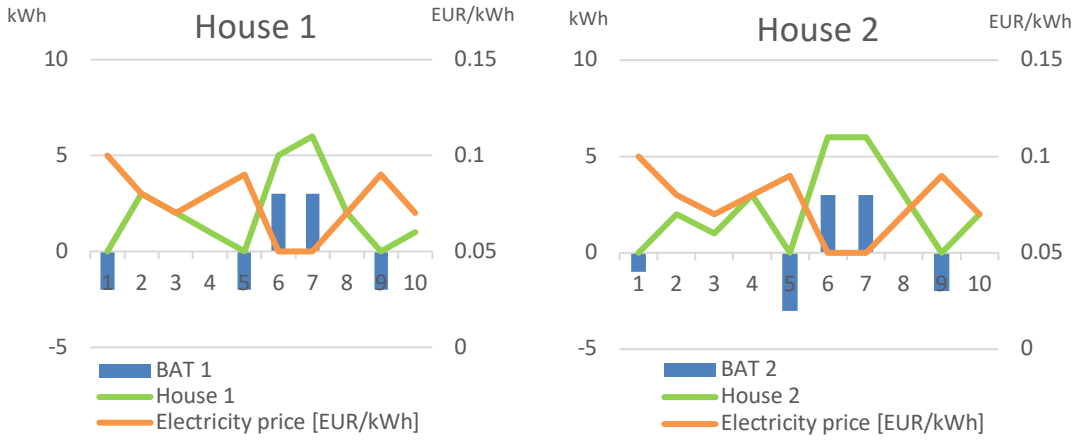


Figure 9: Example 2: baseline (BAT > 0 means battery charge and BAT <0 means battery discharge).

However, their consumption creates a peak in the grid and the DSO sends a flexibility request for up and down regulation to avoid this peak in the grid. The flexibility request involves batteries and therefore the DSO must indicate the spare capacity to show the FO when it can compensate end-users for the pending energy charges. In this example it is in periods from 8 to 10.

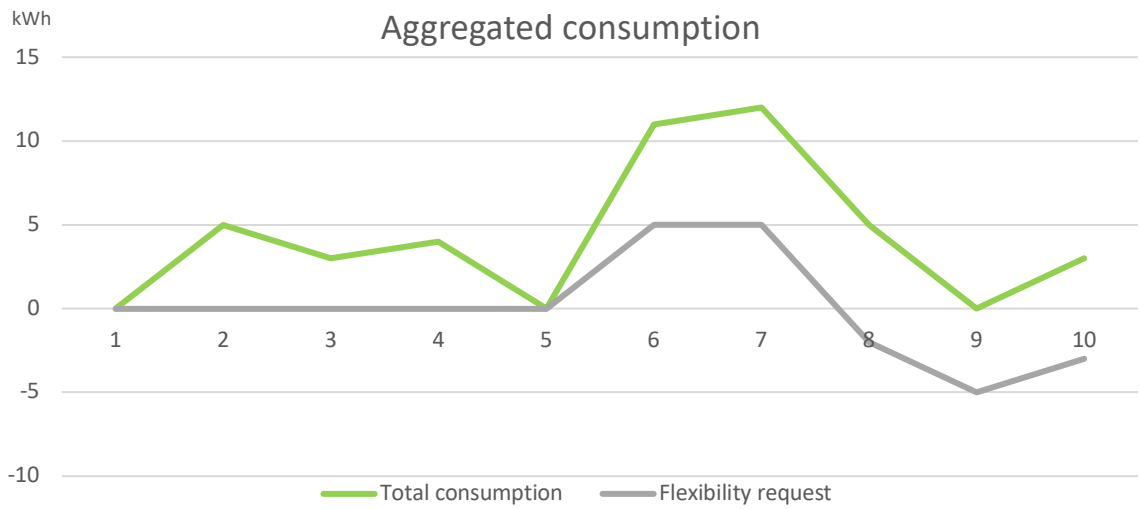


Figure 10: Example 2: aggregated baseline and flexibility request.

The results are shown in Figure 11. Notice that both batteries are charged smoothly and the baseline for the flexibility request activation is the FO optimized baseline.

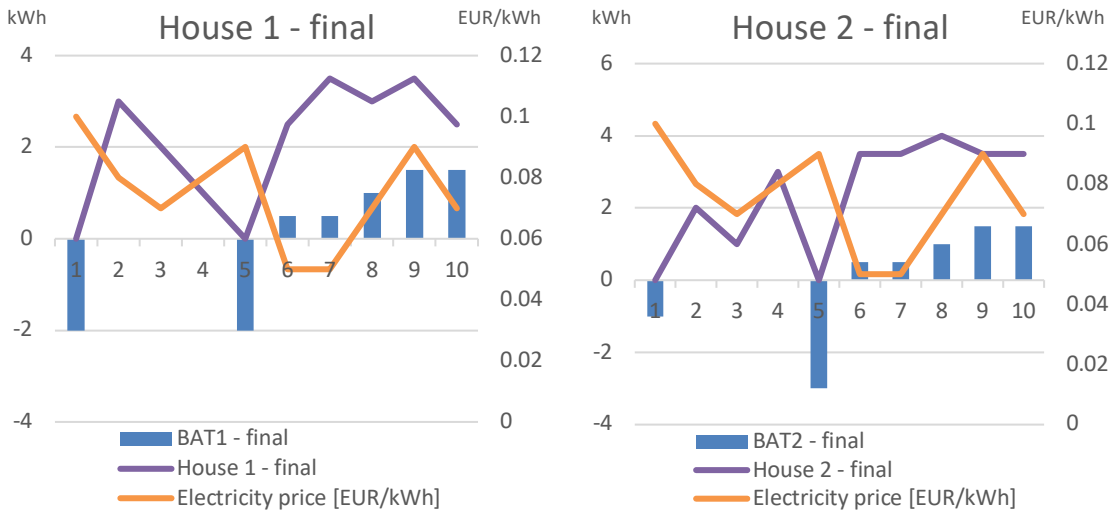


Figure 11: Example 2: individual schedule accepting the flexibility request.

4 Sequence diagrams

4.1 Generic sequence diagram

The following diagram (Figure 12) describes the generic interactions for offering flexibility services to DSOs, BRPs and prosumers using a flexibility device (battery, controllable load, EV charging, etc.) coordinated by a FO.

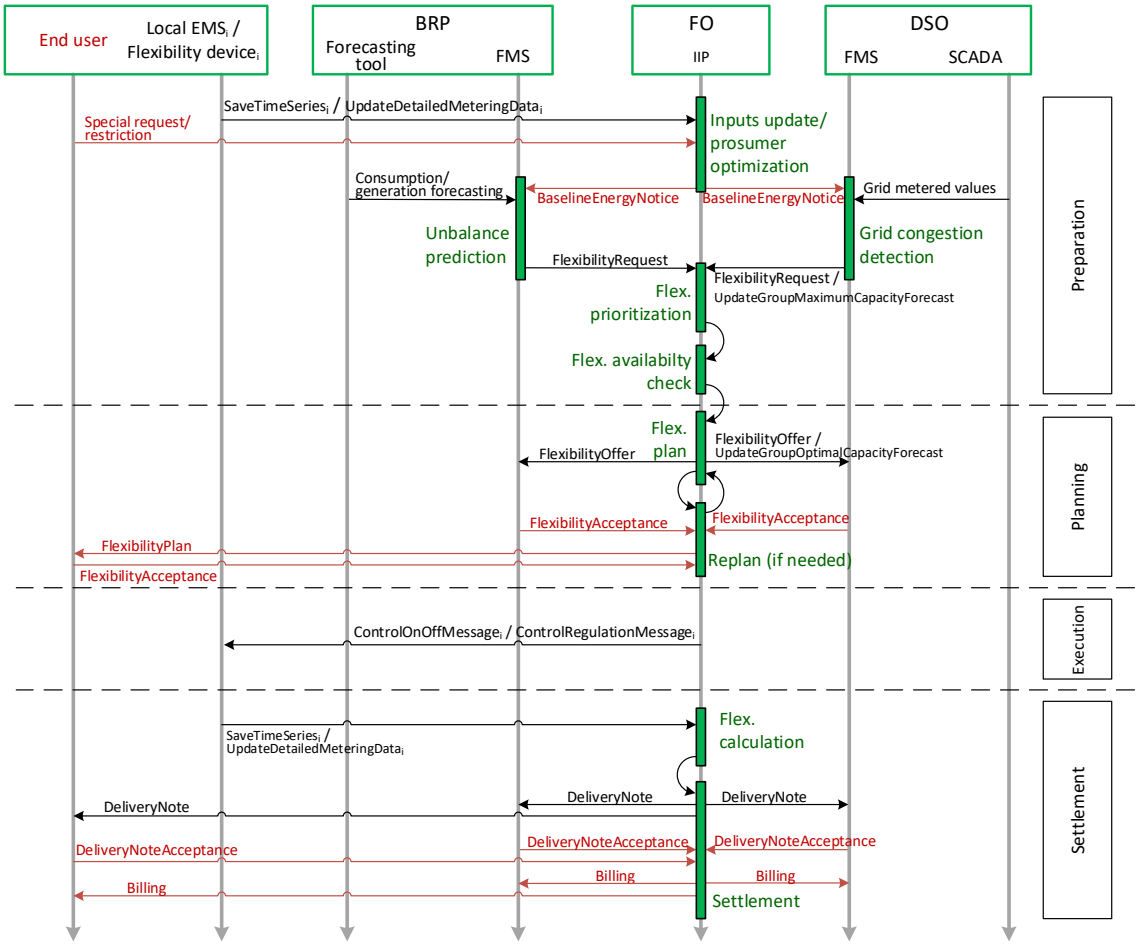


Figure 12: Generic sequence diagram.

This is a generic diagram and it needs to be adapted to each of the pilots, as additional intermediate actors can also be incorporated to this ecosystem (e.g. Smartly or Greenflux platforms). In red, there are optional interactions that are not compulsory to be implemented (they are mainly related to enable modifications into the flexibility plan and receive acceptance of the FO actions). The identification names used for each message are the ones defined in the FO API (see D7.2 for more details on these messages).

The whole procedure is divided into 4 consecutive phases: preparation, planning, execution and settlement.

Preparation phase: it is used to receive and prepare all the information needed before running the optimization to create the flexibility plan. The first needed step is to create the individual optimized baselines for each of the flexibility devices prior to any BRP or DSO flexibility request. This phase also includes creating the DSO and BRP flexibility request, prioritising these requests based on the grid state and updating all metered and status values. Optionally, the end user can be able to send a special request that will modify the flexibility plan (e.g. if you want to have the EV charged earlier than usual). The most important messages associated to this phase are:

- Baseline (BaselineEnergyNotice): It is optional. It describes the baseline scenario prior to the flexibility request based on disaggregated local optimizations. It should help the DSO/BRP to formulate its flexibility request and it will be used as a basis for flexibility acceptance, in terms of evaluation delivered flexibility in the settlement phase. It contains the aggregated net consumption and production as a time series in kWh for each BRP/DSO zone/area. It can also include the spare capacity or the maximum available flexibility capacity.
- Flexibility requests (FlexibilityRequest): It includes, at least, the requested flexibility amount and schedule (kWh and when it is needed), and the area (area id). It covers a maximum of 48-hour ahead ending at intraday requests. It must have a time series of type:
    - Flexibility Up (reduced consumption or increased production compared to the baseline)
    - Flexibility down (increased consumption or decreased production compared to the baseline)
    - It may contain a SpareCapacity time series indicating the period with available spare capacity to provide the flexibility request (due to charging/discharging necessities, rebound effects or shifting allocation).

Planning phase: based on the previous inputs, the flexibility plan is generated. Optionally, DSOs, BRPs and end users can send an acceptance of the flexibility offer or plan. Then, if the DSO, BRP or end-user rejects it, the plan can be recalculated with the proposed changes. New requests completely supersede any previous messages as there is allowed only one active request for each sender at any point in time. If the DSO/BRP sends a new flexibility request after flexibility offers have already been accepted for the same period, the existing already accepted offer will remain in force until the new request has been through the communication loop and had a corresponding

offer back to the BRP/DSO accepted by it. Sending more than one request for the same period will happen only in emergency cases and it is not the common expected behaviour. The most important associated messages to this phase are:

- Flexibility offer (FlexibilityOffer): It indicates the quantity from the flexibility request that the FO is able to deliver. Similar to the flexibility request, it includes the schedule (kWh and when it is needed), and the area (area id) mentioned in the DSO request. It covers a maximum of 48-hour ahead ending at intraday requests. It must have a time series of type:
    - Flexibility Up (reduced consumption or increased production compared to the baseline)
    - Flexibility down (increased consumption or decreased production compared to the baseline)
    - It may contain an AdjustedCapacity time series, indicating where any loads/generation have been shifted to. It shows which spare capacity is used.
- Flexibility acceptance (FlexibilityAcceptance): It has same details as the flexibility offer. If no offer can be found matching the acceptance, it will be rejected.
- Flexibility plan (FlexibilityPlan): It is similar to the flexibility offer, but with a detailed schedule for each device inside a site instead of aggregated values per zone or area.

Execution phase: only the last accepted offers are going to be applied in the execution phase.

- Control signals to the flexibility device (ControlOnOffMessage or ControlRegulationMessage): It can be an ON/OFF command or a specific set point in kW to be followed by the flexible device to be controlled. It can be sent in real-time or in advance if there is some kind of local intelligence. By default, it is sent in real-time.

Settlement process: it can be quite different depending on the flexibility services offered, the flexibility devices and actors involved in each pilot. In addition, contract contents and role allocations can be somewhat different in each pilot. For these reasons, each pilot owner has to specify in detail its own settlement and billing processes. This process at least needs to include the following interactions:

- Receive metered values from the flexibility device.

- FO then uses the metered values to calculate the flexibility activated by the FO and to identify if the user by-passed the FO orders.
- With the calculated activated flexibility, the FO creates a delivery note for assessing the activated flexibility and for the services provided with this flexibility.
- Finally, invoices are sent to all involved actors, including payments, reimbursements or penalizations, if needed.

In addition to all the messages, metered values can be pushed to the Invade platform via the restful API method SaveTimeSeriesV1 documented in D7.2. Typically, the pilots will push all meter readings and related time series data to IIP via this method on a regular frequency such as every 15 minutes, the response from the API method will indicate if saving was successful. As an alternative, if the pilot had a large number of time series to send for device level metered data the data can be interfaced via the Event hub to Invade described in D7.2 Azure Event hub.

4.2 Sequence diagrams adapted to pilots

The sequence diagrams, including the red lines, show one way to implement the flexibility request. However, not all this functionality will be implemented and tested in the pilots. The decision on what will be implemented in pilots will depend on the discussion between WP10, WP4 and WP8.

The following sections describe the adaptation of the generic interaction diagram to each of the pilots. Although they try to follow the same principle, it results in different event scenarios due to the use cases that are going to be demonstrated, the context of the country where it will be set up, or the singularities of each pilot site.

4.2.1 Norwegian pilot

The Norwegian pilot is based on the use of several flexibility sources within a household to provide prosumer services that try to reduce the prosumer bill. The retailer company Lyse plays the FO role in this pilot site. Figure 13 shows its generic interaction diagram.

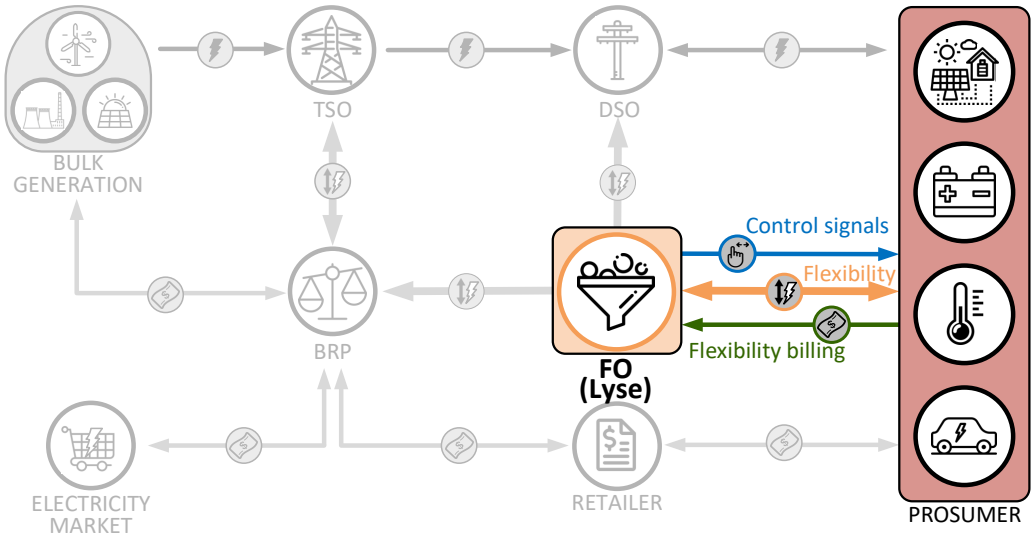


Figure 13: Generic interaction diagram of the Norwegian pilot.

In this case, the INVADE platform interacts with the household devices through the Smartly Hub, who is in charge of the interaction with the flexibility devices. Figure 14 shows the detailed sequence diagram of the Norwegian pilot.

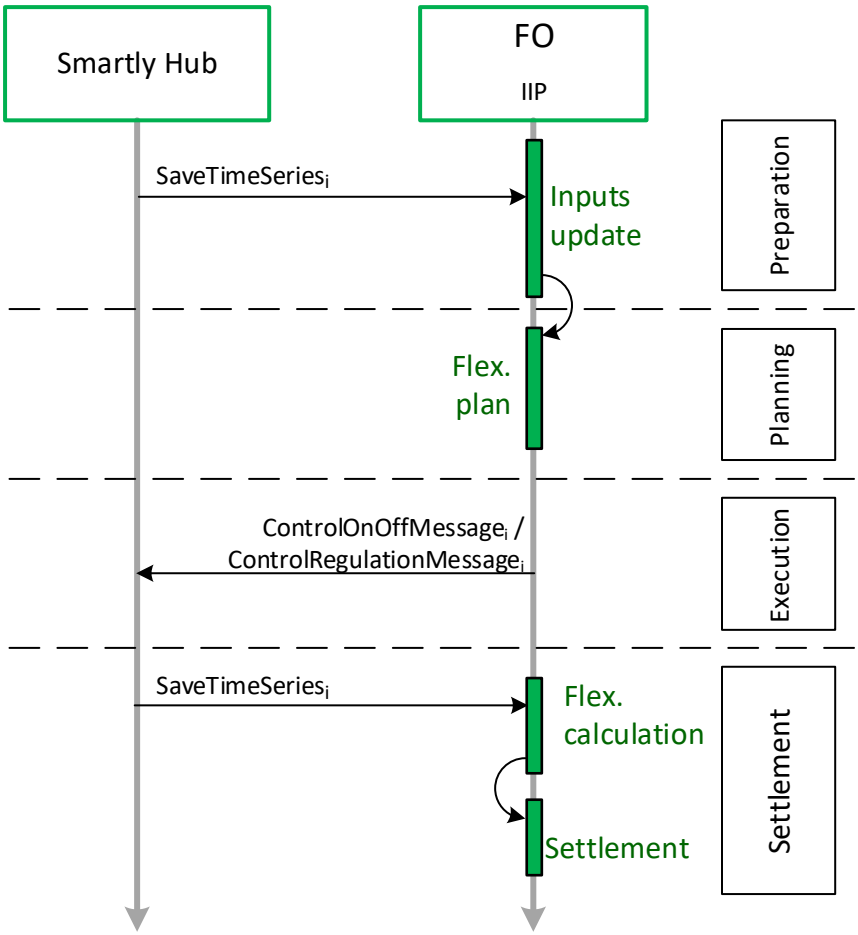


Figure 14: Detailed sequence diagram of the Norwegian pilot.

As there are no aggregated services, the sequence is more simple than in other cases. The sequence is as follows:

- Based on the metered values, the status of the flexibility sources and the forecast, the INVADE platform creates the flexibility plan with the optimized consumption profile of all the controllable flexibility sources inside the prosumer premises.
- The FO will send the signals to the Smartly Hub as soon as they are ready, which means that signals will be sent for the whole planning horizon every time without waiting until the execution time arrives. Then, the Smartly Hub is in charge of sending the control signals to each device.
- In the settlement process, the FO uses the metered values to calculate the flexibility activated by the FO in comparison to the forecasted baseline. The delivery note created by the FO is only used for internal purposes and it is not communicated to the final costumer.

4.2.2 Dutch pilot

The Dutch pilot is based on the use of flexibility devices, mainly EVs, to provide prosumer optimization with the inclusion of external DSO requests of available capacity. Greenflux, as providers of the Charging Station Management System (CSMS), also plays the FO role in this pilot site. Figure 15 shows its generic interaction diagram.

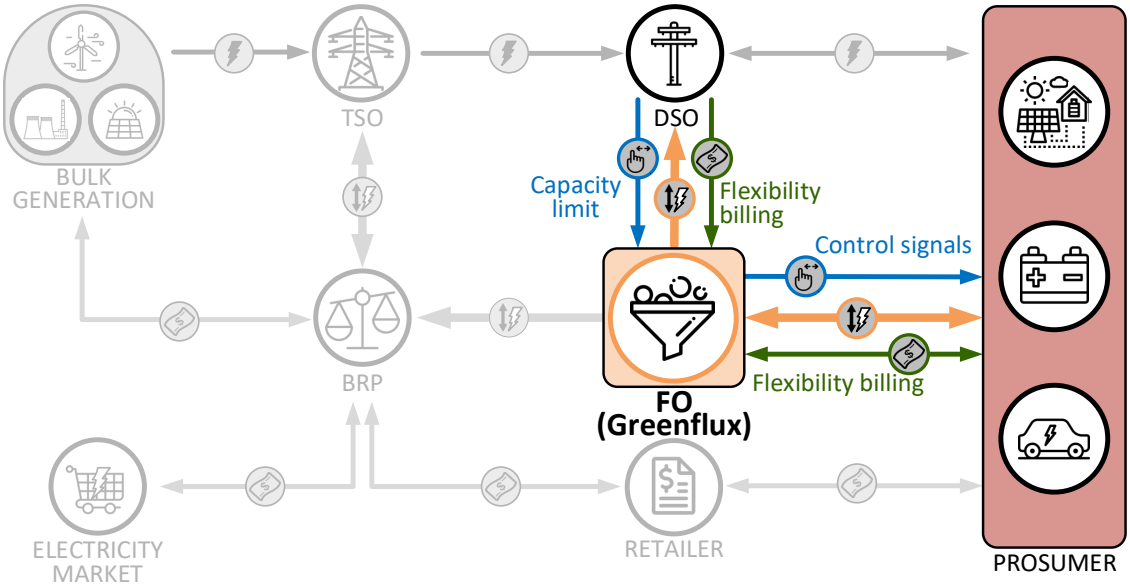


Figure 15: Generic interaction diagram of the Dutch pilot.

The sequence of the messages to make use of the flexibility requests from the DSO are represented in Figure 16. This sequence is based on the OCMP v1.0 standard, which is a pre-release of the next version of OSCP v2.0.

To prevent network congestion, the DSO can measure the load on the local electricity network and based on the maximum capacity of the network components, the DSO can send out the maximum available capacity for EV charging.

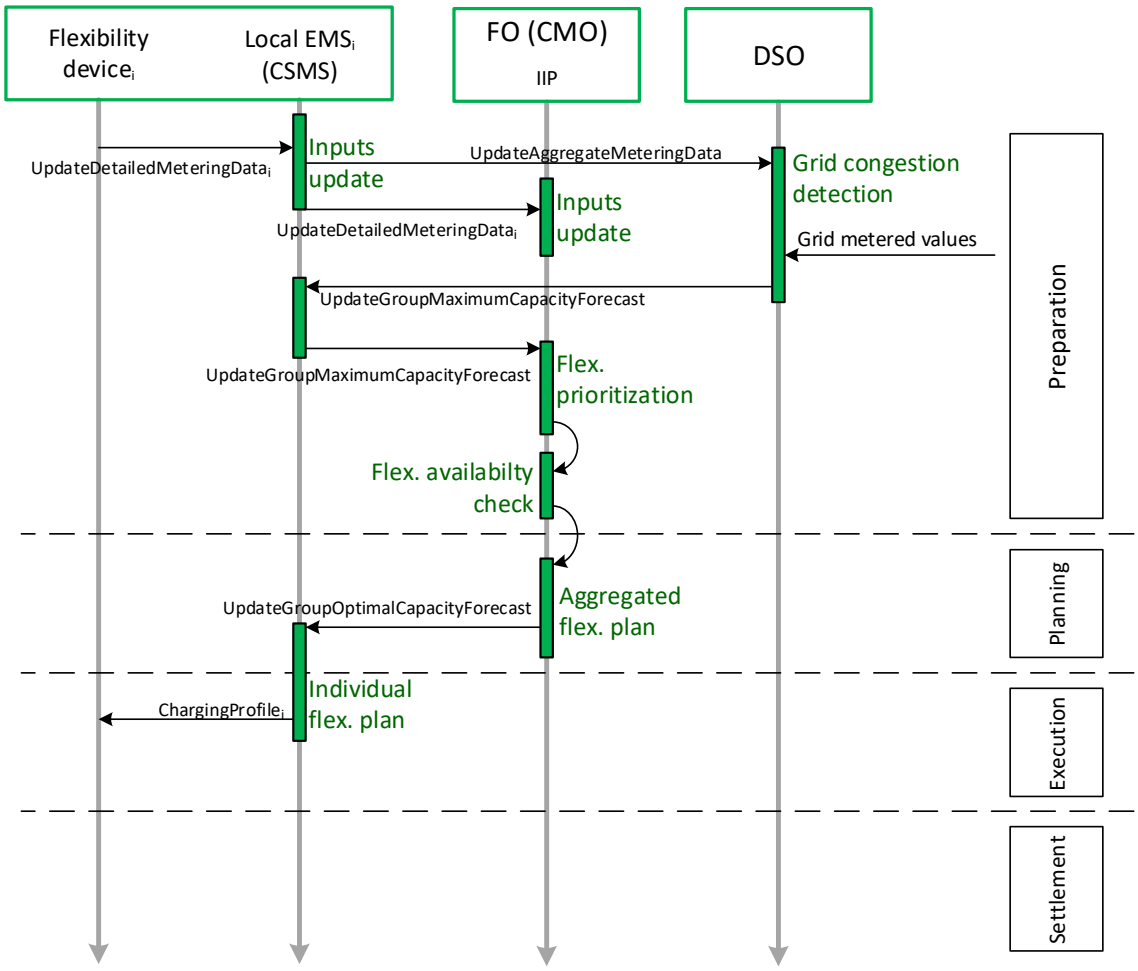


Figure 16: Detailed sequence diagram of the Dutch pilot.

The action sequence is as follows:

- The DSO receives measurements from its distribution grid (outside the scope of the project and the OCMP protocol).
- Based on these measurements, a forecast of the available group capacity is made, which is forwarded to the Charging Station Operator (CSO) using the UpdateGroupMaximumCapacityForecast message to inform the CSO about available capacity. The calculation of the available capacity by the DSO is outside the scope of the INVADE project. The CSO is the party that manages a Charging

Station Management System (CSMS) that manages charging stations and has the information for authorizing users for using its charging stations.

- The CSO redirects UpdateGroupMaximumCapacityForecast message to the FO (Capacity Management System (CMO) in OCMP).
- After FO has calculated the optimal capacity forecast, it is forwarded to the CSO using the UpdateGroupOptimalCapacityForecast message.
- Based on this forecast, the CSO can tell its charging stations what their maximum capacity for the next period of time is. Subsequently, the EVs at the charging stations can be charged, which might result in the EV not charging at full capacity.

In this case, there is no settlement phase related to flexibility. Invoicing is based on delivered energy [kWh] based on meter values.

4.2.2.1 Dutch DSO flexibility request calculation

In order to correctly determine the flexibility need, the DSO makes a prognosis of the expected energy net exchange with the main grid. This is usually done in periods of 15 minutes for the next day or days. As input, they can use field measurements from substations, historical data, weather data, events and holiday calendars, etc. Once an accurate as possible prediction is created, they can determine the likelihood of problems when they do not intervene, and the severity of those problems.

Within the INVADE project, we pilot a strategy where the DSO sends a daily maximum capacity to a CSO. This is done on a grouping level (e.g. cable, area) and not on device level. It is up to the CSO to translate the request into charging schedules for the charging stations in such a way that the maximum capacity limit is respected. Since the DSO algorithm in this case is very basic, the INVADE platform can add real value. The CSO wants to make best use of its charging infrastructure, but usually lacks the knowledge to optimize. So, our CSO forwards the maximum capacity request to the INVADE platform. In this platform a lot of extra factors such as weather data and the production of renewables, etc. can be taken into account. The CSO receives back from the INVADE platform an optimized capacity limit, still on grouping level and in turn, translates this forecast into optimized charging schedules for ongoing transactions. This approach is

based on a long-term agreement between the DSO and CSO for flexibility and not a per-delivered-flexibility payment system.

4.2.3 Bulgarian pilot

The Bulgarian pilot is based on the use of several flexibility devices to provide prosumer optimization with the inclusion of external BRP requests to adjust its imbalances. Albena is the owner of the hotel resorts and its flexibility sources and it also plays the FO role in this pilot site. Figure 17 shows its generic interaction diagram.

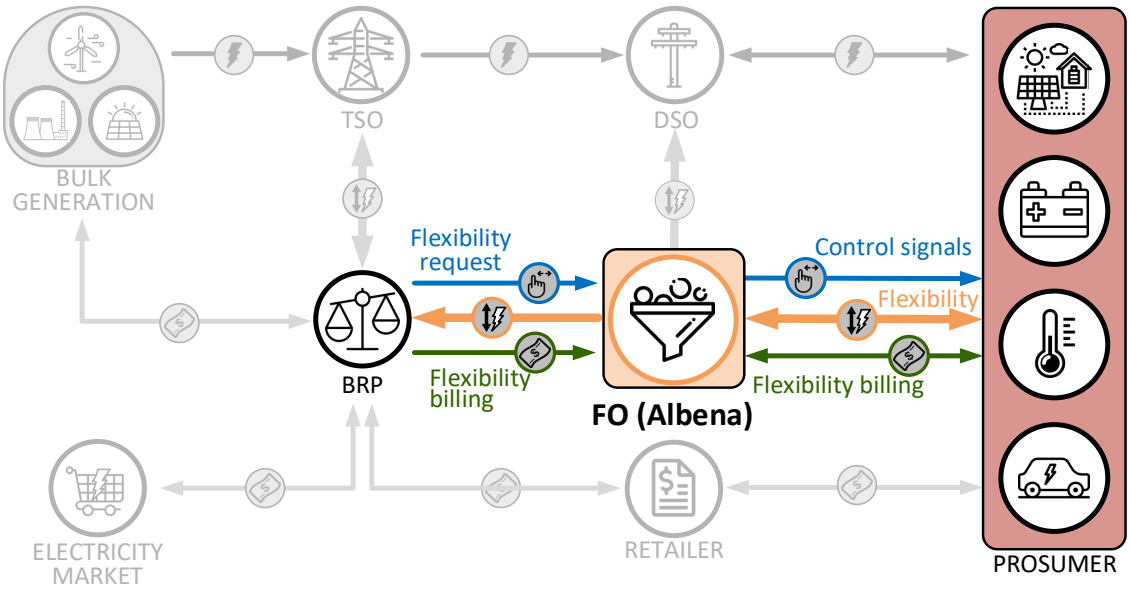


Figure 17: Generic interaction diagram of the Bulgarian pilot.

In this case, the INVADE platform interacts with the hotel devices through a SCADA, who is in charge of the interaction with the flexibility devices. The timeline of the actions to plan, execute and settle the use of flexibility is represented in Figure 18.

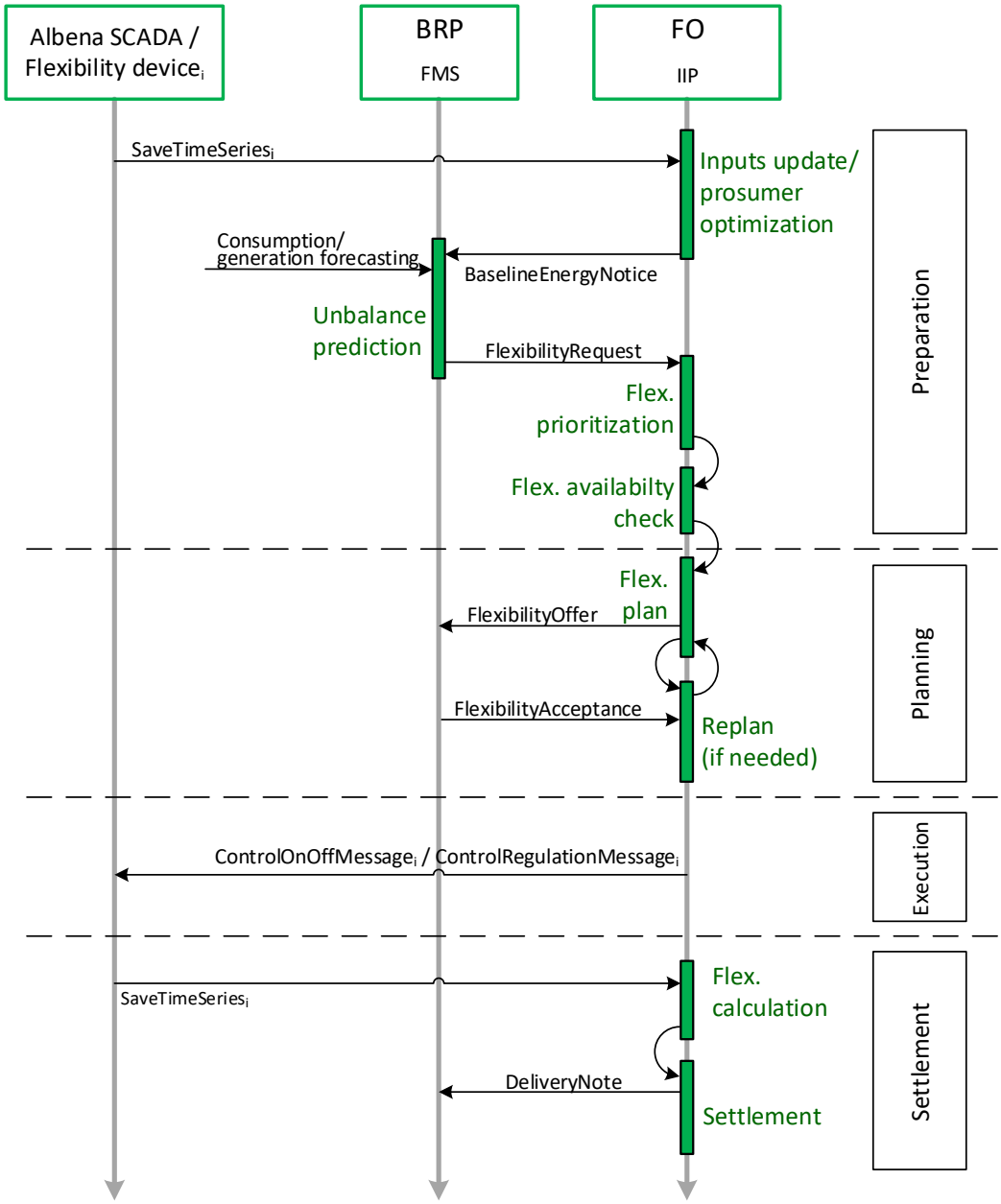


Figure 18: Detailed sequence diagram of the Bulgarian pilot.

The action sequence is as follows:

- The optimized consumption without external requests is calculated by the IIP in a similar way that the one used in the Norwegian pilot. This optimized consumption is the prosumer baseline, which can be sent from the IIP to the BRP if it helps the BRP to create its flexibility request (BaselineEnergyNotice).
- The BRP receives the portfolio forecast and estimates its future imbalances. If it is needed, it sends a flexibility request to the FO to reduce its imbalances (FlexibilityRequest).

- Based on the optimized consumption and the external requests, the new flexibility plan is generated. The FO also needs to check the available flexibility to try to accomplish the flexibility request. The flexibility offer from the IIP is sent to the BRP with an offer for available flexibility (FlexibilityOffer).
- Optionally, the BRPs can send an acceptance of the previously sent flexibility offer or plan (FlexibilityAcceptance).
- When the execution phase arrives, the FO sends the control signals to the SCADA of Albena, which is in charge of forwarding the control signals to each device.
- After making use of the flexibility, at the settlement phase the FO uses the metered values to calculate the flexibility activated by the FO to meet the BRP flexibility request. This can be calculated as the difference between the optimized forecasted profile without the external request and the metered values after the use of the external flexibility request.

4.2.3.1 Bulgarian BRP flexibility request calculation

The BRP makes constant portfolio optimizations within its balance group. Each hour, it makes decisions according to the consumption if they should buy or sell additional energy. Whenever they need, they have the possibility to order to Albena an adjustment of its consumption based on its forecasted portfolio imbalances.

The Bulgarian BRP involved in this pilot is not part of the INVADE consortium. This is the reason why the calculation of its flexibility requests is outside the scope of the project.

4.2.4 Spanish pilot

The Spanish pilot is based on the use of a centralized battery in a secondary substation to provide services to the local DSO (EPESA) and the BRP/retailer (Mercator). The retailer company, which also is the owner of the installed battery, plays the FO role in this pilot site. Figure 19 shows its generic interaction diagram.

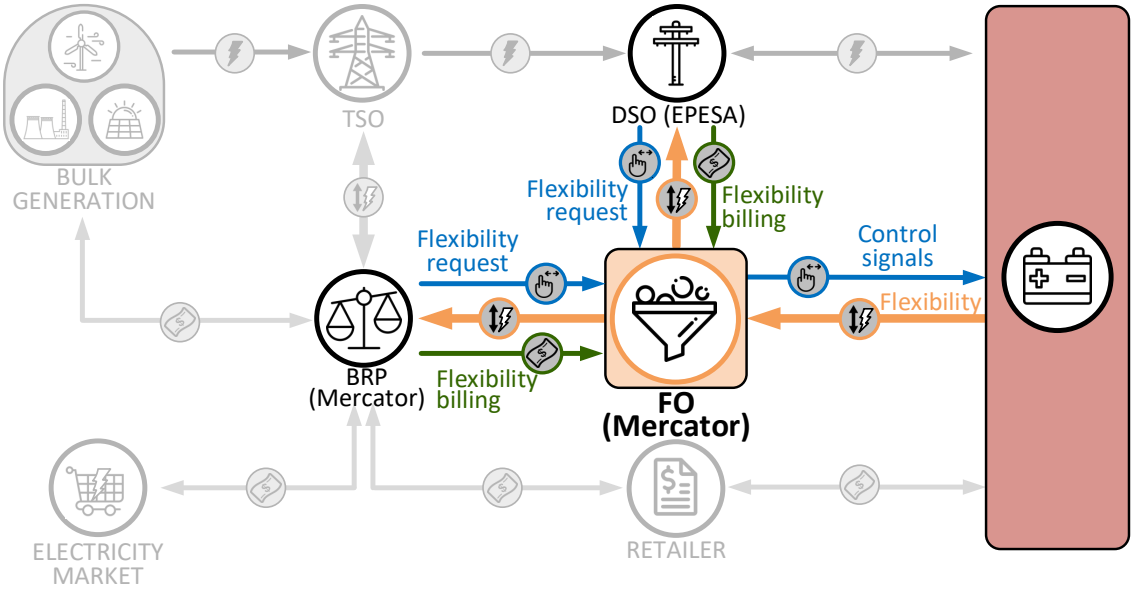


Figure 19: Generic interaction diagram of the Spanish pilot.

For the DSO, the flexibility will be used to improve the distribution system capacity of integrating renewable and distributed energy. The battery will be used to optimize the use of the flexibility for demand management and to reduce the medium voltage grid congestions. Additionally, the battery will improve the power quality of supply in the validation area where a critical building is placed, and it will permit to create an electrical island and keep supplying critical loads.

For the BRP, the flexibility will be used to self-balancing optimization, which aims to reduce imbalances of the BRP within its portfolio to avoid imbalance charges.

The timeline of the actions to plan, execute and settle the use of flexibility are represented in Figure 20. In this case, a battery is the unique flexible resource used. Scheduling tasks will be executed every 15 minutes, although it can happen hours or minutes ahead of the final delivery.

The action sequence is as follows:

- In this case, the baseline is zero as the battery is only used for aggregated services. This is the reason that makes the use of the baseline message (BaselineEnergyNotice) not a real need in this pilot case.
- The DSO receives grid metered values from its SCADA system. These values are used as inputs in the grid congestion detection algorithm to quantify the needed flexibility in the forthcoming periods. Then, the DSO sends a flexibility request to the FO if it is needed.

- Similarly, the BRP receives the portfolio forecasts and estimates the future imbalances. If it is needed, it sends a flexibility request to the FO.
- The FO receives all flexibility requests and establishes a prioritization according to the grid status.
- Before scheduling flexibility, the FO checks the available amount of flexibility from the battery.
- Then, it produces the flexibility plan containing all management signals to be sent later on. If needed, the flexibility plan can be recalculated if new requests are sent.

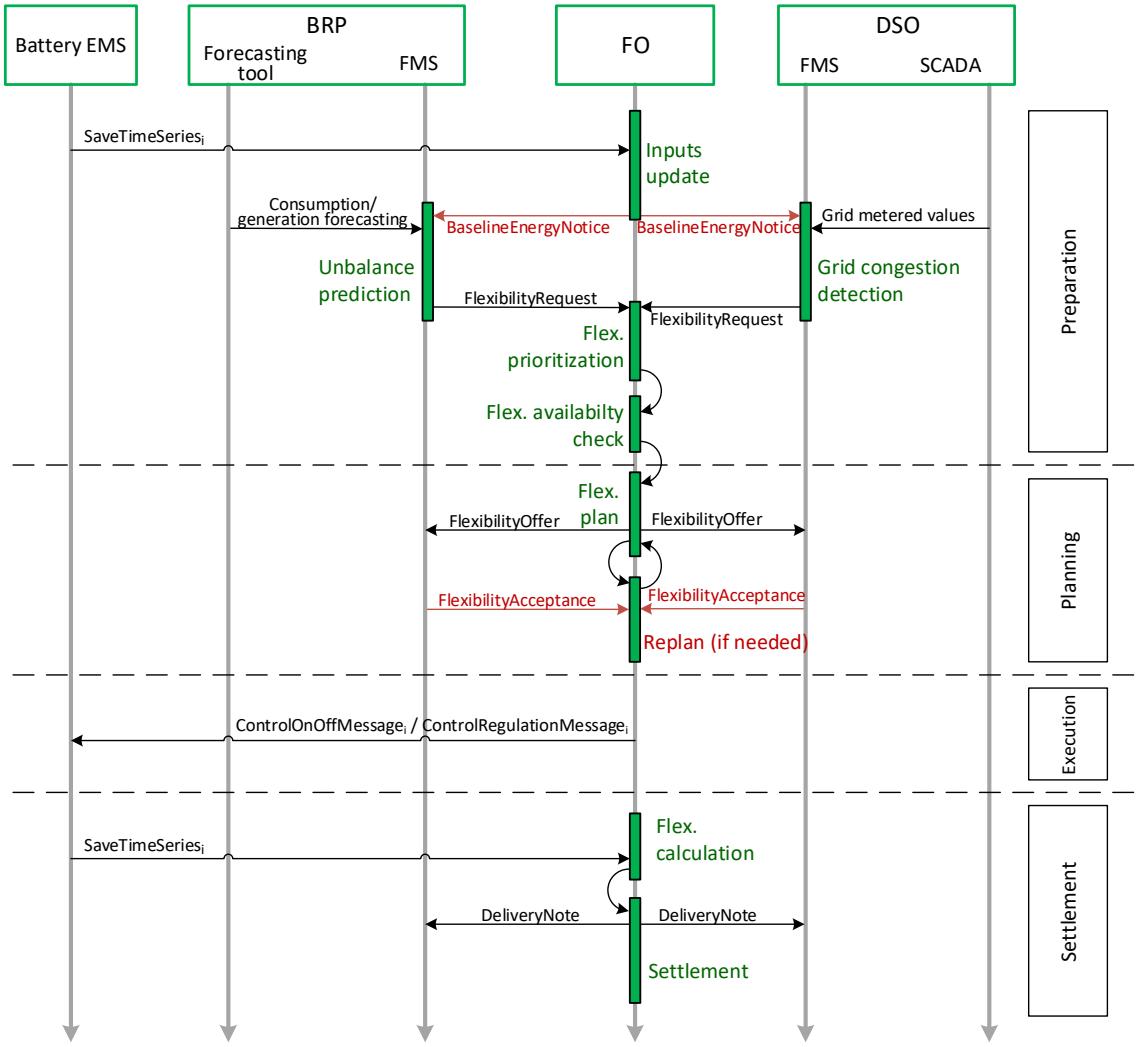


Figure 20: Detailed sequence diagram of the Spanish pilot.

Once the flexibility plan has been applied, the settlement process audits what happened during every period. The settlement process is as follows:

- The FO platform receives metered values from the used battery.

- The FO calculates the activated flexibility during the previous settlement period of operation.
- The FO sends delivery notes to the BRP and the DSO for the activated flexibility service.

If external flexibility sources are used, they also need to be paid for their activated flexibility. In this case, as the FO owns the battery, it is not needed to delivery notes and bills to the activated flexibility source owner.

4.2.4.1 Spanish DSO flexibility request calculation

The Spanish DSO is responsible for calculating their flexibility requests for congestion management purposes.

In this pilot, three scenarios for congestion management are possible:

- Congestion management in the Secondary Substation Transformer
- Congestion management in the Primary Substation Transformer
- Congestion management along the distribution grid.

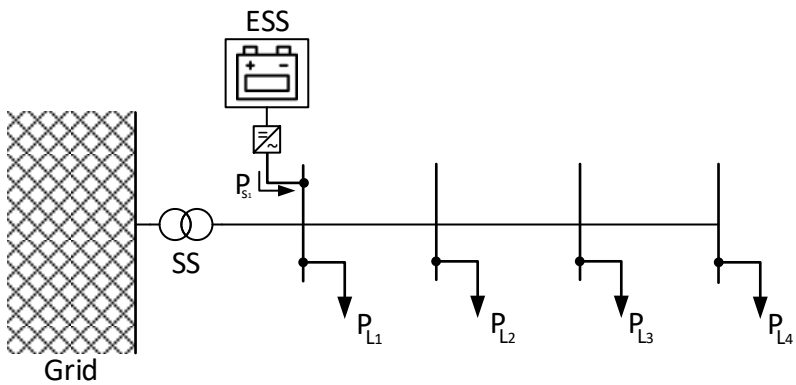


Figure 21: Spanish pilot site distribution grid scheme and flexibility injection point.

By means of charging and discharging the ESS located in the secondary substation, the congestion issues raised along the pilot site distribution grid and substation will be managed.

An interaction between the DSO and the FO is needed, to calculate the flexibility requests. The data flow is depicted in Figure 22 below. Based on the load forecast and the current grid status and topology, a power flow is done to identify the points where a congestion can occur. When a congestion is identified, the results of this power flow are sent to the optimal power flow tool to create the flexibility request to avoid the congestions

identified. Then, the flexibility requests are sent to the FO, which creates the control signals to be sent to the battery.

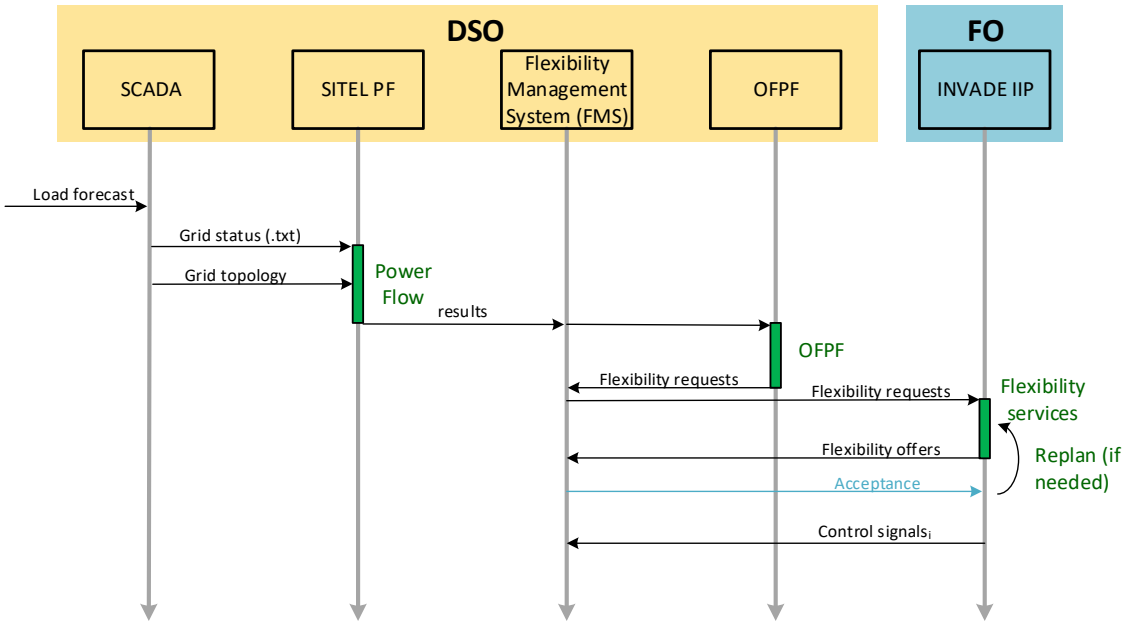


Figure 22: Flexibility requests calculation timeline.

At present, only one flexibility injection point is considered. However, the objective function of the Optimal Flexibility Power Flow (OFPF) will be set up in a generalised manner, considering more than one injection points.

$$MIN \sum_{t \in T} \xi^{flexibility}$$

Where $\xi^{flexibility}$ is the total cost for flexibility activation, which is the accorded price between the FO and the DSO.

According to Figure 22 above, there is a data exchange flow that has to be accorded and defined to execute the flexibility requests calculation under the OFPF. The pilot site distribution grid topology has to be known in advance to consider the congestion issues before calculating the flexibility requests. Related to that, there are two type of data needed for the flexibility requests calculation: static data, which should be received once; and dynamic data, which should be provided every fifteen minutes.

STATIC DATA	DYNAMIC DATA (every 15 minutes)
Nodes ID Lines characteristics	P, Q, v Demand forecast (P, Q, v)

Transformers characteristics Operating limits of the network elements	Grid topology Power flow results (congestion points)

4.2.4.2 Spanish BRP flexibility request calculation

At present, the BRP (simulated by the electric retailer Mercator) only participates in the day-ahead market to manage its portfolio. Within the INVADE pilot, the battery and the interaction with the FO are used to simulate the self-balancing portfolio optimization and therefore analysing the possibility of having economic benefits from activating flexibility to lower the BRP's portfolio imbalances.

The methodology used to create value in this case is based on the adaptation of the existing IT tools of the BRP and subsequent interaction with the INVADE platform through the FO.

Still, in a testing phase, the forecasting tool of the BRP calculates hourly predictions of the energy demand of its clients. During this timestamp, the newest prediction is compared with the previous one, and the amount of energy by which the BRP might be deviating itself from the real consumption is identified. Thereafter, the BRP makes a prediction of the deviation costs for the corresponding period, in order to identify if the flexibility request should be sent or not. This scenario intends to evaluate whether there is an economic benefit for the BRP to increase the frequency of its predictions and therefore cover the gap between closure of continuous intraday market and physical delivery of electricity to grid.

To evaluate the impact of the battery in the BRPs' business, two scenarios are created, and its costs compared. Scenario zero corresponds to the current situation, where the BRP only interacts in the day-ahead market, and it is used as a baseline scenario whereas scenario one simulates the self-balancing portfolio optimization. Costs are compared between options to evaluate the most interesting one.

4.2.5 German pilot

The German pilot is based on the use of centralized and decentralized batteries to provide flexibility services to the DSO "bn-Netze". badenova plays the FO role in this pilot site and it tries to manage the interaction of external prosumer optimized baselines and DSO services. Figure 23 shows the generic interaction diagram of this pilot.

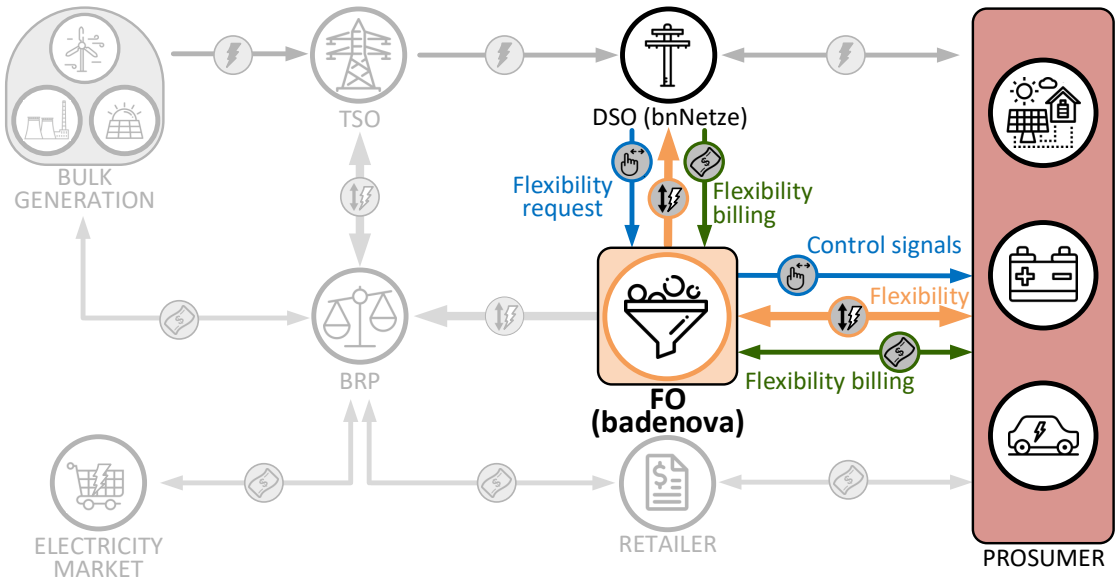


Figure 23: Generic interaction diagram of the German pilot.

In this pilot, it is needed to distinguish between the centralized and decentralized use cases. The sequence of the messages to make use of the flexibility requests from the DSO are represented in Figure 24 for the distributed use case and in Figure 25 for the centralized use case.

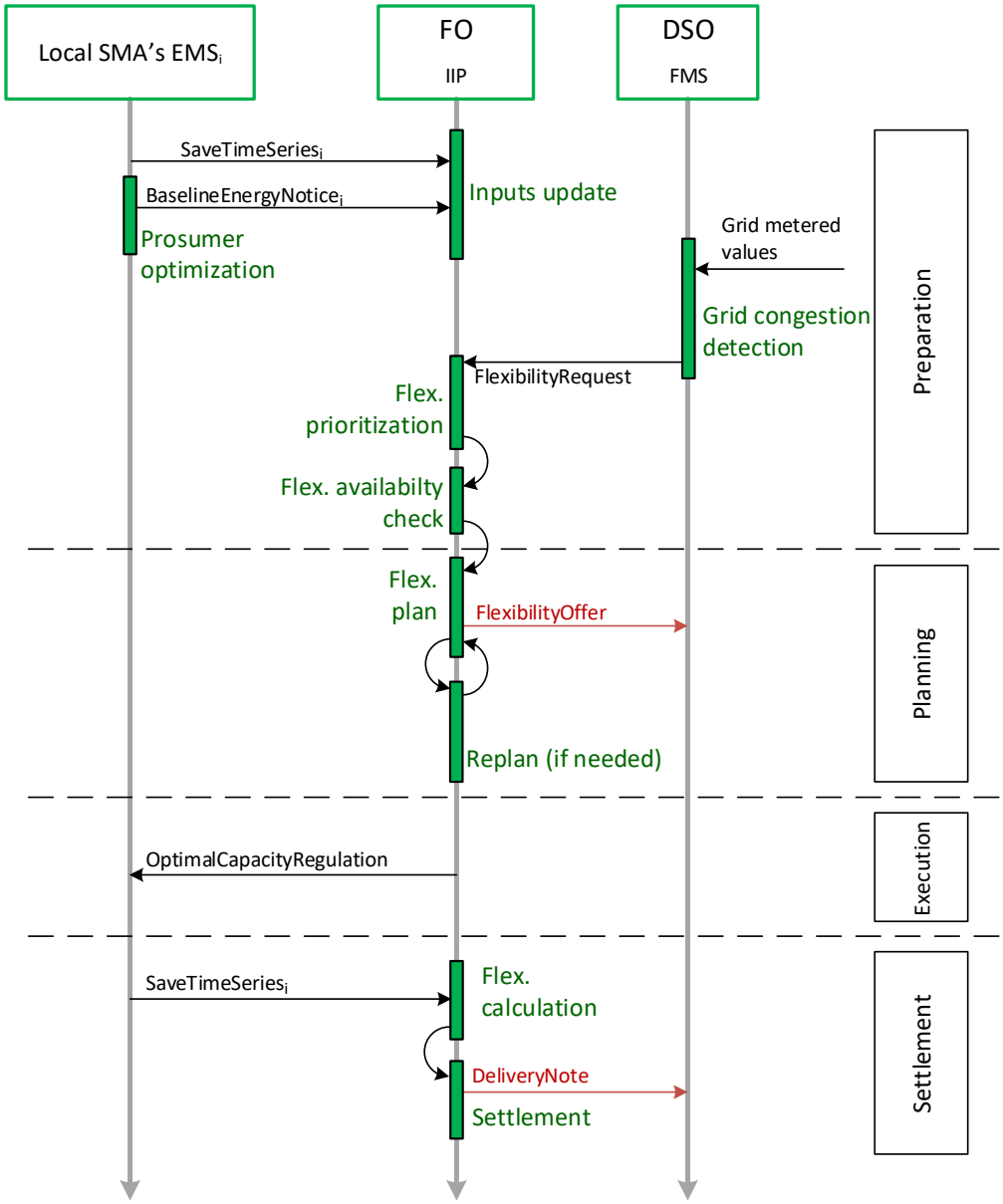


Figure 24: Detailed sequence diagram of the DES German pilot.

The action sequence for the distributed case is as follows:

- Under normal circumstances, the home management system of SMA (Sunny Home Manager) maximizes self-sufficiency of the household by using forecasts of PV production and consumption and by active control of the local storage (as well as dedicated loads as heat pumps and EV charging as far as available). This optimized consumption is the prosumer baseline, which can be sent from the SMA's EMS to the IIP (BaselineEnergyNotice). It is also needed to send information about expected SOC in the end of the last period before the FO takes over the responsibility.

- By request of the DSO (FlexibilityRequest), this normal operation mode of the Sunny Home Manager is overruled by the FO only in certain times of the year. These requests can be sent one day-ahead at the earliest, but they can be modified intraday by adapted requests. When a DSO request is received, the INVADE algorithm is responsible to reduce the consumption of the household as much as possible for a certain time period to accomplish the DSO request. When this time period is over, the Sunny Home Manager takes over control again and recalculates operation schedules for the local equipment based on its actual status.
- The flexibility offer from the IIP is sent to the DSO with an offer for available flexibility (FlexibilityOffer). It can be avoided if it is not specially required by the DSO.
- However, the INVADE platform will never control individual devices in the households. INVADE algorithm calculates the minimum net energy exchange with the main grid based on the capabilities of the components in the household and their actual status. Then, this net energy value is sent to the Sunny Home Manager (OptimalCapacityRegulation), which is in charge of defining how this restriction is going to be divided into each controlled device.
- After making use of the flexibility, the FO uses the metered values to monitor the flexibility activated as the difference between the former provided baseline and the metered values. The delivery note created by the FO is communicated to the DSO in defined intervals, summarizing what was requested and what was executed.

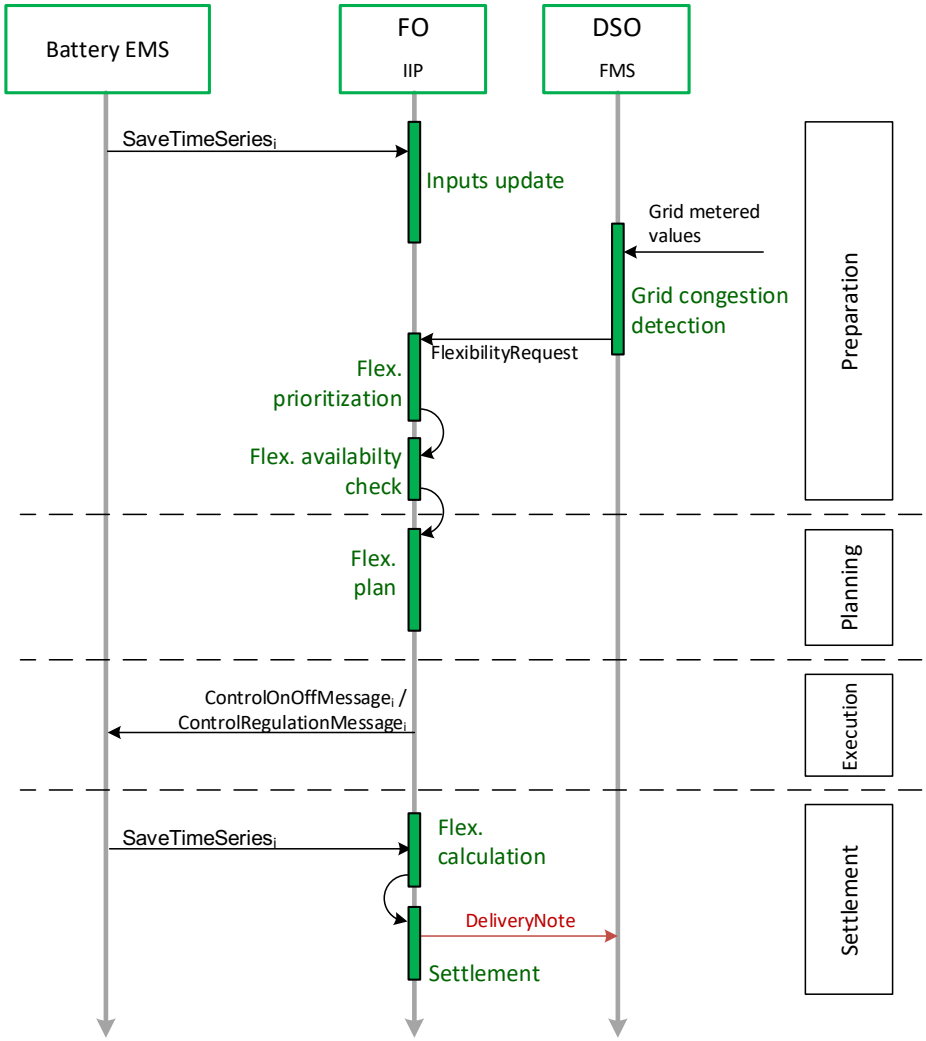


Figure 25: Detailed sequence diagram of the CES German pilot.

In a similar way, the message sequence for the centralised case is as follows:

- Under normal circumstances, the local control algorithm of the centralized battery tries to maintain the voltage under certain limits. This control has priority over external requests. As it is performed in real-time, there is no baseline usage of the battery in this case.
- By external request of the DSO (FlexibilityRequest), this operation mode can be overruled by the FO only if the voltage is maintained under a certain level. These requests can be sent one day-ahead at the earliest, but they can be modified intraday by adapted requests.
- When a DSO request is received, the INVADE algorithm is used to accomplish the DSO request. When this time period is over, the local control takes control again and recalculates operation schedules for the battery based on the voltage level.

- When the execution time arrives, the FO sends the control signal to the battery management system, which is in charge of executing them.
- After making use of the flexibility, the FO calculates the flexibility activated by comparing the power flow generated by the battery to the last value before starting execution of the request. The delivery note created by the FO is communicated to the DSO in defined intervals, summarizing what was requested and what was executed.

4.2.5.1 German DSO flexibility request calculation

In the German hybrid use case, the DSO flexibility requests will only ask to lower the load demand for electricity as the main target is to avoid power peak fees from the upstream network operator (peak shaving).

Lowering the peak load demand from the upstream network operator can be achieved with two different measurements:

- Down regulation of consumption in the distribution grid
- Increasing the energy generation in the distribution grid

As of now, there is no financial benefit for the DSO to increase the consumption in any entire zone of the distribution grid.

The DSO runs each day a load forecast for the next day regarding the entire electrical grid. This d-1 prognosis is the trigger for the flexibility request. If it is foreseen, that on the next day an accountable power peak is to be expected, the DSO sends a flexibility request to the FO. This request uses the theoretical maximum controllable power of all flexible devices controlled by the FO as a baseline. The FO is responsible to calculate the real available controllable power in the given time frame and send it back to the DSO in form of the above described flexibility offer.

5 Standards and communication protocols

This section includes a brief description of the standards and communication protocols that have been developed within the project. More on the work and progress with regard to protocols and standards can be found in deliverable D3.3.

5.1 New OCPP v2.0 protocol for integrated smart charging and V2G

The past year has seen the release of OCPP 2.0 [6] as a specification, implementation guide and some clarifications. These documents have been published on the Open Charge Alliance website (https://www.openchargealliance.org). Elaad and Greenflux, as members of the Open Charge Alliance, are now actively supporting implementations and tenders involving OCPP. In addition to this, addendums and clarifications to the older OCPP 1.6 have also been created and published.

5.2 New OCMP 1.0 for information between EMSP and the DSO

The goal of this new standard is to describe a protocol for smart charging electrical vehicles based on available capacity that is divided by the DSO. Although the Open Smart Charging Protocol (OSCP) protocol offers these kind of functionalities, Open Capacity Management Protocol (OCMP) [7] is complementary to OSCP 1.0. OCMP is a development name to the Invade pilot project specific version of OSCP. This protocol will be used as input for the new OSCP2.0 protocol to be published also by the Open Charge Alliance.

As it is still not published by the Open Charge Alliance, OCMP specifications can be found as an appendix at section 9 of the Annex.

6 Conclusions

This document describes the updates of the concept design and architecture of the pilots focused on the second implementation phase, which covers the aggregated services for the DSOs and BRPs.

It first identifies the changes and updates that have been recognized in all pilots with respect to the last version of D4.2 [2]. In D4.2 there were several uncertainties with the aggregated services for the DSO and/or BRP that are going to be implemented. The current summary of flexibility services that are going to be used in each pilot can be seen in Table 5.

Table 5: Flexibility services to be used in each pilot (Y: yes; N: no).

Flexibility customer	Flexibility services INVADE	Norwegian pilot	Dutch pilots	Bulgarian pilot	German pilot	Spanish pilot
DSO	Congestion management	N	Y	N	Y	Y
	Voltage / Reactive power control	N	Y	N	Y	Y
	Controlled islanding	N	N	N	N	Y
BRP	Day–ahead portfolio optimization	N	N	N	N	N
	Intraday portfolio optimization	N	N	N	N	N
	Self-balancing portfolio optimization	N	N	Y	N	Y
Prosumer	ToU optimization	Y	Y	Y	N	N
	kWmax control	Y	Y	Y	N	N
	Self-balancing	Y	Y	Y	Y[3]	N
	Controlled islanding	N	N	N	N	N

According to these services, the objective of each pilot can be summarized in the following way:

- Norwegian pilot: it develops standalone prosumer optimization.
- Dutch pilot: it tests the interaction of internal prosumer optimization based on EVs and DSO services.
- Bulgarian pilot: it is based on examining the interaction of internal prosumer optimization and BRP services.
- Spanish pilot: it shows how to solve the interaction of DSO and BRP services.

[3] Prosumer self-balancing service is going to be supplied by SMA outside the scope of the INVADE project.

- German pilot: it demonstrates how to integrate external prosumer optimized baselines and DSO services.

To accomplish with these services, the sequence diagram has been defined for each pilot to specify the messages, timeline and information exchanged.

7 References

[1] P. Lloret, P. Olivella, G. T. Berger, J. Timbergen, and P. Rademakers. D4.1 Overall INVADE architecture, INVADE project, 2017.
[2] P. Lloret, P. Olivella. D4.2 v2.0 INVADE architecture of pilots, INVADE project, 2018.
[3] Flo Bødal. Espen, P. Crespo del Granado, H. Farahmand, M. Korpås, P. Olivella, I. Munné, and P. Lloret, "D5.1 Challenges in distribution grid with high penetration of renewables." INVADE project, 2017.
[4] Olivella-Rosell, P.; Bullich-Massagué, E.; Aragüés-Peñalba, M.; Sumper, A.; Ottesen, S.Ø.; Vidal-Clos, J.A.;Villafáfila-Robles, R. Optimization problem for meeting distribution system operator requestsin local flexibility markets with distributed energy resources. Appl. Energy 2018, 210, 881–895, doi:10.1016/j.apenergy.2017.08.136.
[5] Olivella-Rosell, P.; Lloret-Gallego, P.; Munné-Collado, Í.; Villafafila-Robles, R.; Sumper, A.; Ottessen, S.Ø.; Rajasekharan, J.; Bremdal, B.A. Local Flexibility Market Design for Aggregators Providing Multiple Flexibility Services at Distribution Network Level. Energies 2018, 11, 822.
[6] Open Charge Point Protocol (OCPP) 2.0. Open Charge Alliance, 2018. www.openchargealliance.org/downloads
[7] Open Capacity Management Protocol (OCMP) 1.0 - Specification. Open Charge Alliance, 2018.

8 Annex 1. Detailed updates of pilot's architecture

8.1 Dutch pilot

8.1.1 Communication layer

As described in D4.2 [2], the Dutch pilot can also be divided into 4 different pilot cases:

- Large scale offices case
- Large scale public case
- Small scale home case
- Small scale public office case

The communication SGAM layers with the protocols used in each pilot case are represented in Figure 26, Figure 27, Figure 28 and Figure 29.

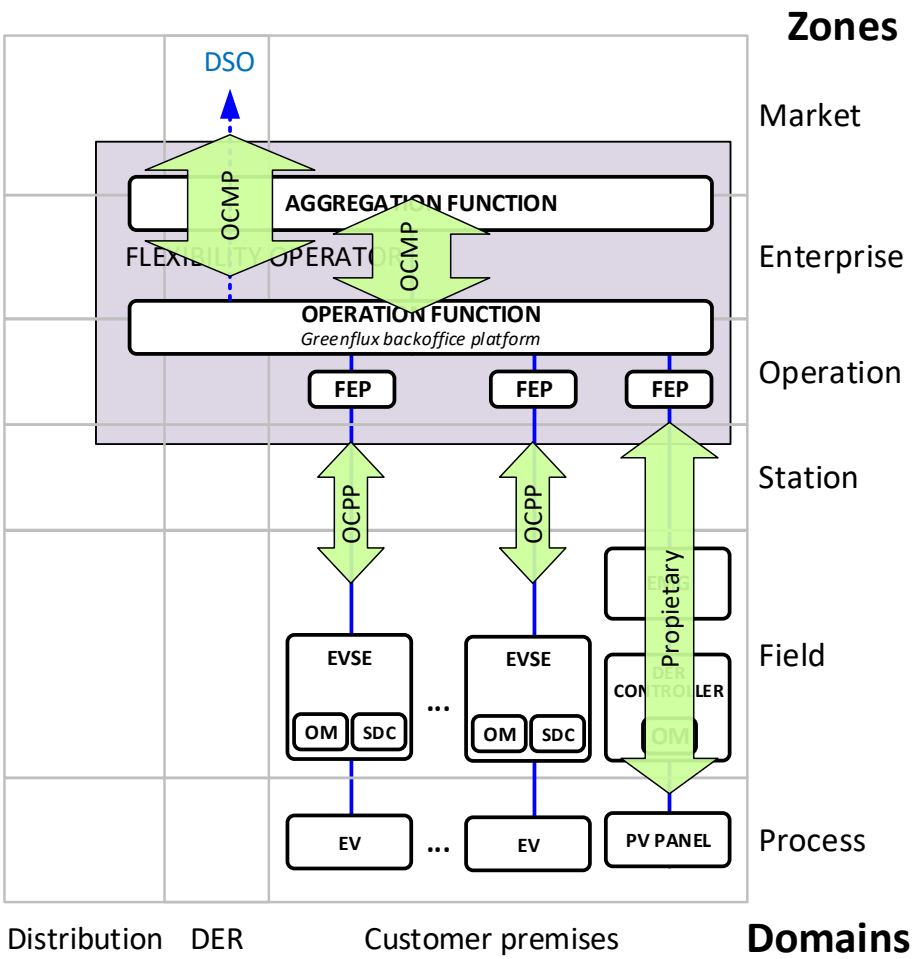


Figure 26: SGAM communication layer in the large-scale office in the Dutch pilot.

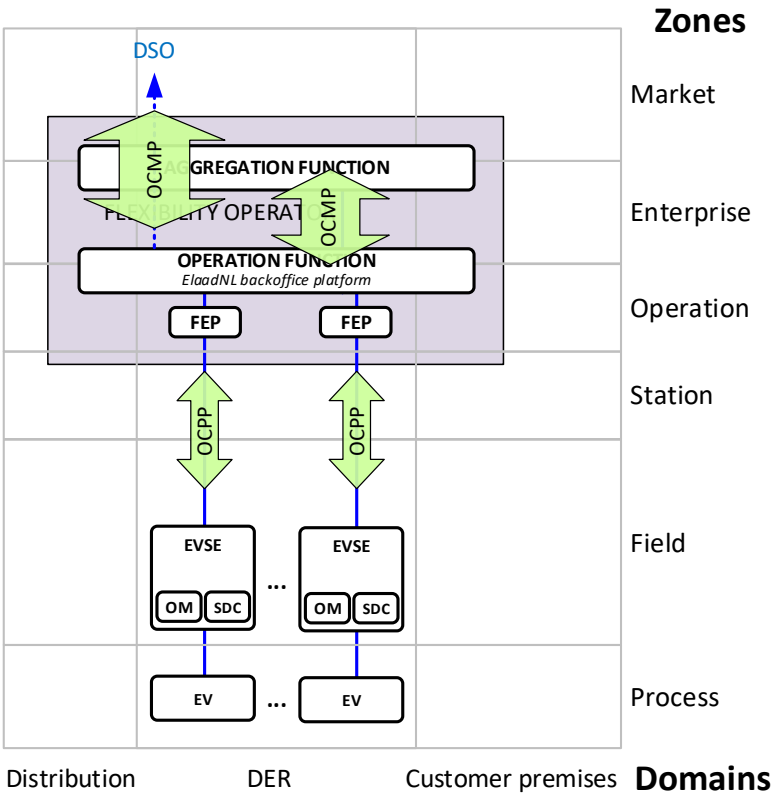


Figure 27: SGAM communication layer in the large-scale public Dutch pilot.

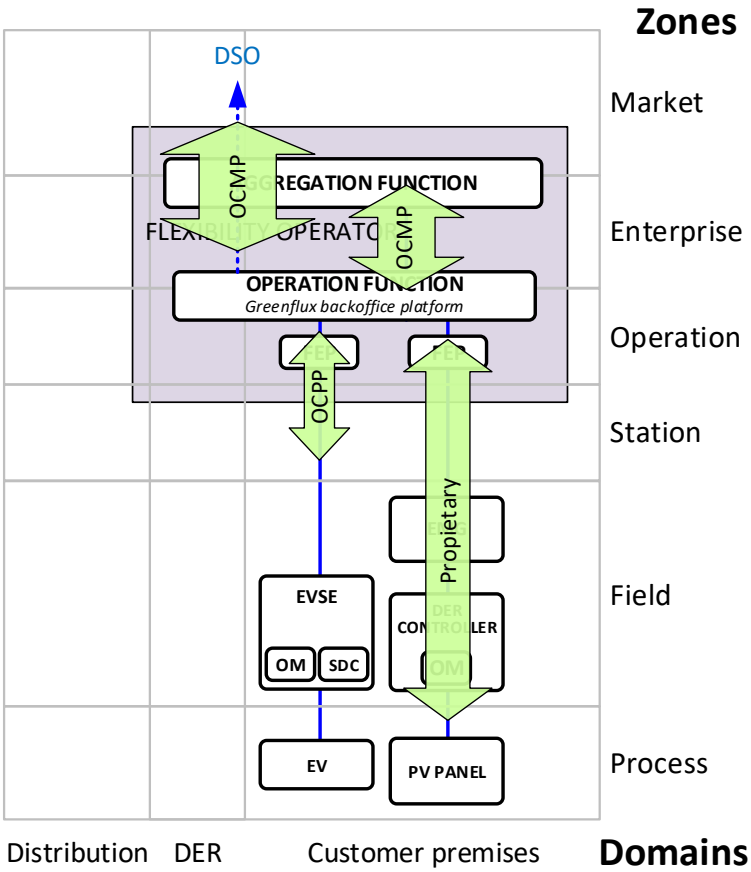


Figure 28: SGAM communication layer in the small-scale home Dutch pilot.

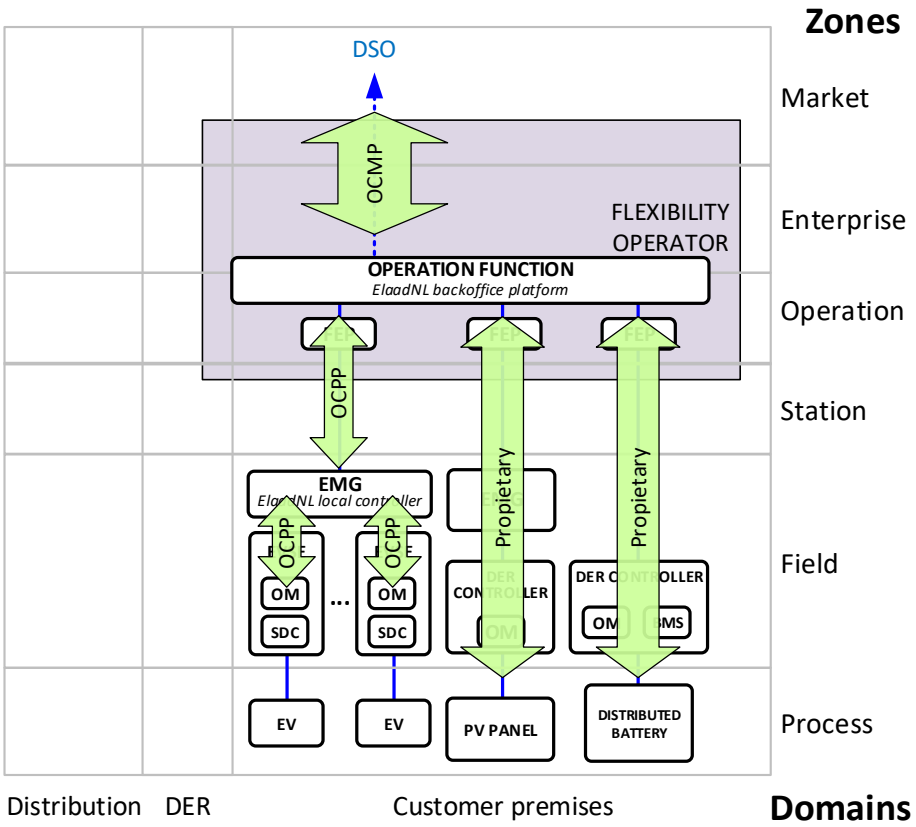


Figure 29: SGAM communication layer in the small-scale public & office Dutch pilot

8.2 Spanish pilot

8.2.1 Component layer

The component SGAM layer diagram for the Spanish pilot is showed in Figure 30.

8.2.2 Communication layer

The communication SGAM layer diagram for the Spanish pilot is showed in Figure 31.

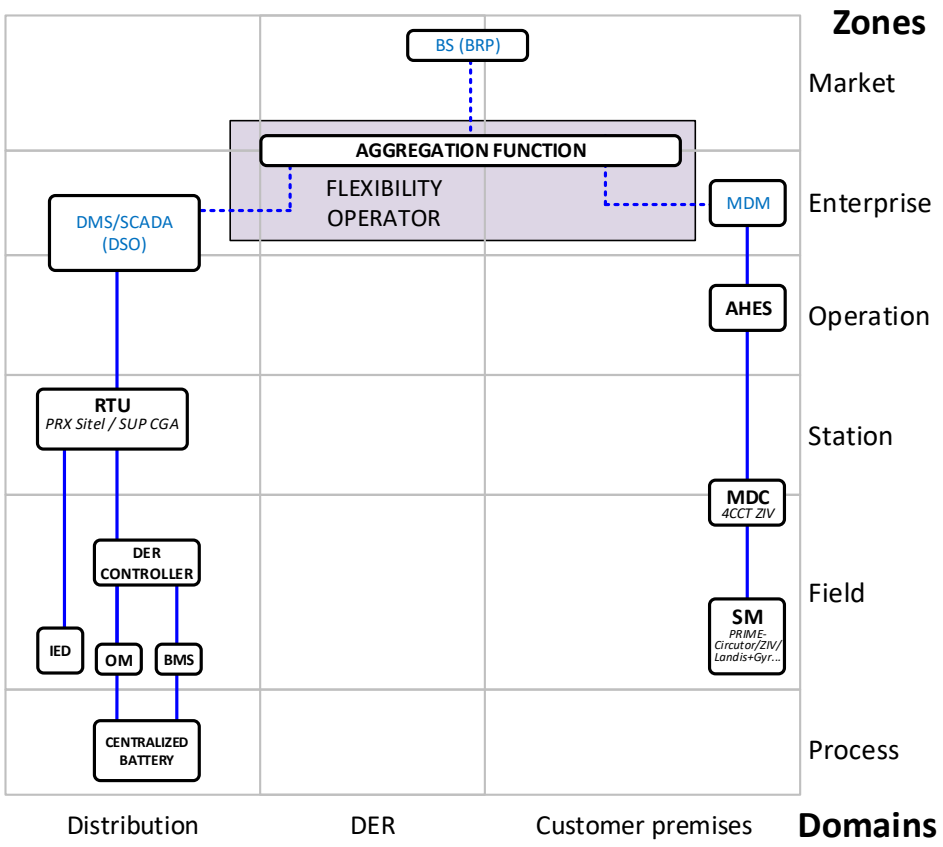


Figure 30: SGAM component layer in the Spanish pilot.

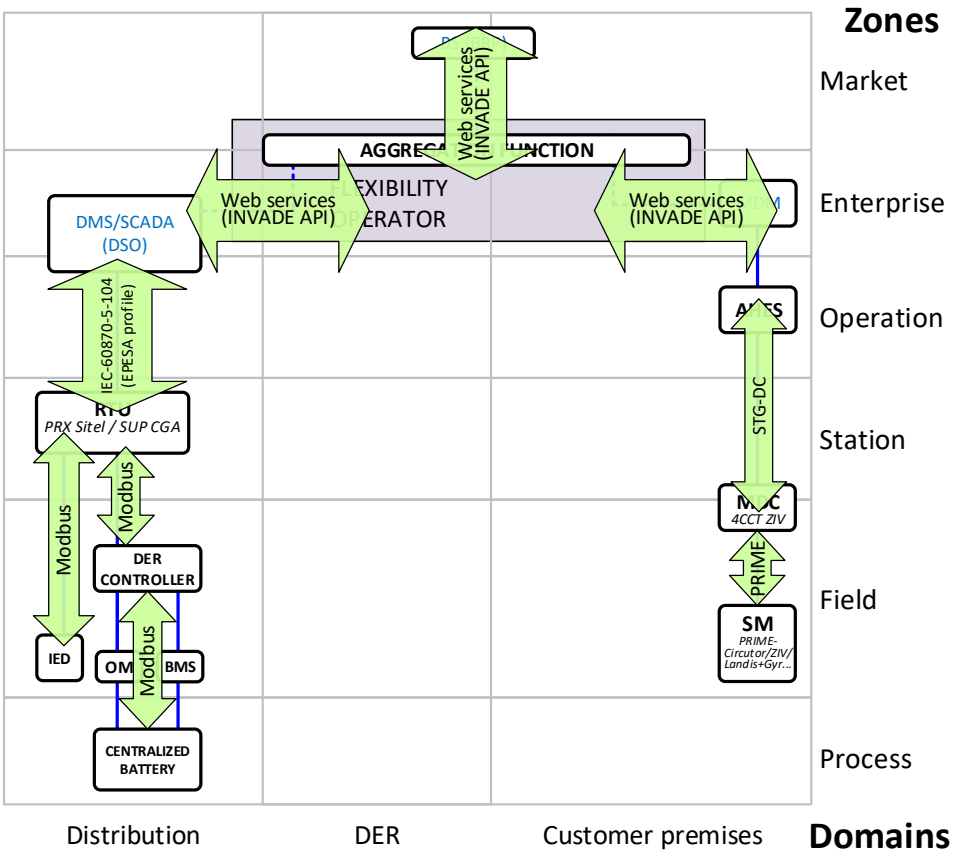


Figure 31: SGAM communication layer in the Spanish pilot.

8.3 German pilot

8.3.1 Communication layer

The communication SGAM layer diagrams for the German pilot are showed in Figure 32 and Figure 33.

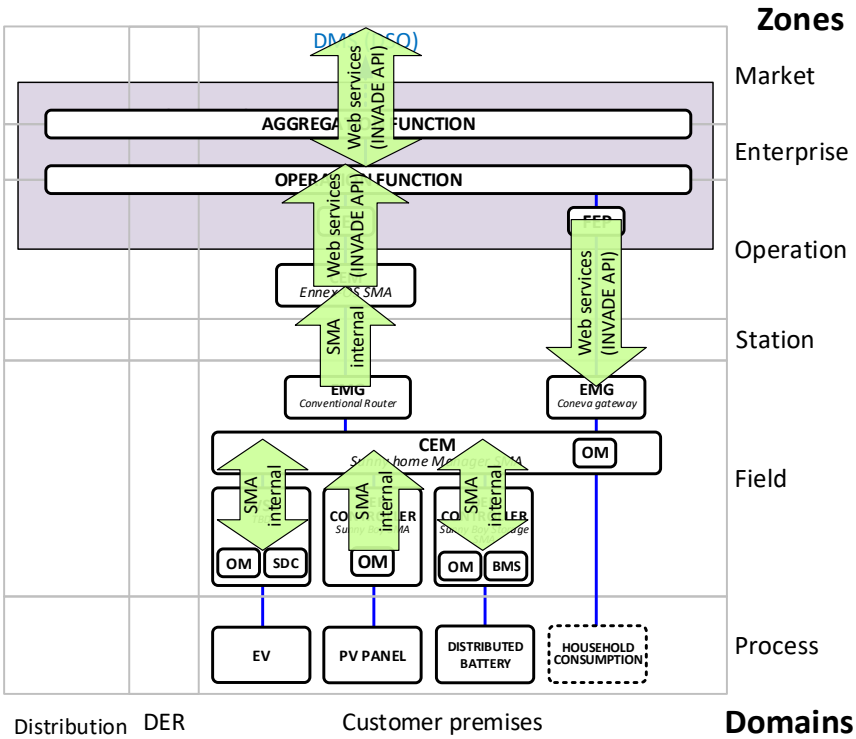


Figure 32: SGAM communication layer in the German pilot for DES.

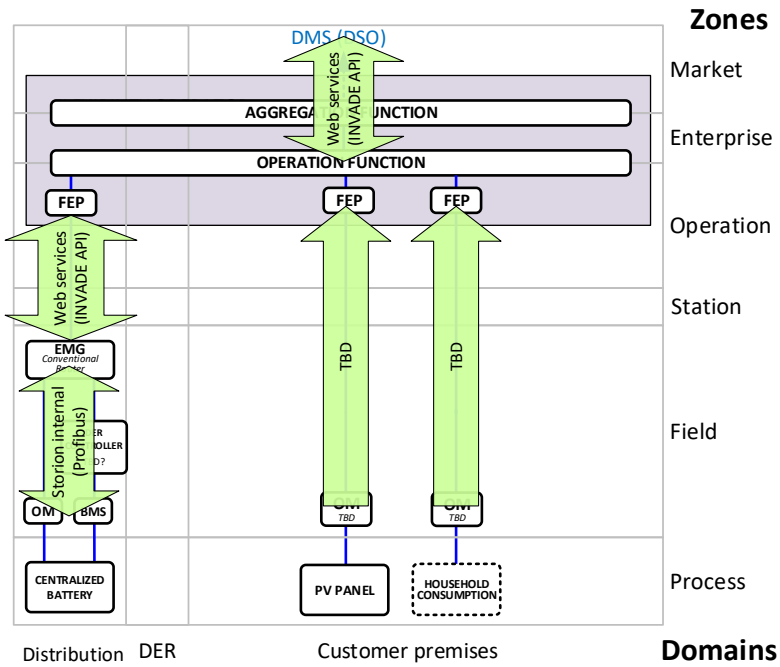


Figure 33: SGAM communication layer in the German pilot for CES.

9 Annex 2. OCMP 1.0 - Specification

OCMP 1.0 - Specification

2018-04-09

Table of Contents

Disclaimer

This OCMP 1.0 document is a development name to an Invade pilot specific version of OSCP. It is based on OSCP 1.0 and some extensions. OCMP can be used by Open Charge Alliance as draft proposal for OSCP 2.0 development.

1. Introduction

The goal of this document is to describe a protocol for smart charging electrical vehicles based on available capacity that is divided by the DSO. Although the *Open Smart Charging Protocol* (OSCP) protocol offers these kind of functionalities *Open Capacity Management Protocol* (OCMP) is complementary to OSCP 1.0. OCMP is a development name to the Invade pilot project specific version of OSCP.

OCMP can be used as draft proposal for OSCP 2.0 development.

1.1. Scope

This document describes the message flow defined for smart charging. The scope of this document is primarily focused on the communication between *Distribution System Operator* (DSO) system and a *Charging Station Operator* (CSO). Though the roles of DSO and CSO are expanded with the role of *Capacity Management Operator* (CMO).

The communication between the CSO and the local controller / Charging Stations are out of scope of this document. This is addressed in OSCP.

This document mentiones DSO adressing multiple CSOs. This is not in scope of the Invade pilot project; only addressing a single CSO is supported.

1.2. Version history

Version	Date changed	Author	Change
1.0 Draft 1	2018-02-23	B. Praster	Initial draft incl. changes from Greenflux.
1.0 Draft 2	2018-02-28	B. Praster, R. Hagenaars	Various clarifications.
1.0 Draft 3	2018-03-09	B. Praster, R. Hagenaars	Changes in measuring models, added JSON schemas and some business rules.
1.0 Draft 4	2018-03-16	B. Praster, R. Hagenaars	Processed review comments M. Fisher and O. Amundsen (eSmart). Introduced EnergyFlowDirection.
1.0 Draft 5	2018-03-29	B. Praster, R. Hagenaars	Processed review comments M. Fisher and O. Amundsen (eSmart).
1.0	2018-04-09	B. Praster, R. Hagenaars	Processed review comments O. Amundsen (eSmart) and P. Klapwijk (ElaadNL)

1.3. Conventions

1.3.1. Primitive Datatypes

The primitive datatypes comply to the JSON standard. The JSON format supports the following.

Datatype	Description
string	double-quoted Unicode (UTF-8) with backslash escaping. For limitations see String terminology.
number	integer or double precision floating-point format
object	an unordered collection of key:value pairs
array	an ordered sequence of values
boolean	true or false
null	empty

1.3.2. Cardinalities

The column "card." should be interpreted as follows.

Cardinality	Description
0..1	The field is optional. Empty fields must be omitted; fields with value *null* are prohibited unless explicitly stated otherwise.

Cardinality	Description
1..1	The field is mandatory.
0..*	The field holds an array (of a given type) with zero or more entries. In case of zero entries, an empty array must be given.
1..*	The field holds an array (of a given type) with one or more entries.

1.3.3. Use Cases

Unless otherwise specified, all use cases and requirements assume normal communication between DSO, CSO and CMO.

1.4. Terminology

This section contains the terminology that is used throughout this document.

Terminology	Description
Group capacity	This is the maximum capacity (in Amps) that can be fed to an aggregation before something goes wrong (burning fuse or damaging cables). A group is a representation of an aggregated area.
groupId	A group representing an aggregated area can be uniquely identified using this ID.
Charging Station	The Charging Station is the physical system where an EV can be charged. A Charging Station has one or more EVSEs.
CSO	Charging Station Operator also known as Charge Point Operator (CPO). It is the party that operates a network of charging stations and has contracts with EMSPs to allow their customers to use the charging facilities.
DSO	Distribution System Operator. The DSO manages the distribution network and has the interest of not overloading the (local) grid.
CMO	Capacity Management Operator. This is a 3rd party that can be optionally involved to improve on charging profiles.
EVSE	An EVSE is considered as an independently operated and managed part of the Charging Station that can deliver energy to one EV at a time.
Datetime	String representation of UTC dates and times. Please refer to ISO8601.
String	Case Sensitive string value in UTF-8. All strings in messages and enumerations are case sensitive, unless explicitly stated otherwise. All enumerations are written in uppercase.

1.5. Abbreviations

Table 1. Abbreviations

Abbreviation	Description
CMO	Capacity Management Operator
CS	Charging Station
CSO	Charging Station Operator
DSO	Distribution System Operator
EV	Electric Vehicle
EVSE	EV Supply Equipment
JSON	JavaScript Object Notation
HTTP(S)	HyperText Transport Protocol (Secure)
UTC	Coordinated Universal Time
V2G	Vehicle To Grid

1.6. Actors

In OCMP, system actors are covering functions or devices.

Table 2. Actors

Actor name	Actor type	Actor description
EV Driver	Actor	The Driver of an EV who wants to charge the EV at a Charging Station.
CSO	Subsystem	The Charging Station Operator manages Charging Stations and has the information for authorizing Users for using its Charging Stations.

Actor name	Actor type	Actor description
CS	Device	The Charging Station is the physical system where an EV can be charged. A Charging Station has one or more EVSEs.
Electric Vehicle	Device	Electric Vehicle, distributed energy resource with a remote battery and socket.
DSO	Actor	A Distribution System Operator.
CMO	Subsystem	Capacity Management System: is optionally involved to improve on charging profiles.

1.7. References

Reference	Description
[IEC62559-2:2015]	Definition of the templates for use cases, actor list and requirements list. https://webstore.iec.ch/publication/22349
[OSCP10]	Open Smart Charging Protocol 1.0 (final) OCA, 2015
[ISO8601]	"Date and time format" http://www.iso.org/iso/home/standards/iso8601.htm
[JSON]	A data exchange standard. https://www.json.org/
[JSON-style-guide]	A style convention on JSON. https://www.w3resource.com/slides/json-style-guide.php
[UTF8]	Character encoding. https://tools.ietf.org/html/rfc3629

1.8. Time Format Requirements

Date/time formatting related matters can be found in terminology Datetime.

1.9. Message timeouts

OCMP does not specify timing requirements.

2. Use Cases

2.1. DSO distributes a capacity to a CSO

No.	Type	Description
1	**Name**	DSO distributes an amount of capacity to a CSO.
2	**ID**	1
3	**Objectives**	Enable the DSO to distribute the maximum available capacity for electricity usage to a CSO. The CSO will in turn refine the capacity model to an opmimal capacity model using the CMO.
4	**Description**	To prevent network congestion, the DSO can measure the load on the local electricity network and based on the maximum capacity of the network components, the DSO can send out the maximum available capacity for electricity usage.
	Actors	DSO, CSO, CMO, CS
	Scenario description	**1.** The DSO receives measurements from the transformer station (out of document scope). **2.** Based on these measurements a forecast of the available group capacity is made. This group capacity is forwarded to the CSO using the UpdateGroupMaximumCapacityForecast message. **3.** The CSO redirects UpdateGroupMaximumCapacityForecast message to the CMO. **4.** After CMO has calculated the optimal capacity forecast, it is sent to the CSO using the UpdateGroupOptimalCapacityForecast message. **5.** Based on this forecast, the CSO can tell its Charging Stations what their (maximum) capacity for the next period of time is. Subsequently, the EVs at the Charging Stations can be charged, be it throttled, depending on the capacity. **6.** Meanwhile for a certain to be determined period, CSO is assembling metering data from the Charging Stations. **7.** The CSO combines all metering data assembled over the last period and wraps it in an UpdateDetailedMeteringData message that is sent to the CMO. **8.** Same goes for the DSO, but since it has an aggregated interface, the CSO sends an UpdateAccumulatedMeteringData message to it.
5	**Prerequisites**	The CSO has Smart Charging enabled Charging Stations. The Handshaking mechanism was successful between the DSO and CSO. The Handshaking mechanism was successful between the CSO and CMO. All Charging Stations report their active EV session metering data to the CSO in time.
6	**Postcondition(s)**	**Successful postcondition:** The CSO has received a forecast of optimal available capacity and has, depending on the capacity, throttled one or more charge sessions. The CMO has received detailed metering data. The DSO has received accumulated metering data. **Failure postcondition:** The forecast of available maximum capacity has not arrived at the CSO. The forecast of optimal capacity has not arrived at the CSO. The detailed metering data has not arrived at the CMO. The accumulated metering data has not arrived at the DSO.

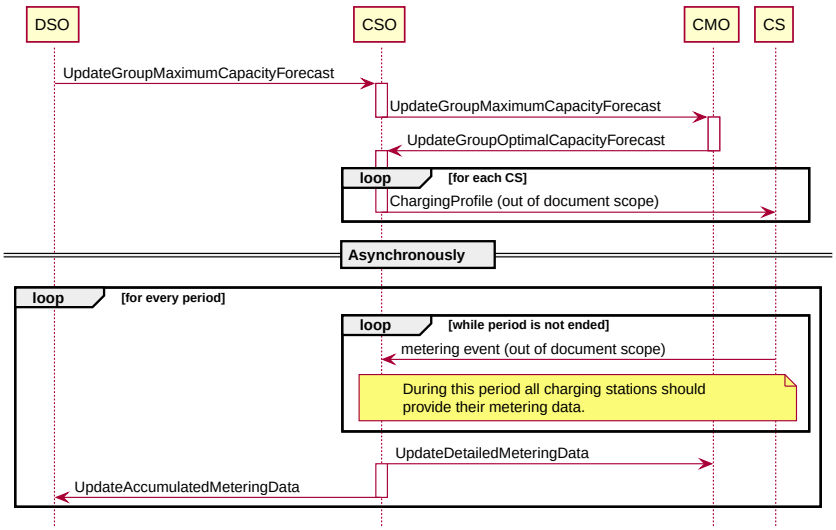


Figure 1. Sequence Diagram: DSO distributes capacity to a CSO

7	**Error Handling**	n/a
8	**Remarks**	n/a

2.2. Handshaking

2.2.1. Handshaking between DSO and CSO

No.	Type	Description
1	**Name**	DSO initiates a handshake with a CSO
2	**ID**	2
3	**Objectives**	Reveal the base url of the DSO and the CSO. Define the heartbeat intervals.
4	**Description**	The DSO and CSO should come to an agreement on the base url and heartbeat intervals. The Handshaking mechanism allows for this.
	Actors	DSO, CSO
	Scenario description	**1.** The DSO sends a Handshake. **2.** The CSO responds with a HandshakeAcknowledge using the corresponding base url that was defined in the request.
5	**Prerequisites**	The DSO knows the initial endpoint on which handshaking can be initiated.
6	**Postcondition(s)**	**Successful postcondition:** The DSO knows the CSO's base url for OCMP messaging. The CSO knows the DSO's base url for OCMP messaging. The DSO sends heartbeats with an interval as defined by the CSO. The CSO sends heartbeats with an interval as defined by the DSO. **Failure postcondition:** The Handshake is not responded to with a HandshakeAcknowledge. Not all endpoints are known.

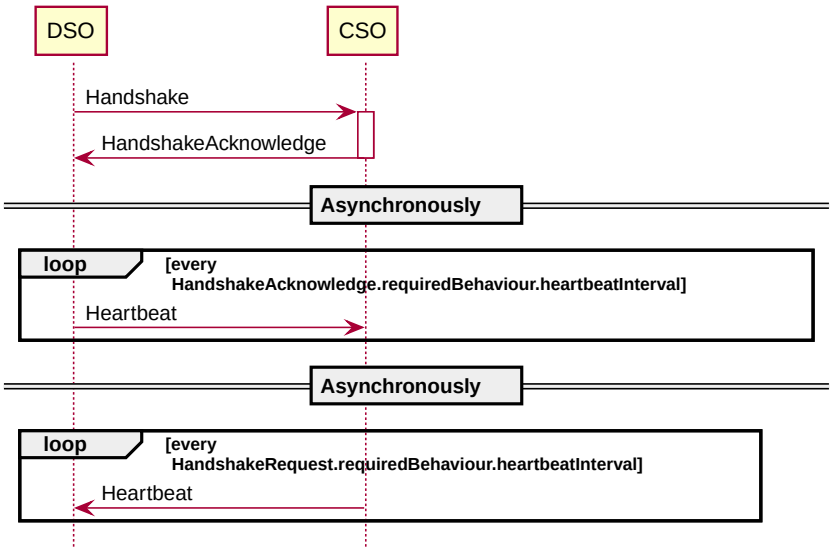


Figure 2. Sequence Diagram: DSO initiates a handshake with a CSO

7	**Error Handling**	n/a
8	**Remarks**	n/a

2.2.2. Handshaking between CSO and CMO

No.	Type	Description
1	**Name**	CSO initiates a handshake with a CMO
2	**ID**	3
3	**Objectives**	Reveal the base urls of the CSO and the CMO. Define the heartbeat intervals.
4	**Description**	The CSO and CMO should come to an agreement on the base urls and heartbeat intervals. The Handshaking mechanism allows for this.
	Actors	CSO, CMO

	Scenario description	**1.** The CSO sends a Handshake. **2.** The CMO responds with a HandshakeAcknowledge using the corresponding base url that was defined in the request.
5	**Prerequisites**	The CSO knows the initial endpoint on which handshaking can be initiated.
6	**Postcondition(s)**	**Successful postcondition:** The CSO knows the CMO's base url for OCMP messaging. The CMO knows the CSO's base url for OCMP messaging. The CSO sends heartbeats with an interval as defined by the CMO. The CMO sends heartbeats with an interval as defined by the CSO. **Failure postcondition:** The Handshake is not responded to with a HandshakeAcknowledge. Not all endpoints are known.

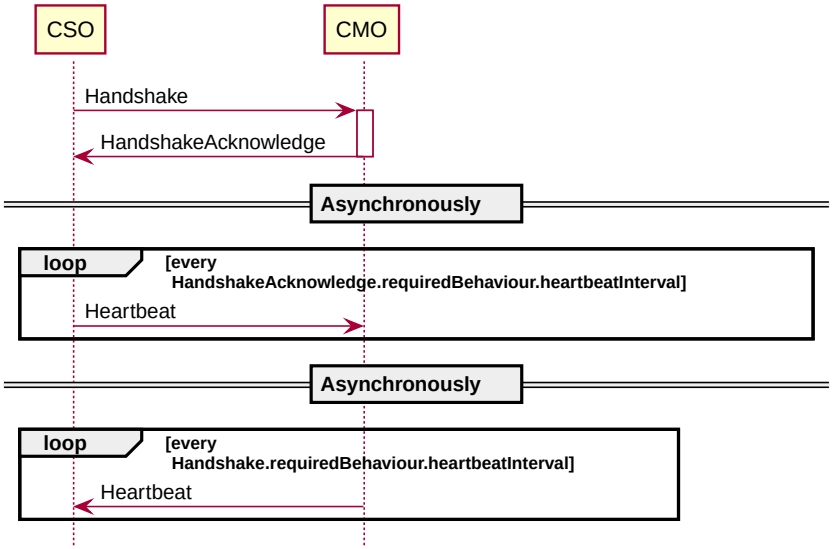


Figure 3. Sequence Diagram: CSO initiates a handshake with a CMO

7	**Error Handling**	n/a
8	**Remarks**	n/a

3. Messages

In this chapter the messages are discussed in detail.

All messages are JSON objects and may be extended at will, as long as the schemas are respected.

3.1. General

3.1.1. HTTP Requests

Envelope

All HTTP request messages are wrapped in a request envelope as described below.

Definition

Field Name	Field Type	Card.	Description
id	string	1..1	A unique identifier of this message.
data	object	1..1	The wrapped content of this message.
referenceId	string	0..1	Optional. The id of an earlier message that is referred to.
priority	number	0..1	Optional. This field determines the importance of this message, where values < 0 are less important and values > 0 are more important. Omitting this field is effectively equal to a priority of 0.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/envelope.json",
  "title": "Envelope",
  "type": "object",
  "properties": {
    "id": {
      "type": "string"
    },
    "data": {
      "type": "object"
    },
    "referenceId": {
      "type": "string"
    },
    "priority": {
      "type": "integer"
    }
  },
  "required": [
    "id",
    "data"
  ]
}
```

3.1.2. HTTP Responses

Valid response

If an HTTP request is considered valid and no error has occurred, an HTTP response without payload must be returned with HTTP status code **200**.

ErrorEnvelope

In case of an error, an error response envelope *may* be returned with a corresponding HTTP status code.

HTTP status code **500 Internal Server Error** however is not expected to contain an error envelope. The reason is that this status code suggests a serious problem at the *recipient's* side. In most cases, the *recipient* will most likely not be capable of wrapping the error in an error envelope anyway. The HTTP 500 is not intended to be raised deliberately by the application that runs OCMP. The same goes for HTTP status code **404 Not Found**.

Delivering an error evelope in error situations is not mandatory, but it is highly recommended to do so. Proper debugging information will help troubleshooting in development and deploy situations.

Definition

Field Name	Field Type	Card.	Description
referenceId	string	1..1	The id of an earlier message that is referred to.
message	string	1..1	A human readable message explaining the cause of this message.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/error-envelope.json",
  "title": "ErrorEnvelope",
  "type": "object",
  "properties": {
    "referenceId": {
      "type": "string"
    },
    "message": {
      "type": "string"
    }
  },
  "required": [
    "referenceId",
    "message"
  ]
}
```

3.2. UpdateGroupMaximumCapacityForecast

The UpdateGroupMaximumCapacityForecast notification contains forecast of the **maximum** capacity of a certain aggregated group for a period of time. This forecast can for example be created based on measurements from a trafo or household energy consumption statistics at a certain moment in time. This message is sent from the DSO to the CSO and from CSO to CMO which should generate an **optimal** forecast value for the capacity that can be used in this specific group. This optimal forecast should be followed by CSO as a maximum value for the capacity that should be used in the group. The message is based on the principle of time division, so the message contains blocks.

Endpoints

Endpoint	/ocmp/cso/1.0/update_group_maximum_capacity_forecast

Type	HTTP request
Method	POST
Direction	DSO → CSO

Endpoint	/ocmp/cmo/1.0/update_group_maximum_capacity_forecast
Type	HTTP request
Method	POST
Direction	CSO → CMO

Definition

Field Name	Field Type	Card.	Description
groupId	string	1..1	The id of the area in which the CSO has connections for EV.
forecastedBlocks	ForecastedBlock	1..*	The technical content of this message.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/update-group-maximum-capacity-
forecast.json",
  "title": "UpdateGroupMaximumCapacityForecast",
  "type": "object",
  "properties": {
    "groupId": {
      "type": "string"
    },
    "forecastedBlocks": {
      "type": "array",
      "items": { "$ref": "forecasted-block.json" },
      "minItems": 1
    }
  },
  "required": [
    "groupId",
    "forecastedBlocks"
  ]
}
```

3.3. UpdateGroupOptimalCapacityForecast

The UpdateGroupOptimalCapacityForecast notification is a more sophisticated capacity forecast that is based on the UpdateGroupMaximumCapacityForecast message enriched with for instance weather forecasts (for PV), type of energy etc. This message is sent from CMO to the CSO which should follow this capacity as a maximum value for the capacity that can be used in this specific group. The message is based on the principle of time division, so the message contains blocks.

Endpoints

Endpoint	/ocmp/cso/1.0/update_group_optimal_capacity_forecast
Type	HTTP request
Method	POST
Direction	CMO → CSO

Definition

Field Name	Field Type	Card.	Description

groupId	string	1..1	The id of the area in which the CSO has connections for EV.
forecastedBlocks	ForecastedBlock	1..*	The technical content of this message.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/update-group-optimal-capacity-
forecast.json",
  "title": "UpdateGroupOptimalCapacityForecast",
  "type": "object",
  "properties": {
    "groupId": {
      "type": "string"
    },
    "forecastedBlocks": {
      "type": "array",
      "items": { "$ref": "forecasted-block.json" },
      "minItems": 1
    }
  },
  "required": [
    "groupId",
    "forecastedBlocks"
  ]
}
```

3.4. UpdateAccumulatedMeteringData

This notification is for communicating the total usage per CSO back to the DSO. This information is necessary for the DSO to know how much each CSO has used as input for checking whether parties have not used too much / little. Furthermore it can be used to determine a division of the forecast over the different parties.

The total usage can be 'nothing'. Therefore metering data (measurements) can be empty.

Endpoints

Endpoint	/ocmp/dso/1.0/update_accumulated_meteringdata
Type	HTTP request
Method	POST
Direction	CSO → DSO

Definition

Field Name	Field Type	Card.	Description
groupId	string	1..1	The id of the area containing the connection(s) of which the CSO gives the consumed capacity.
measurements	AccumulatedMeasurement	0..*	The technical content of this message.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/update-accumulated-metering-
data.json",
  "title": "UpdateAccumulatedMeteringData",
  "type": "object",
  "properties": {
    "groupId": {
      "type": "string"
    },
    "measurements": {
      "type": "array",
      "items": { "$ref": "accumulated-measurement.json" }
    }
  },
  "required": [
    "groupId",
    "measurements"
  ]
}
```

3.5. UpdateDetailedMeteringData

This notification contains various types of meter values and is sent by the CSO to the CMO. The CMO can use this information for composing an optimized profile (which in turn is sent back within an UpdateGroupOptimalCapacityForecast message).

Metering data (measurements) is mandatory because a notification without any makes no sense. When CSO has no available metering data, for instance no running sessions are available, this notification will not be sent to the CMO.

For each asset multiple measurements can be made, for instance: the measuring of the assets net and import/export register or the stop of an EV session and the start of another one.

Endpoints

Endpoint	/ocmp/cmo/1.0/update_detailed_meteringdata
Type	HTTP request
Method	POST
Direction	CSO → CMO

Definition

Field Name	Field Type	Card.	Description
groupId	string	1..1	The id of the area containing the connection(s) of which the CSO gives the consumed capacity.
measurements	DetailedMeasurement	1..*	This field represents the measured meter values.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/update-detailed-metering-data.json",
  "title": "UpdateDetailedMeteringData",
  "type": "object",
  "properties": {
    "groupId": {
      "type": "string"
    },
    "measurements": {
      "type": "array",
      "items": { "$ref": "detailed-measurement.json" },
      "minItems": 1
    }
  },
  "required": [
    "groupId",
    "measurements"
  ]
}
```

3.6. Handshaking

The purpose of this message is to exchange settings between DSO-CSO and CSO-CMO. A handshake consists of a request and a response and *must* be sent upon startup. Handshaking *may* be initiated at any time.

3.6.1. Handshake

A handshake request is the message that initiates the handshaking mechanism. It is followed by a HandshakeAcknowledge. For OCMP message protocol specific timing limitations refer to Message timeouts.

Endpoints

Endpoint	/ocmp/cso/1.0/handshake
Type	HTTP request
Method	POST
Direction	DSO → CSO

Endpoint	/ocmp/cmo/1.0/handshake
Type	HTTP request
Method	POST
Direction	CSO → CMO

Definition

Field Name	Field Type	Card.	Description
offeredBehaviour	OfferedBehaviour	1..1	Contains configurations of the sender of this message.
requiredBehaviour	RequiredBehaviour	0..1	Optional. Contains configurations that define slight adjustments to the behaviour of the other party. The requiredBehaviour defines how the *other party* should behave. If none is given, the other party is free to implement its own behaviour.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/handshake.json",
  "title": "Handshake",
  "type": "object",
  "properties": {
    "offeredBehaviour": { "$ref": "offered-behaviour.json" },
    "requiredBehaviour": { "$ref": "required-behaviour.json" }
  },
  "required": [
    "offeredBehaviour"
  ]
}
```

3.6.2. HandshakeAcknowledge

A handshake acknowledge is the message that is sent in response to an incoming Handshake message. This request contains a *baseUrl* that should from now on be used, refer to OfferedBehaviour.

Endpoints

Endpoint	/ocmp/dso/1.0/handshake_acknowledge
Type	HTTP request
Direction	CSO → DSO

Endpoint	/ocmp/cso/1.0/handshake_acknowledge
Type	HTTP request
Direction	CMO → CSO

Definition

Field Name	Field Type	Card.	Description
offeredBehaviour	OfferedBehaviour	1..1	Contains configurations of the sender of this message.
requiredBehaviour	RequiredBehaviour	0..1	Optional. Contains configurations that define slight adjustments to the behaviour of the other party. The requiredBehaviour defines how the *other party* should behave. If none is given, the other party is free to implement its own behaviour.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/handshake-acknowledge.json",
  "title": "HandshakeAcknowledge",
  "type": "object",
  "properties": {
    "offeredBehaviour": { "$ref": "offered-behaviour.json" },
    "requiredBehaviour": { "$ref": "required-behaviour.json" }
  },
  "required": [
    "offeredBehaviour"
  ]
}
```

3.7. Heartbeat

The purpose of the OCMP heartbeat is to periodically notify the senders' availability to the hand-shaken party. The interval in which a heartbeat is sent should be determined using the Handshaking mechanism. Therefore sending a heatbeat is only permitted **after** the handshake is completed.

Endpoints

Endpoint	/ocmp/dso/1.0/heartbeat
Type	HTTP request
Method	POST
Direction	CSO → DSO

Endpoint	/ocmp/cso/1.0/heartbeat
Type	HTTP request
Method	POST
Direction	DSO → CSO, CMO → CSO

Endpoint	/ocmp/cmo/1.0/heartbeat
Type	HTTP request
Method	POST
Direction	CSO → CMO

Definition

Field Name	Field Type	Card.	Description
currentTime	DateTime	1..1	The current system time of the system in UTC.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/heartbeat.json",
  "title": "Heartbeat",
  "type": "object",
  "properties": {
    "currentTime": {
      "type": "string",
      "format": "date-time"
    }
  },
  "required": [
    "currentTime"
  ]
}
```

4. Datatypes

4.1. ForecastedBlock

A ForecastedBlock represents a timeslot with a specific maximum capacity for a group that is forecasted by the DSO. The capacity per group is divided over the parties (can be the same CSO) that have connections in that specific group.

Definition

Field Name	Field Type	Card.	Description
capacity	number	1..1	The value is the maximum forecast.
phase	PhaseIndicator	1..1	This field identifies the phase that the forecast is meant for.
unit	ForecastedBlockUnit	1..1	Unit of the capacity.
startTime	DateTime	1..1	Start time of this block
endTime	DateTime	1..1	End time of this block

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/forecasted-block.json",
  "title": "ForecastedBlock",
  "type": "object",
  "properties": {
    "capacity": {
      "type": "number",
      "minimum": 0
    },
    "phase": { "$ref": "phase-indicator.json" },
    "unit": { "$ref": "forecasted-block-unit.json" },
    "startTime": {
      "type": "string",
      "format": "date-time"
    },
    "endTime": {
      "type": "string",
      "format": "date-time"
    }
  },
  "required": [
    "capacity",
    "phase",
    "unit",
    "startTime",
    "endTime"
  ]
}
```

4.2. InstantaneousMeasurement

A measurement that holds the current value that flows through the meter.

Definition

Field Name	Field Type	Card.	Description
value	number	1..1	The actual measured value.

phase	PhaseIndicator	1..1	This field identifies the phase that is measured.
unit	InstantaneousMeasurementUnit	1..1	Unit of the value.
measureTimestamp	DateTime	1..1	Date and time of the actual meter-reading.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/instantaneous-measurement.json",
  "title": "InstantaneousMeasurement",
  "type": "object",
  "properties": {
    "value": {
      "type": "number"
    },
    "phase": { "$ref": "phase-indicator.json" },
    "unit": { "$ref": "instantaneous-measurement-unit.json" },
    "measureTimestamp": {
      "type": "string",
      "format": "date-time"
    }
  },
  "required": [
    "value",
    "phase",
    "unit",
    "measureTimestamp"
  ]
}
```

4.3. AccumulatedMeasurement

A measurement that is by itself accumulated. An object of this type is synchronous to a meter reading.

Definition

Field Name	Field Type	Card.	Description
value	number	1..1	The actual measured value.
phase	PhaseIndicator	1..1	This field identifies the phase that is measured.
unit	AccumulatedMeasurementUnit	1..1	Unit of the value.
energyFlow	EnergyFlowDirection	1..1	The energy flow direction. Indicates energy active net, import or export register
measureTimestamp	DateTime	1..1	Date and time of the actual current meter reading.
measureInitialMeterTimestamp	DateTime	0..1	Optional. Date and time of the moment the measurement has (re)started (i.e. the moment an EV charge session starts). If the other party defined a RequiredBehaviour with *INTERMITTENT* as part of the *accumulatedMeasurementConfigurations* field, then this field *must* be set.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/accumulated-measurement.json",
  "title": "AccumulatedMeasurement",
  "type": "object",
  "properties": {
    "value": {
      "type": "number"
    },
    "phase": { "$ref": "phase-indicator.json" },
    "unit": { "$ref": "accumulated-measurement-unit.json" },
    "energyFlow": { "$ref": "energy-flow-direction.json" },
    "measureTimestamp": {
      "type": "string",
      "format": "date-time"
    },
    "measureInitialMeterTimestamp": {
      "type": "string",
      "format": "date-time"
    }
  },
  "required": [
    "value",
    "phase",
    "unit",
    "energyFlow",
    "measureTimestamp"
  ]
}
```

4.4. DetailedMeasurement

An assembly of measurements and their details.

Definition

Field Name	Field Type	Card.	Description
assetId	string	1..1	Uniquely identifies the device.
assetCategory	AssetCategory	1..1	Defines the asset/unit that is metered.
instantaneousMeasurement	InstantaneousMeasurement	0..1	Optional. Represents an instantaneous measuring.
accumulatedMeasurement	AccumulatedMeasurement	0..1	Optional. Represents a read out of an accumulative energy meter.

Either **instantaneousMeasurement** or **accumulatedMeasurement** should be provided.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/detailed-measurement.json",
  "title": "DetailedMeasurement",
  "type": "object",
  "properties": {
    "assetId": {
      "type": "string"
    },
    "assetCategory": { "$ref": "asset-category.json" },
    "instantaneousMeasurement": { "$ref": "instantaneous-measurement.json" },
    "accumulatedMeasurement": { "$ref": "accumulated-measurement.json" }
  },
  "oneOf": [
    {
      "required": [
        "assetId",
        "assetCategory",
        "instantaneousMeasurement"
      ]
    }, {
      "required": [
        "assetId",
        "assetCategory",
        "accumulatedMeasurement"
      ]
    }
  ]
}
```

4.5. OfferedBehaviour

The *behaviour of the sender* that is notified to the other party.

Definition

Field Name	Field Type	Card.	Description
baseUrl	string	1..1	Contains the address of the REST server of the sender. This address must be used to send OCMP messages to. The message specific endpoints are to be concatenated to this address. This baseUrl must start with a protocol and protocol delimiter, i.e. *https://*. It must *not* end with a slash (/).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/offered-behaviour.json",
  "title": "OfferedBehaviour",
  "type": "object",
  "properties": {
    "baseUrl": {
      "type": "string",
      "format": "uri",
      "pattern": "^.*?(?<!/)$"
    }
  },
  "required": [
    "baseUrl"
  ]
}
```

4.6. RequiredBehaviour

The *behaviour of the other party* that is required for the sender to function properly.

Definition

Field Name	Field Type	Card.	Description
heartbeatInterval	number	0..1	Optional. The interval (in seconds) in which the sender of this response expects heartbeats to receive. If the sender is not interested in the heartbeat of the receiver, this field can be omitted.
accumulatedMeasurementConfigurations	AccumulatedMeasurementConfiguration	0..*	For determining how measurements are accumulated. Providing multiple configurations is allowed. An empty array represents no configurations.

At least *one* of the above fields must be given; empty RequiredBehaviour objects are *not* allowed.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/required-behaviour.json",
  "title": "RequiredBehaviour",
  "type": "object",
  "properties": {
    "heartbeatInterval": {
      "type": "number",
      "minimum": 0,
      "exclusiveMinimum": true
    },
    "accumulatedMeasurementConfigurations": {
      "type": "array",
      "uniqueItems": true,
      "items": { "$ref": "accumulated-measurement-configuration.json" }
    }
  },
  "required": [
    "accumulatedMeasurementConfigurations"
  ]
}
```

4.7. AccumulatedMeasurementConfiguration

A unit for *accumulatedMeasurementConfigurations* of RequiredBehaviour.

Definition *Enumeration*

CONTINUOUS	The measurement is accumulated without breaks and the accumulation is never reset. It is assumed that the asset in question keeps on accumulating its meter value from the moment it was installed, similarly to a home/office situation.
INTERMITTENT	The accumulation of the measurement is reset at certain points in time (i.e. when a charging session starts).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/accumulated-measurement-
configuration.json",
  "title": "AccumulatedMeasurementConfiguration",
  "type": "string",
  "enum": ["CONTINUOUS", "INTERMITTENT"]
}
```

4.8. ForecastedBlockUnit

A unit for ForecastedBlock.

Definition *Enumeration*

A	Ampère (current).
W	Watt (power).
KW	Kilowatt (power).
WH	Watt-hours (energy).

KWH	Kilowatt-hours (energy).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/forecasted-block-unit.json",
  "title": "ForecastedBlockUnit",
  "type": "string",
  "enum": ["A", "W", "KW", "WH", "KWH"]
}
```

4.9. InstantaneousMeasurementUnit

A unit for InstantaneousMeasurement.

Definition *Enumeration*

A	Ampère per phase (current).
W	Watt (power).
KW	Kilowatt (power).
WH	Watt-hours (energy). In case of instantaneous measuring State Of Charge (battery).
KWH	Kilowatt-hours (energy). In case of instantaneous measuring State Of Charge (battery).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/instantaneous-measurement-unit.json",
  "title": "InstantaneousMeasurementUnit",
  "type": "string",
  "enum": ["A", "W", "KW", "WH", "KWH"]
}
```

4.10. AccumulatedMeasurementUnit

A unit for AccumulatedMeasurement.

Definition *Enumeration*

WH	Watt-hours (energy).
KWH	Kilowatt-hours (energy).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/accumulated-measurement-unit.json",
  "title": "AccumulatedMeasurementUnit",
  "type": "string",
  "enum": ["WH", "KWH"]
}
```

4.11. PhaseIndicator

An indicator of the phases on which the measurement was done. It is an indicator for various measurements.

Definition *Enumeration*

UNKNOWN	Phase on which is measured is not known or irrelevant.
ONE	Represents measurement on phase 1.
TWO	Represents measurement on phase 2.
THREE	Represents measurement on phase 3.
ALL	Measurement represents the sum of all phases (1, 2, and 3).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/phase-indicator.json",
  "title": "PhaseIndicator",
  "type": "string",
  "enum": ["UNKNOWN", "ONE", "TWO", "THREE", "ALL"]
}
```

4.12. AssetCategory

A category of assets that are supported by OCMP.

Depending on the category one or more energy flows can be measured.

Definition *Enumeration*

CHARGING	All EV supply equipment. Could be bi-directional (V2G).
CONSUMPTION	Consumption unit with all loads in the group other than charging and storage units.
GENERATION	All one-way energy generation units. Examples are PV, wind turbines, generators.
STORAGE	All stationary two-way energy storage units (batteries).

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/asset-category.json",
  "title": "AssetCategory",
  "type": "string",
  "enum": ["CHARGING", "CONSUMPTION", "GENERATION", "STORAGE"]
}
```

4.13. EnergyFlowDirection

A definition of the energy flow direction. Flows are always positive.

An energy active **import** register represents the amound of energy that has been put into the asset. The corresponding asset category in this case will be consumption. An energy active **export** shows how much energy is extracted from a generation asset. The energy active **net** is effectively the difference between import and export.

Depending on the type of asset one or more registers are available. Though net is mandatory.

Definition *Enumeration*

NET	Indicates an energy active net register.
IMPORT	Indicates an energy active import register.
EXPORT	Indicates an energy active export register.

Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://www.evnet.nl/schemas/ocmp/energy-flow-direction.json",
  "title": "EnergyFlowDirection",
  "type": "string",
  "enum": ["NET", "IMPORT", "EXPORT"]
}
```

5. Appendix

5.1. Appendix A: Examples

Some code examples with their flow are provided here.

5.1.1. Handshaking

CSO initiates handshaking with CMO

The CSO starts handshaking by sending a Handshake message. According to the OCMP spec, it is wrapped inside an envelope. In the following example, the CSO wants to receive heartbeats with a rate of 15 seconds. No other required behaviour is defined, so the CSO is neutral on any other configurations.

Send the following message …

```
{
  "id": "msg#998",
  "data": {
    "offeredBehaviour": {
      "baseUrl": "https://evnet.nl:8080"
    },
    "requiredBehaviour": {
      "heartbeatInterval": 15
    }
  }
}
```

to the API …

```
POST: https://www.esmartsystems.com/ocmp/cmo/1.0/handshake
```

and wait for response.

```
HTTP status response: 200 OK
```

CMO responds handshaking with CSO

The CMO receives the Handshake message and sends a HandshakeAcknowledge to the OCMP endpoint with the base url of that Handshake message. In the following example the CMO requires the AccumulatedMeasurement to be configured as INTERMITTENT. Additionally the CMO is also interested in the assets' continuously increasing counter values. It therefore also enables CONTINUOUS AccumulatedMeasurement.

No other required behaviour is defined, so the CMO is neutral on any other configurations. The CMO uses the "topic" attribute to refer to the Handshake message that was sent by the CSO. This is to indicate that this HandshakeAcknowledge belongs to that Handshake.

Send the following message …

```
{
  "id": "id_703",
  "topic": "msg#998",
  "data": {
    "offeredBehaviour": {
      "baseUrl": "https://www.esmartsystems.com"
    },
    "requiredBehaviour": {
      "accumulatedMeasurementConfigurations": ["CONTINUOUS", "INTERMITTENT"]
    }
  }
}
```

to the API …

```
POST: https://evnet.nl:8080/ocmp/cso/1.0/handshake_acknowledge
```

and wait for response.

```
HTTP status response: 200 OK
```

5.1.2. Energy flows

Let's say there's a battery/storage that has the following registers (in kWh); import = 4000, export = 3950, net = 50.

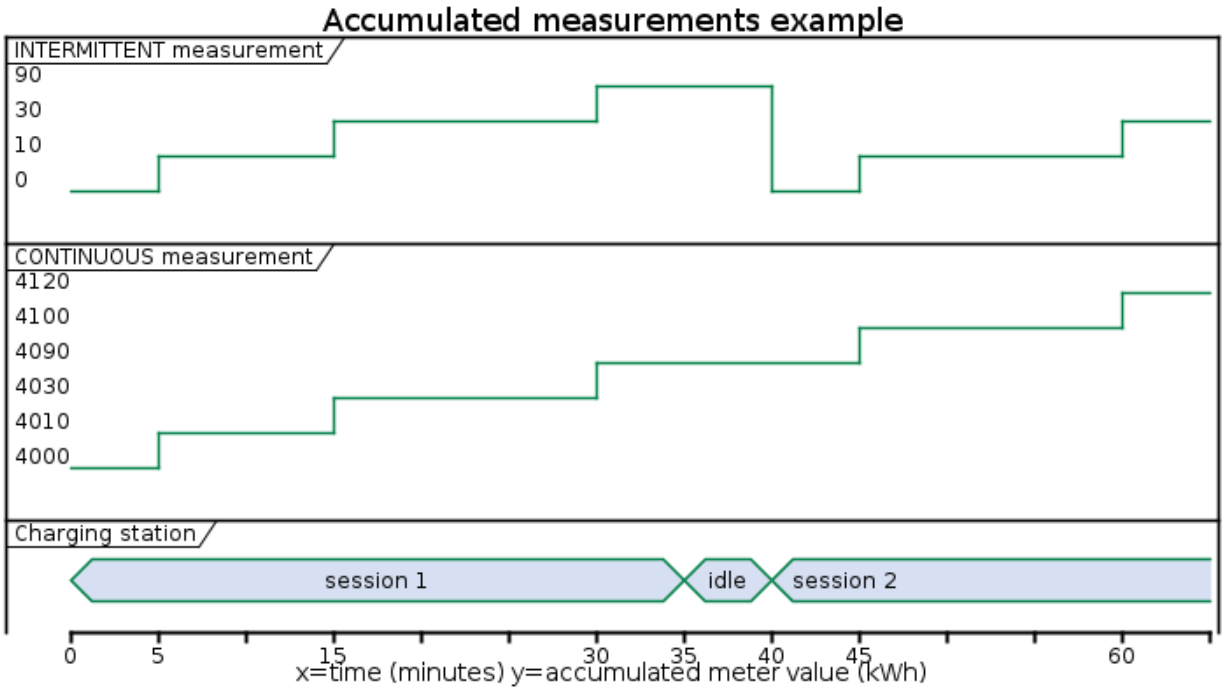
At a certain moment in time 35kWh is put into this battery. The registers should be as following; import = 4035, export = 3950, net = 85.

Then 10kWh is extracted and the registers should be as following; import = 4035, export = 3960, net = 75.

In some cases, depending on the quality of the meters, the net can also represent the State of Charge.

5.1.3. Measuring EV sessions

Below is given a diagram that illustrates the use of accumulated measurements during an EV charging session. Each accumulated metervalue (y axis) represents an AccumulatedMeasurement instance.



5.2. Appendix B: Single JSON schema

The following JSON schema is a concatenation of all JSON schemas in the OCMP document. It can be used to validate JSON messages or to generate sourcecode from it.

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "$id": "root.json",
    "title": "Root",
    "type": "object",
    "description" : "Dummy class which contains all messages and datatypes.",
    "properties": {
        "Envelope": {
            "$ref": "envelope.json"
        },
        "ErrorEnvelope": {
            "$ref": "error-envelope.json"
        },
        "UpdateGroupMaximumCapacityForecast": {
            "$ref": "update-group-maximum-capacity-forecast.json"
        },
        "UpdateGroupOptimalCapacityForecast": {
            "$ref": "update-group-optimal-capacity-forecast.json"
        },
        "UpdateAccumulatedMeteringData": {
            "$ref": "update-accumulated-metering-data.json"
        },
        "UpdateDetailedMeteringData": {
            "$ref": "update-detailed-metering-data.json"
        },
        "Handshake": {
            "$ref": "handshake.json"
        },
        "HandshakeAcknowledge": {
            "$ref": "handshake-acknowledge.json"
        },
```

```
        "Heartbeat": {
            "$ref": "heartbeat.json"
        },
        "ForecastedBlock": {
            "$ref": "forecasted-block.json"
        },
        "InstantaneousMeasurement": {
            "$ref": "instantaneous-measurement.json"
        },
        "AccumulatedMeasurement": {
            "$ref": "accumulated-measurement.json"
        },
        "DetailedMeasurement": {
            "$ref": "detailed-measurement.json"
        },
        "OfferedBehaviour": {
            "$ref": "offered-behaviour.json"
        },
        "RequiredBehaviour": {
            "$ref": "required-behaviour.json"
        },
        "AccumulatedMeasurementConfiguration": {
            "$ref": "accumulated-measurement-configuration.json"
        },
        "ForecastedBlockUnit": {
            "$ref": "forecasted-block-unit.json"
        },
        "InstantaneousMeasurementUnit": {
            "$ref": "instantaneous-measurement-unit.json"
        },
        "AccumulatedMeasurementUnit": {
            "$ref": "accumulated-measurement-unit.json"
        },
        "PhaseIndicator": {
            "$ref": "phase-indicator.json"
        },
        "AssetCategory": {
            "$ref": "asset-category.json"
        },
        "EnergyFlowDirection": {
            "$ref": "energy-flow-direction.json"
        }
    },

    "definitions": {
        "Envelope": {
            "$id": "envelope.json",
            "title": "Envelope",
            "description" : "All HTTP request messages are wrapped in a request
envelope as described below.",
            "type": "object",
            "properties": {
                "id": {
                    "type": "string",
                    "description" : "A unique identifier of this message."
                },
                "data": {
                    "type": "object",
```

```
                    "description" : "The wrapped content of this message."
                },
                "referenceId": {
                    "type": "string",
                    "description" : "Optional. The id of an earlier message that
is referred to."
                },
                "priority": {
                    "type": "integer",
                    "description" : "Optional. This field determines the
importance of this message, where values < 0 are less important and values > 0
are more important. Omitting this field is effectively equal to a priority of
0."
                }
            },
            "required": [
                "id",
                "data"
            ]
        },

        "ErrorEnvelope": {
            "$id": "error-envelope.json",
            "title": "ErrorEnvelope",
            "type": "object",
            "properties": {
                "referenceId": {
                    "type": "string",
                    "description" : "The id of an earlier message that is
referred to."
                },
                "message": {
                    "type": "string",
                    "description" : "A human readable message explaining the
cause of this message."
                },
                "data": {
                    "type": "object",
                    "description" : "Optional. This field may contain details
about the error."
                }
            },
            "required": [
                "referenceId",
                "message"
            ]
        },

        "UpdateGroupMaximumCapacityForecast": {
            "$id": "update-group-maximum-capacity-forecast.json",
            "title": "UpdateGroupMaximumCapacityForecast",
            "type": "object",
            "description" : "A rolling forecast of the maximum capacity of a
certain aggregated group at a certain moment in time.",
            "properties": {
                "groupId": {
                    "type": "string",
                    "description" : "The id of the area in which the CSO has
```

```
connections for EV."
                },
                "forecastedBlocks": {
                    "type": "array",
                    "description" : "The technical content of this message.",
                    "items": {
                        "$ref": "forecasted-block.json"
                    },
                    "minItems": 1
                }
            },
            "required": [
                "groupId",
                "forecastedBlocks"
            ]
        },

        "UpdateGroupOptimalCapacityForecast": {
            "$id": "update-group-optimal-capacity-forecast.json",
            "title": "UpdateGroupOptimalCapacityForecast",
            "type": "object",
            "description" : "A more sophisticated capacity forecast that is
enriched with for instance weather forecasts (for PV), type of energy etc.",
            "properties": {
                "groupId": {
                    "type": "string",
                    "description" : "The id of the area in which the CSO has
connections for EV."
                },
                "forecastedBlocks": {
                    "type": "array",
                    "description" : "The technical content of this message.",
                    "items": {
                        "$ref": "forecasted-block.json"
                    },
                    "minItems": 1
                }
            },
            "required": [
                "groupId",
                "forecastedBlocks"
            ]
        },

        "UpdateAccumulatedMeteringData": {
            "$id": "update-accumulated-metering-data.json",
            "title": "UpdateAccumulatedMeteringData",
            "type": "object",
            "description" : "For communicating the total usage per CSO back to
the DSO.",
            "properties": {
                "groupId": {
                    "type": "string",
                    "description" : "The id of the area containing the
connection(s) of which the CSO gives the consumed capacity."
                },
                "measurements": {
                    "type": "array",
```

```
                    "description" : "The technical content of this message.",
                    "items": {
                        "$ref": "accumulated-measurement.json"
                    }
                }
            },
            "required": [
                "groupId",
                "measurements"
            ]
        },

        "UpdateDetailedMeteringData": {
            "$id": "update-detailed-metering-data.json",
            "title": "UpdateDetailedMeteringData",
            "type": "object",
            "description" : "Contains various types of meter values and is sent
by the CSO to the CMO.",
            "properties": {
                "groupId": {
                    "type": "string",
                    "description" : "The id of the area containing the
connection(s) of which the CSO gives the consumed capacity."
                },
                "measurements": {
                    "type": "array",
                    "description" : "This field represents the measured meter
values.",
                    "items": {
                        "$ref": "detailed-measurement.json"
                    },
                    "minItems": 1
                }
            },
            "required": [
                "groupId",
                "measurements"
            ]
        },

        "Handshake": {
            "$id": "handshake.json",
            "title": "Handshake",
            "type": "object",
            "description" : "The message that initiates the handshaking
mechanism",
            "properties": {
                "offeredBehaviour": {
                    "$ref": "offered-behaviour.json",
                    "description" : "Contains configurations of the sender of
this message."
                },
                "requiredBehaviour": {
                    "$ref": "required-behaviour.json",
                    "description" : "Optional. Contains configurations that
define slight adjustments to the behaviour of the other party. The
requiredBehaviour defines how the other party should behave. If none is given,
the other party is free to implement its own behaviour."
```

```
                }
            },
            "required": [
                "offeredBehaviour"
            ]
        },

        "HandshakeAcknowledge": {
            "$id": "handshake-acknowledge.json",
            "title": "HandshakeAcknowledge",
            "type": "object",
            "description" : "The message that is sent in response to an incoming
Handshake message.",
            "properties": {
                "offeredBehaviour": {
                    "$ref": "offered-behaviour.json",
                    "description" : "Contains configurations of the sender of
this message."
                },
                "requiredBehaviour": {
                    "$ref": "required-behaviour.json",
                    "description" : "Optional. Contains configurations that
define slight adjustments to the behaviour of the other party. The
requiredBehaviour defines how the other party should behave. If none is given,
the other party is free to implement its own behaviour."
                }
            },
            "required": [
                "offeredBehaviour"
            ]
        },

        "Heartbeat": {
            "$id": "heartbeat.json",
            "title": "Heartbeat",
            "type": "object",
            "description" : "To periodically notify the senders' availability to
the hand-shaken party",
            "properties": {
                "currentTime": {
                    "type": "string",
                    "description" : "The current system time of the system in
UTC.",
                    "format": "date-time"
                }
            },
            "required": [
                "currentTime"
            ]
        },

        "ForecastedBlock": {
            "$id": "forecasted-block.json",
            "title": "ForecastedBlock",
            "type": "object",
            "description" : "A timeslot with a specific maximum capacity for a
group that is forecasted by the DSO.",
            "properties": {
```

```
                "capacity": {
                    "type": "number",
                    "description" : "The value is the maximum forecast.",
                    "minimum": 0
                },
                "phase": {
                    "$ref": "phase-indicator.json",
                    "description": "This field identifies the phase that the
forecast is meant for."
                },
                "unit": {
                    "$ref": "forecasted-block-unit.json",
                    "description" : "Unit of the capacity."
                },
                "startTime": {
                    "type": "string",
                    "description" : "Start time of this block.",
                    "format": "date-time"
                },
                "endTime": {
                    "type": "string",
                    "description" : "End time of this block.",
                    "format": "date-time"
                }
            },
            "required": [
                "capacity",
                "phase",
                "unit",
                "startTime",
                "endTime"
            ]
        },

        "InstantaneousMeasurement": {
            "$id": "instantaneous-measurement.json",
            "title": "InstantaneousMeasurement",
            "type": "object",
            "description" : "A measurement that holds the current value that
flows through the meter.",
            "properties": {
                "value": {
                    "type": "number",
                    "description" : "The actual measured value."
                },
                "phase": {
                    "$ref": "phase-indicator.json",
                    "description" : "This field identifies the phase that is
measured."
                },
                "unit": {
                    "$ref": "instantaneous-measurement-unit.json",
                    "description" : "Unit of the value."
                },
                "measureTimestamp": {
                    "type": "string",
                    "description" : "Date and time of the actual meter-
reading.",
```

```
                    "format": "date-time"
                }
            },
            "required": [
                "value",
                "phase",
                "unit",
                "measureTimestamp"
            ]
        },

        "AccumulatedMeasurement": {
            "$id": "accumulated-measurement.json",
            "title": "AccumulatedMeasurement",
            "type": "object",
            "properties": {
                "value": {
                    "type": "number",
                    "description" : "The actual measured value."
                },
                "phase": {
                    "$ref": "phase-indicator.json",
                    "description" : "This field identifies the phase that is
measured."
                },
                "unit": {
                    "$ref": "accumulated-measurement-unit.json",
                    "description" : "Unit of the value."
                },
                "energyFlow": {
                    "$ref": "energy-flow-direction.json",
                    "description" : "The energy flow direction. Indicates energy
active net, import or export register."
                },
                "measureTimestamp": {
                    "type": "string",
                    "description" : "Date and time of the actual current meter
reading.",
                    "format": "date-time"
                },
                "measureInitialMeterTimestamp": {
                    "type": "string",
                    "description" : "Optional. Date and time of the moment the
measurement has (re)started (i.e. the moment an EV charge session starts). If
the other party defined a RequiredBehaviour with INTERMITTENT as part of the
accumulatedMeasurementConfigurations field, then this field must be set.",
                    "format": "date-time"
                }
            },
            "required": [
                "value",
                "phase",
                "unit",
                "energyFlow",
                "measureTimestamp"
            ]
        },
```

```
        "DetailedMeasurement": {
            "$id": "detailed-measurement.json",
            "title": "DetailedMeasurement",
            "type": "object",
            "description" : "An assembly of measurements and their details.",
            "properties": {
                "assetId": {
                    "type": "string",
                    "description" : "Uniquely identifies the device."
                },
                "assetCategory": {
                    "$ref": "asset-category.json",
                    "description" : "Defines the asset/unit that is metered."
                },
                "instantaneousMeasurement": {
                    "$ref": "instantaneous-measurement.json",
                    "description" : "Optional. Represents an instantaneous
measuring."
                },
                "accumulatedMeasurement": {
                    "$ref": "accumulated-measurement.json",
                    "description" : "Optional. Represents a read out of an
accumulative energy meter."
                }
            },
            "oneOf": [{
                "required": [
                    "assetId",
                    "assetCategory",
                    "instantaneousMeasurement"
                ]
            }, {
                "required": [
                    "assetId",
                    "assetCategory",
                    "accumulatedMeasurement"
                ]
            }]
        },

        "OfferedBehaviour": {
            "$id": "offered-behaviour.json",
            "title": "OfferedBehaviour",
            "type": "object",
            "description" : "The behaviour of the sender that is notified to the
other party.",
            "properties": {
                "baseUrl": {
                    "type": "string",
                    "description" : "Contains the address of the REST server of
the sender. This address must be used to send OCMP messages to. The message
specific endpoints are to be concatenated to this address. This baseUrl must
start with a protocol and protocol delimiter, i.e. https://. It must not end
with a slash (/).",
                    "format": "uri",
                    "pattern": "^.*?(?<!/)$"
                }
            },
```

```
            "required": [
                "baseUrl"
            ]
        },

        "RequiredBehaviour": {
            "$id": "required-behaviour.json",
            "title": "RequiredBehaviour",
            "type": "object",
            "description" : "The behaviour of the other party that is required
for the sender to function properly.",
            "properties": {
                "heartbeatInterval": {
                    "type": "number",
                    "description" : "Optional. The interval (in seconds) in
which the sender of this response expects heartbeats to receive. If the sender
is not interested in the heartbeat of the receiver, this field can be omitted.",
                    "minimum": 0,
                    "exclusiveMinimum": true
                },
                "accumulatedMeasurementConfigurations": {
                    "type": "array",
                    "description" : "For determining how measurements are
accumulated. Providing multiple configurations is allowed. An empty array
represents no configurations.",
                    "uniqueItems": true,
                    "items": {
                        "$ref": "accumulated-measurement-configuration.json"
                    }
                }
            },
            "required": [
                "accumulatedMeasurementConfigurations"
            ]
        },

        "AccumulatedMeasurementConfiguration": {
            "$id": "accumulated-measurement-configuration.json",
            "title": "AccumulatedMeasurementConfiguration",
            "type": "string",
            "description" : "A unit for accumulatedMeasurementConfigurations of
RequiredBehaviour.",
            "enum": ["CONTINUOUS", "INTERMITTENT"]
        },

        "ForecastedBlockUnit": {
            "$id": "forecasted-block-unit.json",
            "title": "ForecastedBlockUnit",
            "type": "string",
            "description" : "A unit for ForecastedBlock.",
            "enum": ["A", "W", "KW", "WH", "KWH"]
        },

        "InstantaneousMeasurementUnit": {
            "$id": "instantaneous-measurement-unit.json",
            "title": "InstantaneousMeasurementUnit",
            "type": "string",
            "description" : "A unit for InstantaneousMeasurement.",
```

```
            "enum": ["A", "W", "KW", "WH", "KWH"]
        },

        "AccumulatedMeasurementUnit": {
            "$id": "accumulated-measurement-unit.json",
            "title": "AccumulatedMeasurementUnit",
            "type": "string",
            "description" : "A unit for AccumulatedMeasurement.",
            "enum": ["WH", "KWH"]
        },

        "PhaseIndicator": {
            "$id": "phase-indicator.json",
            "title": "PhaseIndicator",
            "type": "string",
            "description" : "An indicator of the phases on which the measurement
was done.",
            "enum": ["UNKNOWN", "ONE", "TWO", "THREE", "ALL"]
        },

        "AssetCategory": {
            "$id": "asset-category.json",
            "title": "AssetCategory",
            "type": "string",
            "description" : "A category of assets that are supported by OCMP.",
            "enum": ["CHARGING", "CONSUMPTION", "GENERATION", "STORAGE"]
        },

        "EnergyFlowDirection": {
            "$id": "energy-flow-direction.json",
            "title": "EnergyFlowDirection",
            "type": "string",
            "description" : "A definition of the energy flow direction. Flows
are always positive.",
            "enum": ["NET", "IMPORT", "EXPORT"]
        }
    }
}
```

5.3. Appendix C: Business rules

Invade pilot specific decisions on OCMP are described below.

5.3.1. General

Below is given some project-wide decisions.

1. For security, basic authentication over https will be used.
2. The groupId represents a 4 digit Dutch postal code, also known as PC4. This identifies a generally known area whose coordinates are available. In the future, a different identifier could be conceivable.
3. The unit in which forecasted blocks are provided is *Ampère*, since its the purest way of measuring overcapacity.

Error handling

Any additional error details can be put under the error envelope **data** field. The only contraint to this data is that it is a valid JSON formatted object.

5.3.2. DSO-CSO

Below is given some rules applying on the communication between DSO and CSO.

1. As long as CSO has not received any new forecast from the DSO it will assume the previous forecast is still valid.
2. DSO sends UpdateGroupMaximumCapacityForecast every 15 minutes.

5.3.3. CSO-CMO

The interface between CSO and CMO is described here.

Flow

Below is given a diagram that will illustrate the flow between CSO and CMO.

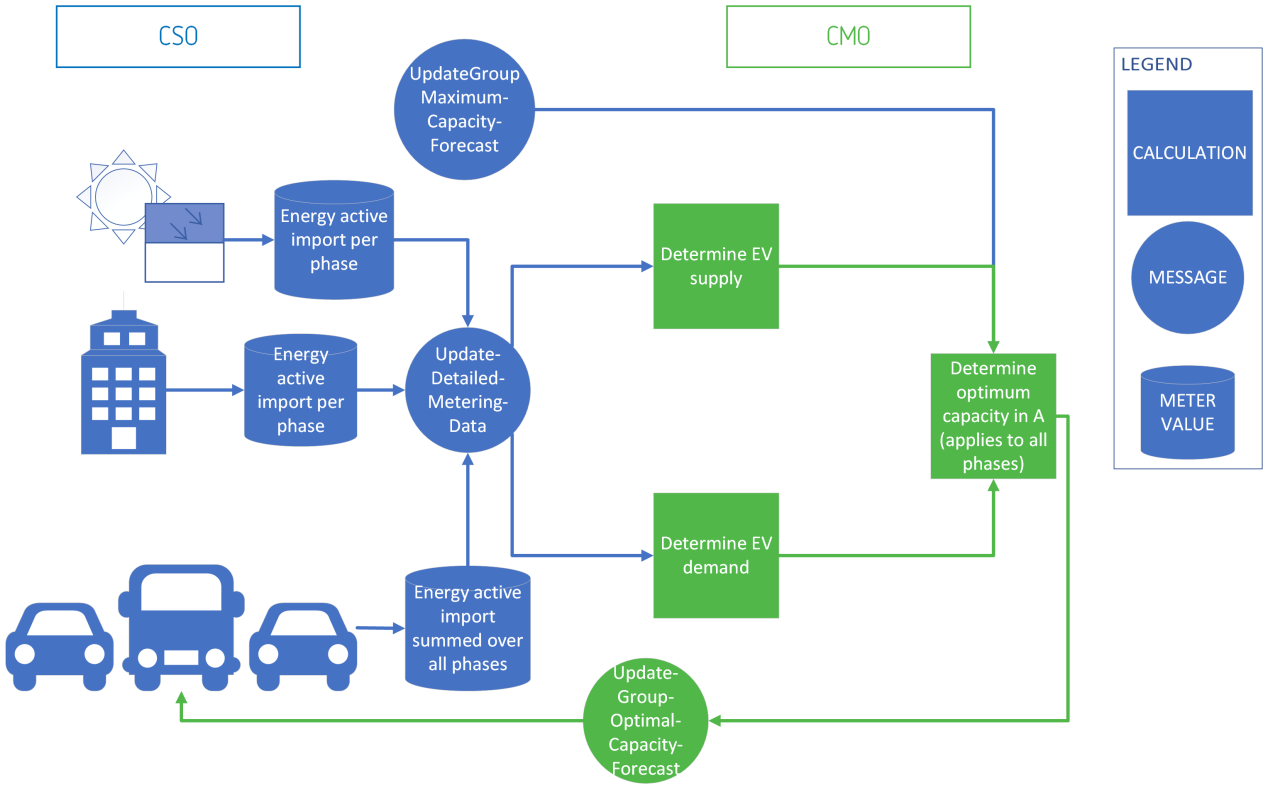


Figure 4. CSO-CMO flow

Rules

Below is given a number of rules applying on the interface between CSO and CMO.

1. CSO sends the UpdateGroupMaximumCapacityForecast message for a forecast period of 24 hours.
2. CSO sends UpdateDetailedMeteringData message every 15 minutes. The model contains asset measurements of the last 15 minutes. Multiple measurings per asset may exist in the model.
3. In case UpdateDetailedMeteringData message is not responded with HTTP status code 200 the CSOs measurements will be send as part of the next message.
4. A measurement of type EV charging represents either an active or a stopped session. A stopped session is notified to CMO via an UpdateDetailedMeteringData message only once.
5. It is not guaranteed that UpdateDetailedMeteringData contains details of all sessions because a Charging Station can be offline for a while. In this case CSO provides the last known measurement. In case no last measurement is known CSO will supply **no measurement**.
6. The field 'measureInitialMeterTimestamp' in AccumulatedMeasurement represents date/time of the moment the EV session has started. When a new session is started this field is updated. It can be interpreted as an EV session identifier. When a session is stopped the field is empty (*null*), and the field 'measureTimestamp' represents date/time of the moment the EV session has stopped.

7. After a stopped session the AccumulatedMeasurement is sent to the CMO as part of the next scheduled UpdateDetailedMeteringData message. After that the measurement will not be part of any upcoming UpdateDetailedMeteringData message anymore. At least for as long as there's no activity on the EV asset.
8. It is assumed that Charging Stations send their metervalues to the CSO 'clock aligned' once every 15 minutes, in case there is an active session.
9. CMO requires accumulated measurements per EV session. Therefore handshaking an INTERMITTENT AccumulatedMeasurementConfiguration is needed. A DetailedMeasurement with charging asset type will then represent a measured session.
10. CSO can send CMO an accumulated measurement (in kWh) per phase for generation and consumption asset types. CMO can use this to calculate a maximum remaining capacity (in kW) per phase for EV charging and converts this to Ampère (per phase) in form of a ForecastedBlock.
11. For convertion from kW to Ampère CMO will assume voltage is 230V.
12. CSO can send CMO an accumulated measurement (in kWh) of all phases together for EV charging asset types. CMO can use this to calculate the EV energy demand to obtain an optimum EV charging capacity.
13. CMO will send UpdateGroupOptimalCapacityForecast messages to the CSO in an arbitrary frequency. The minimal interval between these messages is 1 minute because CSO must have some time to process and send charging profiles to the Charging Stations.
14. CSO is in liberty to decide whether or not to obey CMOs suggested UpdateGroupOptimalCapacityForecast. CSO performs a sanity check on the model before turning it into updates in charging profiles.
15. CSO is able to deliver State Of Charge (in kWh) of a battery/storage using an InstantaneousMeasurement.
16. CMO will receive (per asset) a separate meter reading of NET/IMPORT/EXPORT, depending on which register is available at the assets site. There's no guarantee the assets number of meter readings stay the same over time, for instance in case of maintenance on the asset.
17. The capacity of a ForecastedBlock represents all phases.

5.4. Appendix D: TODOs

- Propose event base metering data; instant notifications not only for starting/stopping sessions
- CSO temporary 'overrule'. If for all sessions, effectively nothing changes. (Nash equilibrium).
- In case group represents a cable, specifying all 3 phases seperately is permitted. In this case there should be one ForecastedBlock instance per phase. How should be determined whether or not ForecastedBlock represents a cable?
- Add a creation timestamp to both envelopes. (This also tackles the 'problem' that the DSO does not know for which period a measurement is sent).
- Discussion: make the forecast period (of 24 hours) configurable using the handshaking mechanism. This makes OCMP less dependent on the business rules.
- Discussion: should OCMP support forecasts of generation (rather than consumption)? Currently, the 'capacity' attribute of ForecastedBlock has a minimum of 0.
- Add an extra example on sending multiple AccumulatedMeasurement objects.