Horizon 2020 – LCE - 2017 - SGS

FLEXCoop

Democratizing energy markets through the introduction of innovative flexibility-based demand response tools and novel business and market models for energy cooperatives

FLEXCoop

D2.6 – FLEXCoop Framework Architecture including functional, technical and communication Specifications – Preliminary Version

Due date: 30.09.2018 **Delivery Date**: 12.10.2018

Author(s): Peter Hasse (Fraunhofer), Karsten Isakovic (Fraunhofer), Germán Martínez (ETRa), Katerina Valalaki (Hypertech), Tsitsanis Tasos (Hypertech), Tsiakoumi Matina (Hypertech), Christos Malavazos (Grindrop), Effie Bachrami (Grindrop), Hrvoje Keko (KONČAR), Stjepan Sučić (KONČAR), Hrvoje Keserica (KONČAR), Eloi Gabaldon (CIMNE), Jordi Cipriano (CIMNE), Dimitris Panopoulos (S5), Ioanna Michael (S5)

Editor: Peter Hasse (Fraunhofer)

Lead Beneficiary of Deliverable: Fraunhofer

Dissemination level: Public **Nature of the Deliverable:** Report

Internal Reviewers: Katerina Valalaki (Hypertech), Andreas Muñoz Zuara (CIRCE)

FLEXCOOP KEY FACTS

Topic:	LCE-01-2016-2017 – Next generation innovative technologies enabling smart grids, storage and energy system integration with increasing share of renewables: distribution network
Type of Action:	Research and Innovation Action
Project start:	01 October 2017
Duration:	36 months from **01.10.2017** to **30.09.2020** (Article 3 GA)
Project Coordinator:	Fraunhofer
Consortium:	13 organizations from nine EU member states

FLEXCOOP CONSORTIUM PARTNERS

Fraunhofer	Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V.
ETRa	ETRA INVESTIGACION Y DESARROLLO SA
HYPERTECH	HYPERTECH (CHAIPERTEK) ANONYMOS VIOMICHANIKI
DTU	DANMARKS TEKNISKE UNIVERSITET
GRINDROP	GRINDROP LIMITED
CIRCE	FUNDACION CIRCE CENTRO DE INVESTIGACION DE RECURSOS Y CONSUMOS ENERGETICOS
KONČAR	KONCAR - INZENJERING ZA ENERGETIKUI TRANSPORT DD
SUITE5	SUITE5 DATA INTELLIGENCE SOLUTIONS Limited
S5	SUITE5 DATA INTELLIGENCE SOLUTIONS Limited
CIMNE	CENTRE INTERNACIONAL DE METODES NUMERICS EN ENGINYERIA
RESCOOP.EU	RESCOOP EU ASBL
SomEnergia	SOM ENERGIA SCCL
ODE	ORGANISATIE VOOR HERNIEUWBARE ENERGIE DECENTRAAL
Escozon	ESCOZON COOPERATIE UA - affiliated or linked to ODE
MERIT	MERIT CONSULTING HOUSE SPRL

Disclaimer: FLEXCoop is a project co-funded by the European Commission under the Horizon 2020 – LCE - 2017 – SGS under Grant Agreement No. 773909.

EXECUTIVE SUMMARY

This deliverable describes in detail the design of the functional components of the FLEXCoop architecture. The interfaces of each component inside the framework as well as to external communication end points are the main focus of this document.

Each component is described in a separate subsection of Section 3. These subsections are divided into four topics to provide a complete picture on how these components are designed and how they interact. In each component, the programming language as well as the libraries and frameworks planned to be used are described based on experience and problem domain. Also, the corresponding deployment strategies are covered in the component sections. This way different approaches on deployment and lifecycle management can be evaluated during the implementation phase of the project.

To allow a heterogeneous software landscape the **Middleware** acts as a central communication hub between all components. The **Middleware** is also responsible of authentication and management of access rights, which will be implemented based on OpenAM. For the intercomponent communication an OpenADR top node is implemented in the **Middleware**.

Another core component of the architecture is the **Open Smart Box**. The **OSB** is a smart hardware device operating as a Real time Monitoring Sensor/ Actuator Node. It gathers ambient sensor information and usage data which are used by the forecasting and profiling components. These components enable the FLEXCoop system to determine the flexibility that can be offered for each household without reducing the comfort of the inhabitants.

Two visualization toolkits provide access to data, contracting and status of the Distributed Energy Resources (DER). The **Prosumer Visualisation Toolkit** enables the Prosumer to monitor the status of its connected DERs as well as its current contracting options and demand response events triggered by the Aggregator. The **Visualization Aggregator Toolkit** on the other hand provides views, specific for the Aggregator role. It supports the Aggregator by its task to balance demand and production and visualizes the contractual offers to the Prosumers.

The **Open Marketplace** and the DER Registry are the background services that enable contracting and DER device handling inside the FLEXCoop architecture. All connected DERs report their status to the DER registry. The Open Marketplace is the core component for contracting. It combines the available flexibility options with the connected DERs into contracts between Aggregator and Prosumer.

This documentation will be the reference for the implementation and more detailed specification during the progress of the project. It is a common effort of all implementing partners.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AGR	Aggregator
CO	Confidential, only for members of the Consortium (including the Commission Services)
CHP	Combined heat and power
CMP	Component
D	Deliverable
DER	Distributed Energy Resources
DHW	Domestic Hot Water
DoW	Description of Work
DR	Demand Response
DSO	Distribution System Operator
DSS	Dispatch Service System
DSSy	Decision Support System
ESB	Message Oriented Middleware
EV	Electric Vehicle
FLOSS	Free/Libre Open Source Software
GDM	Global Demand Manager for Aggregators
GUI	Graphical User Interface
H2020	Horizon 2020 Programme
HVAC	Heating, Ventilation and Air Conditioning
IPR	Intellectual Property Rights
IED	Intelligent Electronic Device
JSP	Java Server Pages
MGT	Management
MS	Milestone
MVC	Model View Controller
O	Other
OS	Open Source
OSB	Open Smart Box
P	Prototype
P2H	Power-to-Heat
PM	Person Month
PROS	Prosumer

PU	Public
R	Report
RES	Renewable Energy System
RTD	Research and Development
SEAC	Security Access Control
SGAM	Smart Grid Architecture Model
TOGAF	The Open Group Architecture Framework
UML	Unified Modelling Language
VPP	Virtual Power Plants
VTES	Virtual Thermal Energy Storage
VTN	Virtual Top Node (OpenADR)
VEN	Virtual End Node (OpenADR)
WSN	Wireless Sensor Network
WP	Work Package
Y1, Y2, Y3	Year 1, Year 2, Year 3

1. INTRODUCTION

The compliance and use of open standards is a key success factor for the project and its further replication and commercialization. FLEXCoop Task 2.5 will provide the required guidance and input to ensure the achievement of this key objective. It will review the standardization landscape and evaluate the latest evolutions in Demand Response (DR), interoperability between energy market stakeholders and communication between devices and systems.

Based on this initial analysis, along with the results of FLEXCoop T2.1, T2.5 will deliver the overall architecture of the FLEXCoop framework and the specifications of the key components and their functionalities. Specifically, the following aspects will be defined:

(i) Conceptual Architecture Design:
An overview of the system architecture describing the components and introducing the various sub-components, their interfaces and the connections with external systems (i.e. interoperability with existing Smart Home systems, multi-sensorial infrastructure or DERs at building and district level, interoperability interfaces for Machine to Machine communication, interfaces for the communication between the different actors in the DR value chain);

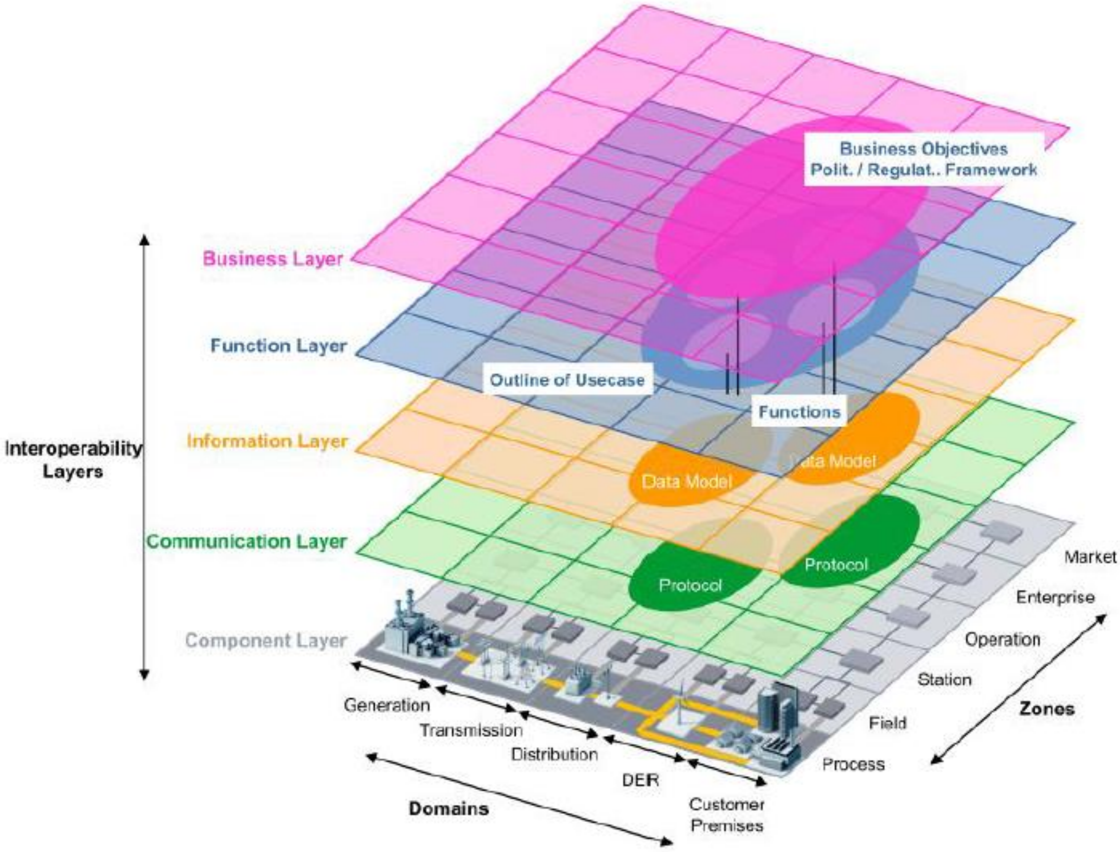
(ii) Modules Functional and Technical Specifications:
The purpose of this part of the architecture is twofold:
   a. to provide a high-level sketch of dependencies among different parts of the framework (e.g. individual components interfaces, etc.) and
   b. to describe in detail the constraints of the system elements in terms of hardware and/or software resources, compatibility with standards, etc.; Detailed Design of Individual Components of the Framework: refers to the detailed description of the functionalities, non-functional specifications as well as communicational requirements for the high-level building blocks of the FLEXCoop framework.

To deliver the aforementioned architectural definitions and to materialise the conceptual architecture design, state of the art software engineering tools will be used (e.g. UML activity and sequence diagrams, actors, etc.). Considering interoperability, scalability and flexibility of the FLEXCoop framework, the Internet of Things paradigm will be followed while analysing and evaluating the suitability of main standard-based communication protocols, smart home communication protocols (Zigbee, Bluetooth, 6LowPan, Z-Wave), open standards and data models (OpenADR, OneM2M/ SAREF, USEF, IEC-61850) and data modelling approaches (JSON, XML). These protocols have been evaluated as part of FLEXCoop D2.3 and are therefore not covered in this document.

1.1. Smart Grid Architecture Model

The Smart Grid Architecture Model Framework (SGAM) aims at offering a support for the design of smart grids use cases with an architectural approach allowing for a representation of interoperability viewpoints in a technology neutral manner.

It is a three-dimensional model merging the dimension of interoperability layers with the two dimensions of the Smart Grid Plane.



[1]

Figure 1: SGAM Architecture Model

The x-axis of the Smart Grid Plane (Domains) divides the problem domain "electrical power supply" into the individual sections:

- Bulk Generation: generation of electrical energy in bulk quantities
- Transmission: infrastructure and organization which transports electricity over long distances
- Distribution: infrastructure and organization which distributes electricity to customers
- Distributed Energy Resource: distributed electrical resources directly connected to the public distribution grid, applying small - scale power generation technologies
- Customer Premises: Hosting end users of electricity and also producers of electricity. The premises include industrial, commercial and home facilities

[1] https://de.wikipedia.org/wiki/Datei:Das_SGAM-Framework.png, Creative Common License

The y-axis of the Smart Grid Plane ("Zones") essentially reflects the automation pyramid, supplemented by the two zones "Enterprise" and "Market" (from the NIST Conceptual Model[2]):

- Process: physical energy supply equipment
- Field: protection, control and monitoring equipment
- Station: spatial aggregation of the field zone, e.g. local SCADA system
- Operation: the energy system, e.g. distribution network control
- Enterprise: commercial and organizational processes
- Market: Market operations and interactions

The z-axis of an SGAM (Interoperability) layers the five interoperability levels. Individual aspects are considered separately on different SGAM planes:

- Business Layer: business view, e.g. economic and regulatory aspects
- Function Layer: functions and services between components from an architectural perspective
- Information Layer: transmitted information objects and data models
- Communication Layer: protocols and mechanisms for information exchange
- Component Layer: physical distribution of the components involved

The resulting SGAM framework thus enables the structured representation of a smart grid system with separate consideration of individual interoperability aspects.

- SGAM Business Layer
    - Business Actors and Business Goals
    - Business Use Case 1-n
- SGAM Function Layer
- SGAM Information Layer
- SGAM Communication Layer
- SGAM Component Layer

In the following, we will focus on the SGAM business layer and we will use it to map FLEXCoop solution constructing the “FLEXCoop smart grid plane”, as this is defined and analysed below.

1.1.1. The SGAM Business Layer Overview

In the SGAM business layer, every entity is located into the appropriate domains and zones, contributing to the detailed description of each market actor, as well as the economic structures and policies of the parties involved. In [1], the need for ensuring alignment between the work of the M/490 working groups SG-CG/SP (Sustainable Processes WG), SG-CG/RA (Reference Architecture WG), and the development of a generic market model by the EU taskforce smart

[2] NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0. Publisher: National Institute of Standards and Technology. 2014.

grids (EG3) has been identified. The basis for alignment has been created by the use of the meta-model of TOGAF 9.1 shown in the figure below.

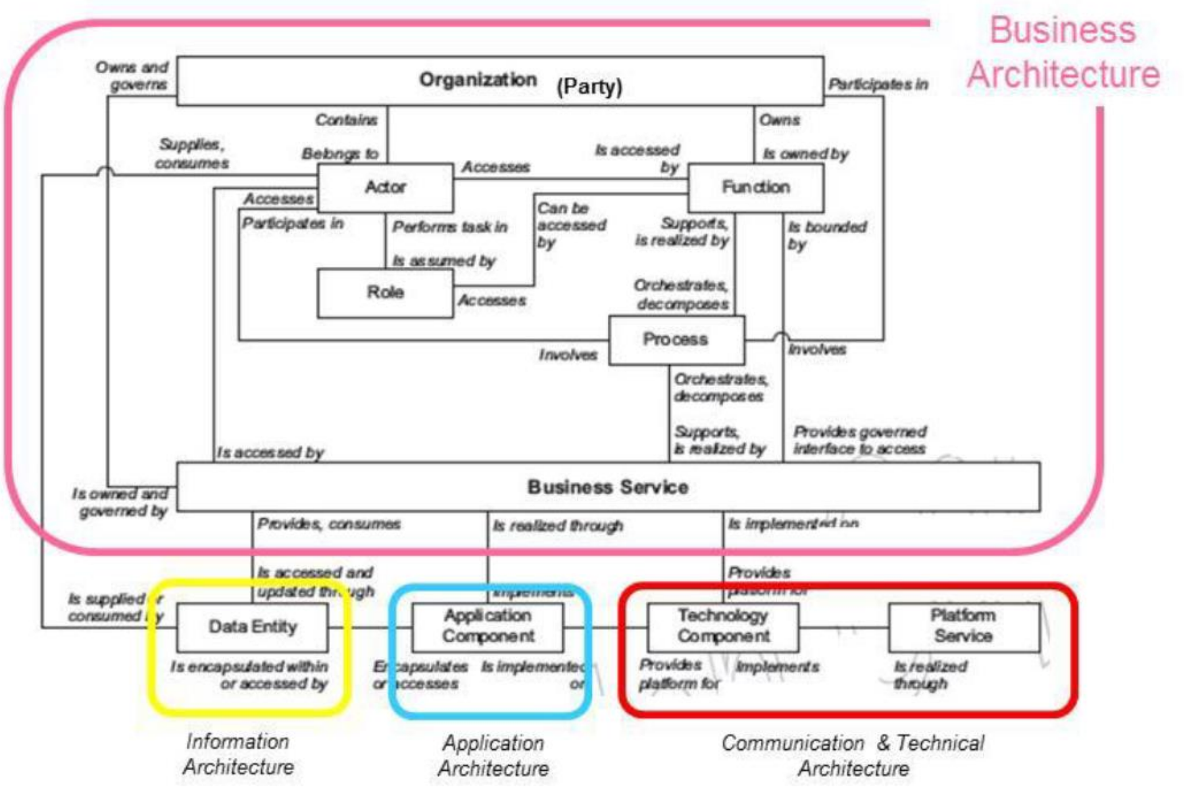


Figure 2: Relation Meta-Model to SGAM [1]

As it can be observed, the business architecture layer involves business organizations (parties), actors, roles, functions, processes and services, as well as the interactions between them (as these are further detailed below).

The business layer can be characterized as the most peculiar layer of the SGAM Framework. While being atop the other four layers, this is where the information provided by each high-level use case is combined, into producing the final business cases. After all, the scope of ICT solutions is the support of the business procedures. Of course, the objectives and constraints of every use case need to be considered.

The interconnection of the business layer to lower SGAM layers can be a somewhat complex matter. The total procedure should be performed with particular attention, in order to avoid the loss of information exchanged. However, based on the DoW, mapping to the lower SGAM layers is out-of-the-scope of the FLEXCoop architecture design. In any case, in order to reach and maintain alignment between market model developments and ICT architecture & services development, the business architecture layer should define and list [1]:

- Roles & actors
- Business functions (or business function model)
- Business services
- Business processes (or business process model)

All these are further analysed in the sections below while the following Figure depicts the alignment process between market model developments and ICT architecture & services development as has been proposed by CEN-CENELEC-ETSI Smart Grid Coordination Group [1].

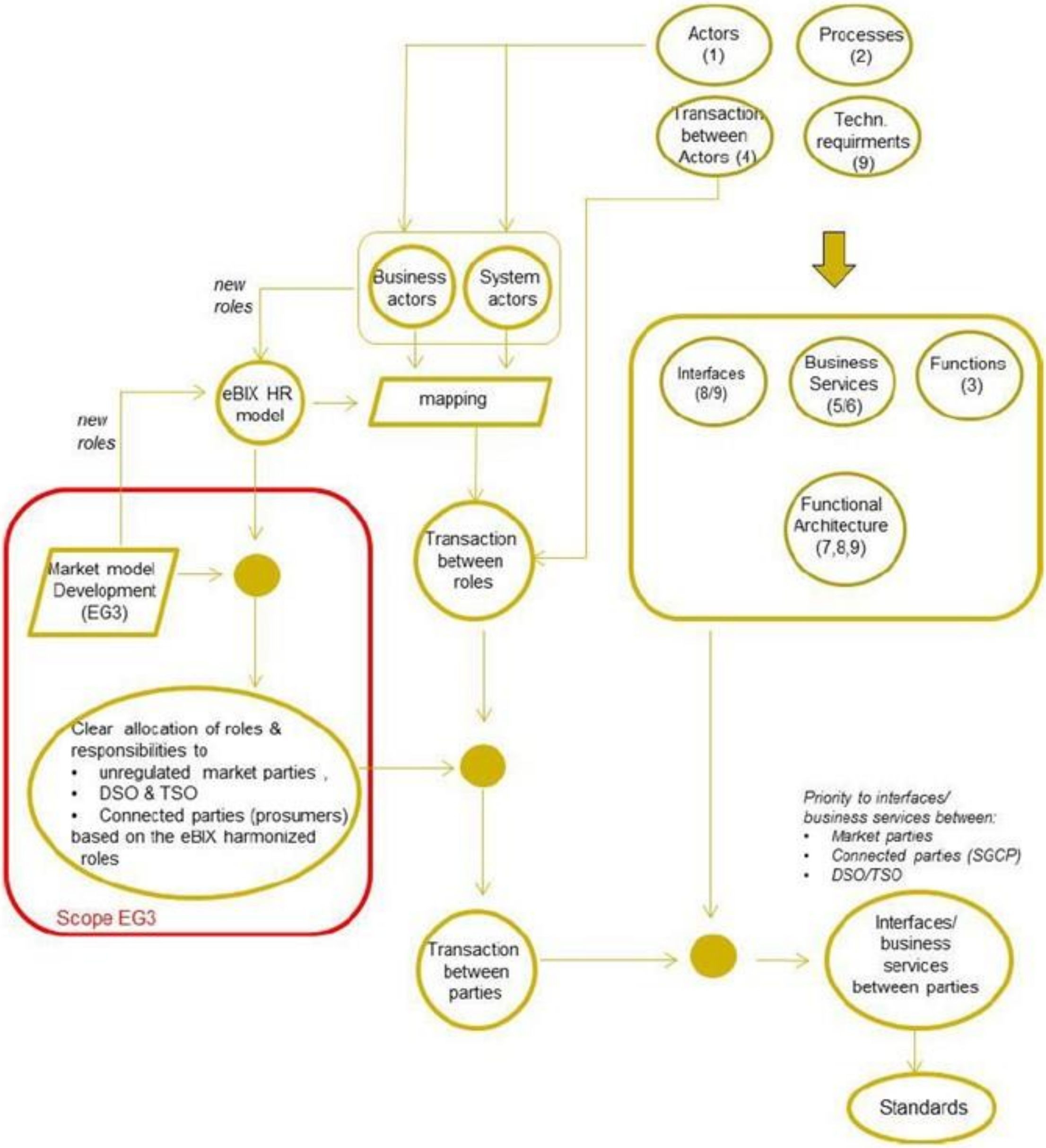


Figure 3: Alignment process between market model developments and ICT architecture & services development [1]

1.1.1.1. SGAM Roles & actors

The ***roles*** are mainly defined in terms of responsibility (ENTSO-E/eBIX [2]) and are allocated to market parties. A party is defined as a legal entity performing one or more roles [3].

An “***Actor***” is very general and can cover people, systems, databases, organizations, and devices. Actors, as identified by SG-CG/SP [3], might be divided into business actors and system actors when referring to technological systems. More specifically [1]:

- ***System actors*** are covering functions or devices which for example are defined in the Interface Reference Model (IEC 61968-1). A system actor will perform a task under a specific role.
- A ***business actor*** specifies in fact a “role” and corresponds with roles defined in the eBIX harmonized role model [2].

1.1.1.2. SGAM Business Functions

SGAM defines business function using the TOGAF definition as follows: a business function delivers business capabilities closely aligned to an organization, but not necessarily governed by the organization [4].

1.1.1.3. SGAM Business Services

Again, SGAM uses TOGAF’s definition of business services [1]. More specifically: a business service supports a business function through an explicitly defined interface and is explicitly governed by an organization.

Actors in the conceptual model are connected by associations. Where these actors are represented by applications, information is exchanged via application interfaces. Where these interfaces cross boundaries between market parties, we define the information exchanged as business services. Through these business services market parties will interact.

A Smart Market in SGAM is defined as an unregulated environment where energy products and energy-related services are freely produced, traded, sold and consumed between many market actors [1].

A Smart Grid in SGAM is defined as a regulated environment where energy is transported and distributed via energy networks, and which provides relevant data & functionality to facilitate envisioned market functioning (e.g. switching customers, providing metering data) [1].

1.1.1.4. SGAM Business Processes

In order to realize business services between markets parties, it is important to have a good insight in the underlying business processes. Furthermore, the business processes drive the requirements for the functional and information architectures.

1.1.2. FLEXCoop Business Architecture

To ensure an interoperable FLEXCoop solution, its relation to markets, products and processes has to be well understood and aligned. Only this way an ICT solution can really support the business and this is one of the FLEXCoop’s main goals. This logic is well presented in the SGAM, showing the business layer as the top layer of the SGAM framework.

1.1.2.1. FLEXCoop actors

In the table below, we have identified the business and system actors of the FLEXCoop business model and mapped them to the actors as these defined in SGAM (see Section 1.1.1.1).

FLEXCoop Business Actors

It should be mentioned that for the FLEXCoop business actors mapping, we have used the newest version of the harmonised electricity market role model provided by ENTSO-E [2].

In FLEXCoop, we have identified two business actors: Aggregators and Prosumers. Their responsibilities/roles as well as how each one of them can interact with the FLEXCoop solution is detailed in the Section 1.2. Herein, we provide a mapping to the SGAM business actors (and more specifically to the roles defined in [2]) to facilitate afterwards the mapping of the FLEXCoop business model to the SGAM business layer. The mapping also takes into account the different FLEXCoop use cases as these have been provided and analysed in the deliverable D2.1 "End-user & Business Requirements".

It should be mentioned that when a role is a sub-set of another role e.g. "consumer" is a sub-set of the "Party Connected to the grid", only the highest one is provided and the sub-sets are inferred.

FLEXCoop Business Actor	Role defined in [2]	Role description defined in [2]	Relevant use-case(s)
Aggregator	Market Information Aggregator	A party that provides market related information that has been compiled from the figures supplied by different actors in the market. This information may also be published or distributed for general use.	UC-02 UC-03 UC-04 UC-05 UC-06 UC-07 UC-08 UC-09(bis) UC-10
	Balance Responsible Party	A party that has a contract proving financial security and identifying balance responsibility with the Imbalance Settlement Responsible of the Market Balance Area entitling the party to operate in the market. This is the only role allowing a party to nominate energy on a wholesale level.	UC-10

	Balance Supplier	A party that markets the difference between actual metered energy consumption and the energy bought with firm energy contracts by the Party Connected to the Grid. In addition, the Balance Supplier markets any difference with the firm energy contract (of the Party Connected to the Grid) and the metered production.	UC-04 UC-05 UC-06 UC-09(bis) UC-10
	Billing Agent	The party responsible for invoicing a concerned party	UC-06
	Scheduling Coordinator	A party that is responsible for the schedule information and its exchange on behalf of a Balance Responsible Party.	UC-04 UC-05 UC-09(bis) UC-10
	Reconciliation Accountable	A party that is financially accountable for the reconciled volume of energy products for a profiled Accounting Point.	UC-02 UC-03 UC-04 UC-05
	Metered Data Aggregator	A party responsible for the establishment and qualification of metered data from the Metered Data Responsible. This data is aggregated according to a defined set of market rules.	UC-03 UC-04 UC-05 UC-06
	Meter Administrator	A party responsible for keeping a database of meters.	UC-07
Prosumer	Party Connected to the grid	A party that contracts for the right to consume or produce electricity at an Accounting Point.	UC-01, UC-02 UC-04, UC-05 UC-06 UC-07 UC-08 UC-09(bis) UC-10 UC-11(bis) UC-12

Table 1: FLEXCoop business actors mapped to the SGAM business actors

As it is clearly deduced from the table above, the FLEXCoop aggregator actor assumes multiple roles based on the FLEXCoop use case ranging from a Balance Responsible Party in use cases where microgrids are formed to a Billing Agent for the use case considering DR settlement and remuneration. In FLEXCoop, the aggregator and, more specifically, the cooperatives that assume the role of aggregator, concentrate a number of different roles aiming at providing an innovative and feasible business model that can be proven valuable for all the stakeholders involved.

FLEXCoop System Actors

Every business actor must achieve a goal, related to a business transaction. While a close correlation between the goals to be achieved and the role that an actor undertakes exists, the majority of actors aim at the achievement of a multiple number of goals. For example, while a prosumer wants to minimize the energy bills, the minimization of emissions produced, is also to his benefit. As a more general business goal, the optimization of the energy that is produced/consumed could be considered. A presentation of the system actors that we have identified in FLEXCoop are summarised in the table below.

FLEXCoop System Actor	Type	Description[3]
Visualisation/Aggregator Toolkit	Application	User interface of the Global Demand Manager, DER Registry and Open Marketplace for the aggregators
Visualisation/Prosumer Toolkit	Application	User interface for DER Registry, Open Marketplace and other information services for the prosumers
Flexibility, forecasting, segmentation and aggregation	Application	A novel module for aggregators for the management of the consumer demand and flexibility profiles
Global Demand Manager	Application	A module continuously analysing demand/storage flexibility, along with signals coming from the Distribution System Operator (DSO) and decide about the optimal configuration of demand-based dynamic Virtual Power Plants (VPPs)
DR Settlement/ Remuneration	Application	An application for accurate baselining of energy performance of prosumers and normalization

[3] Detailed description of system actors' functionalities and technical specifications can be found in the Section 3.

Open Marketplace	Application	A platform accessible by aggregators and prosumers allowing aggregators-prosumer matchmaking
Local Demand Manager	Application	A module using flexibility-based optimization algorithms for intra-building DR optimization
Demand Flexibility Profiling	Application	A module delivering Holistic Context-Aware Flexibility Profiles, reflecting real-time demand and storage flexibility as a function of multiple parameters
DER Registry	Application	A distributed semantically enhanced DER registry which will publish and advertise available DERs by prosumers along with their capabilities
IEC 61850 Server/ DER Management System	Application	An IEC-61850 server enabling communication with district-wide DERs (generation and storage)
Open Smart Box (OSB)	Device	The FLEXCoop real-time monitoring sensor/ actuator node
Distributed Energy Resources (DER)	Device	DER generation/storage and DER loads
Sub-metering equipment	Device	Metering and sub-metering equipment required for acquiring real-time energy data
Sensors/ Actuators	Device	Sensors and actuators that may be required for real-time monitoring of environmental context/ conditions

1.1.2.2. FLEXCoop solution mapped into the SGAM business layer

The FLEXCoop business architecture aims at ensuring that the proposed business model is going to be implemented in the overall FLEXCoop framework using the correct business services and underlying architectures in a consistent and coherent way. The FLEXCoop business architecture is modelled in the business layer of the SGAM, and it is further analysed and enhanced to serve FLEXCoop objectives. Thus, it is not limited in the markets and enterprise zone of the SGAM layer but all the zones and domains are used to map the whole FLEXCoop solution in a conceptual way. This modelling approach can be further utilised towards providing the respective recommendations for wide adoption in national and EU market design frameworks. In particular, a detailed energy and ICT infrastructure layout is proposed, which, combined with the requirements and use cases identified in the FLEXCoop deliverable D2.1 “End-user & Business Requirements”, led us to identify components functional and technical specifications, as these are presented in the Section 3 of the current deliverable. The proposed mapping has been inspired by [5] and was further customised to reflect the FLEXCoop needs. At a high level, the cross-domain smart grid function "end-to-end automated Demand Response optimisation" (introduced in FLEXCoop) is mapped to the business layer of the SGAM model. This will be used as a basis and will be tested and validated throughout the project implementation to check if such a business architecture can be supported by existing information and communications standards.

In general power system management distinguishes between electrical process and information management viewpoints. These viewpoints have been considered and are clearly shown in the FLEXCoop business architecture that is mapped in the business layer of SGAM (see Figure below). In other words, the mapping developed and provided herein can be considered as the ***FLEXCoop smart grid plane***, which is depicted in the Figure 4.

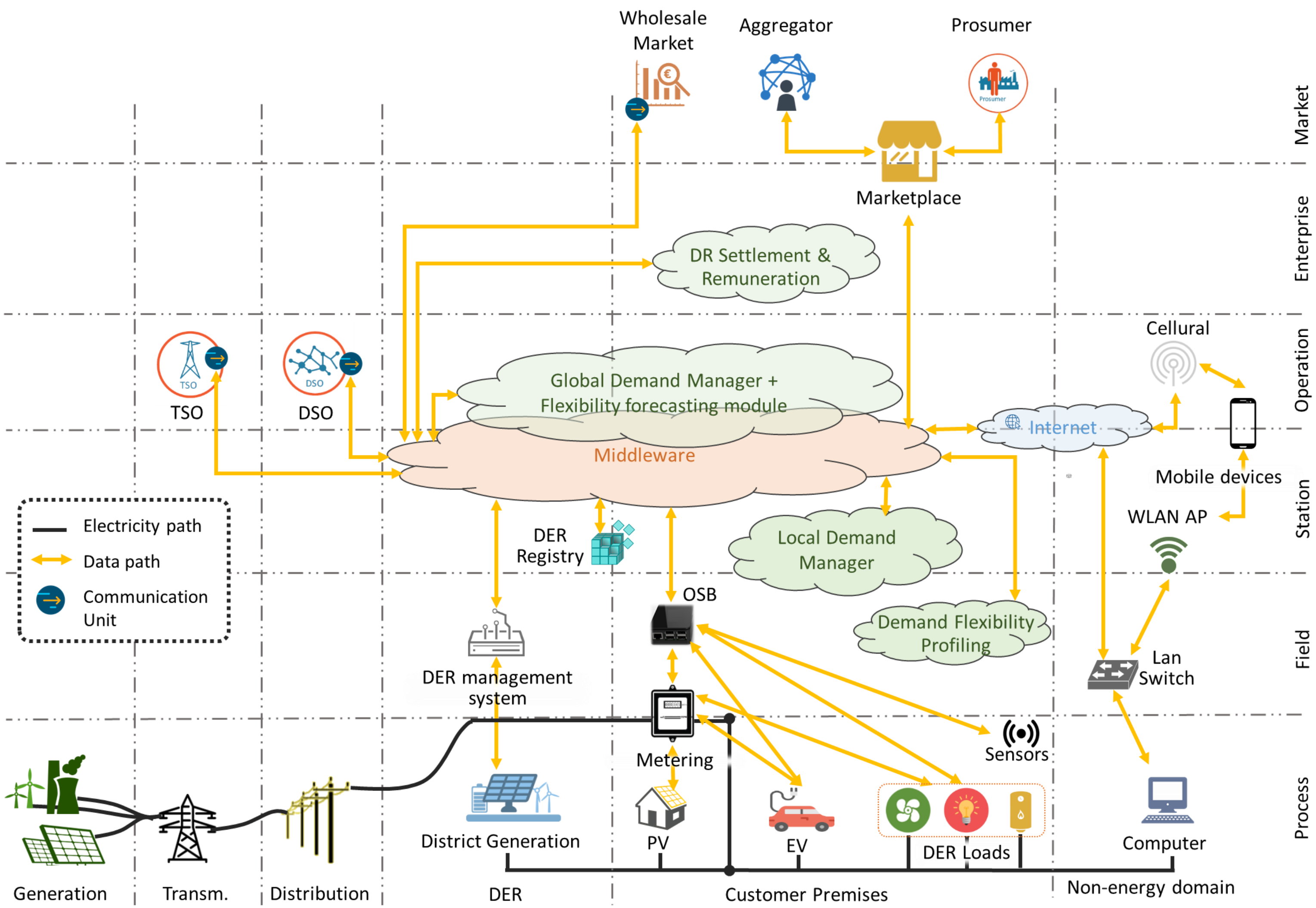


Figure 4: The FLEXCoop smart grid plane

1.2. Business actors’ interaction with FLEXCoop system components

To model the requirements there are two roles of the energy market to take into account. The most important role in terms of control is the Aggregator. An Aggregator in the energy market is responsible of gathering the demand of energy of the customers in his control region as well as their energy production. Based on this information the Aggregator calculates a schedule or plan for the next days. The Aggregator can than optimize the schedule with based on different metrics e.g. for obtaining better prices, services or other benefits.

The second role that is relevant for modelling the Prosumers. Where in the past this role was called a consumer this changes with the transition to DER systems to the term Prosumer. The term Prosumer describes a market participant that is not only consuming energy from the grid, but also can feed energy into the grid. The energy can either be locally generated by e.g. photovoltaics, CHP and wind or taken from the grid and stored in a battery. In the second case there is no local energy generation but by storing overproduction and feeding it back to the grid when needed. Besides of batteries in buildings, batteries in cars can be also integrated.

Visualization Aggregator Toolkit	The Aggregator role can access all relevant information for its role via this component and its subcomponents
Visualization Prosumer Toolkit	The Prosumer role can access all relevant information for its role via this component and its subcomponents
Open Marketplace	The Aggregator and Prosumer can use the Open Marketplace for contracting via the corresponding Visualization Toolkit
DER Registry	The Aggregator and Prosumer can use the DER for status information of the DER systems via the corresponding Visualization Toolkit
Open Smart Box	The Prosumer can use the Open Smart Box to control Smart Home components

Table 2: Business Actor – FLEXCoop component interaction

2. Conceptual Architecture Design

This section provides an overview of the concepts and architecture decisions made so far in this project.

2.1. FLEXCoop architecture

Herein, we provide the conceptual architecture of the FLEXCoop integrated interoperable solution. It summarises in a high-level all the components that are developed in the project and their interactions.

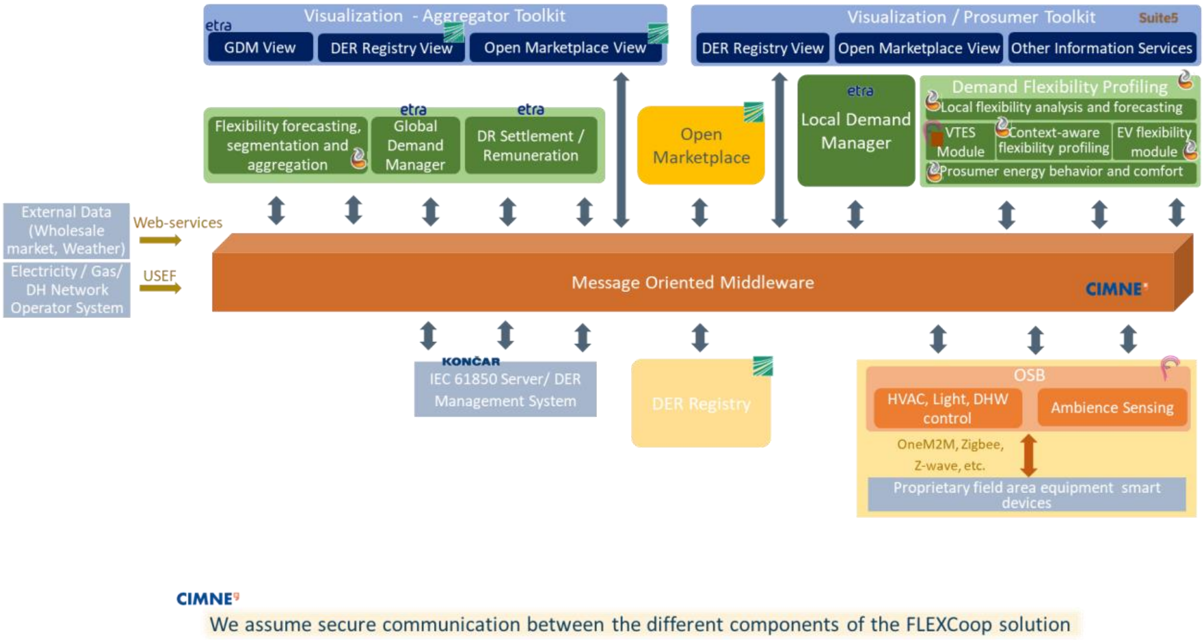


Figure 5: FLEXCoop Architecture Overview

2.2. Sequence Diagrams

Having defined the FLEXCoop business and system actors, and the conceptual architecture of the FLEXCoop solution, we proceed with their mapping to the use cases that have already been defined in the D2.1. To this end, the Unified Modelling Language (UML) sequence diagrams are used to simply depict interaction between FLEXCoop components in a sequential order i.e. the order in which these interactions should take place. The diagrams show how the FLEXCoop business actors (Aggregators, Prosumers) interact with the FLEXCoop solution and, in turn, how each component interacts with the other components to achieve the goal specified in each use case.

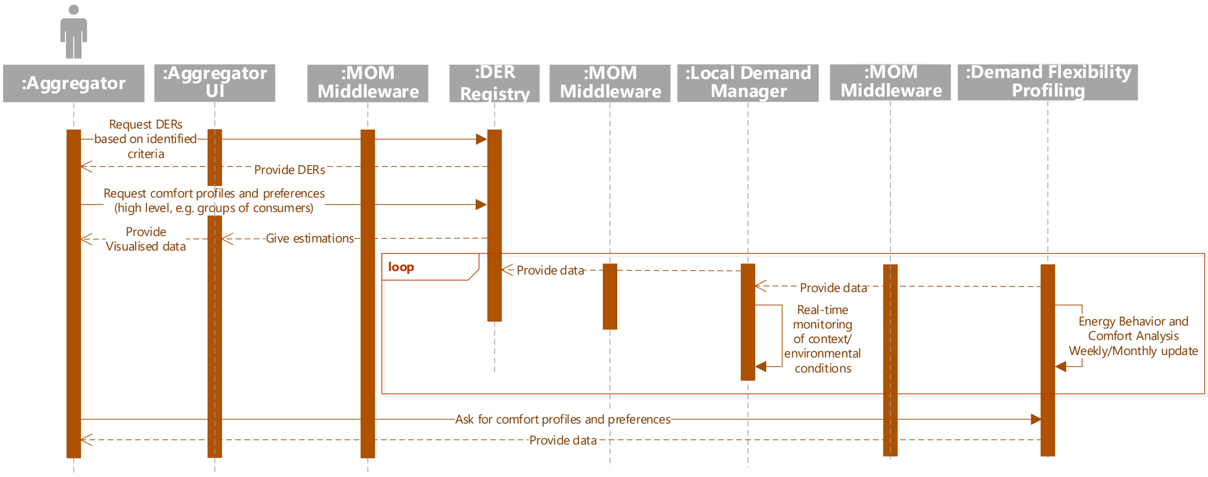


Figure 6: Use Case 1 - Sequence diagram

Use Case 1	Extraction of personalized (dis)comfort profiles towards the establishment of a user-centred management framework
Pre – conditions	The establishment of Wireless Sensor Network (WSN) in premises towards monitoring (in real time) environmental and operational conditions. In addition, setting initial configuration parameters about their comfort preferences
Use Case Path	Real time monitoring of environmental conditions and users' interaction over specific devices under contextual conditions. Further, periodical analysis (correlation) of data towards the extraction of accurate comfort and discomfort profiles. This is a process performed weekly/monthly towards the adaptation of comfort profiles in seasonal partners.
Post Condition	The extraction of (dis)comfort profiles will further facilitate the implementation of control actions in an automated way, ensuring that way the minimum of intrusiveness of end users/customers

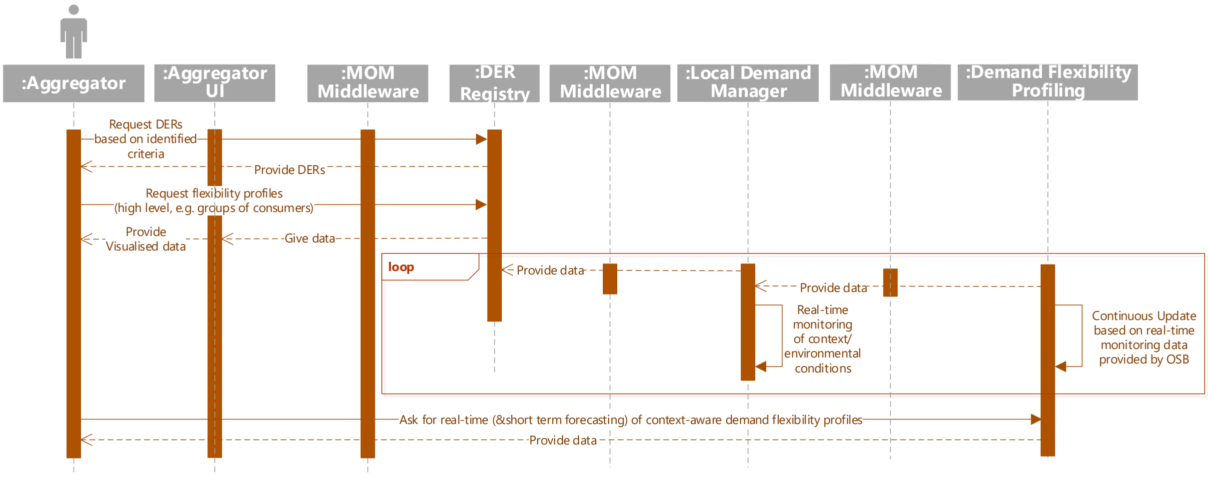


Figure 7: Use Case 2 - Sequence diagram

Use Case 2	Extraction of context-aware demand flexibility profiles
Pre – conditions	The establishment of WSN in premises towards monitoring (in real time) environmental and operational conditions. Extraction of accurate comfort profiles (user preferences), as constrain towards demand flexibility profiling analysis.
Use Case Path	• Extraction of demand flexibility profiling parameters following a training process. Historical data (device operational data & environmental conditions) are required towards the extraction of accurate profiling parameters (DER modelling) • Following the training process; real time (and short-term forecasting) calculation of demand flexibility potential (at device level and further aggregation at asset or portfolio level.
Post Condition	The extraction of context-aware demand flexibility profiles is the main prerequisite towards the accurate implementation of explicit (automated) DR campaigns in a non-intrusive way (comfort profiling parameters incorporated in the flexibility profiling framework)

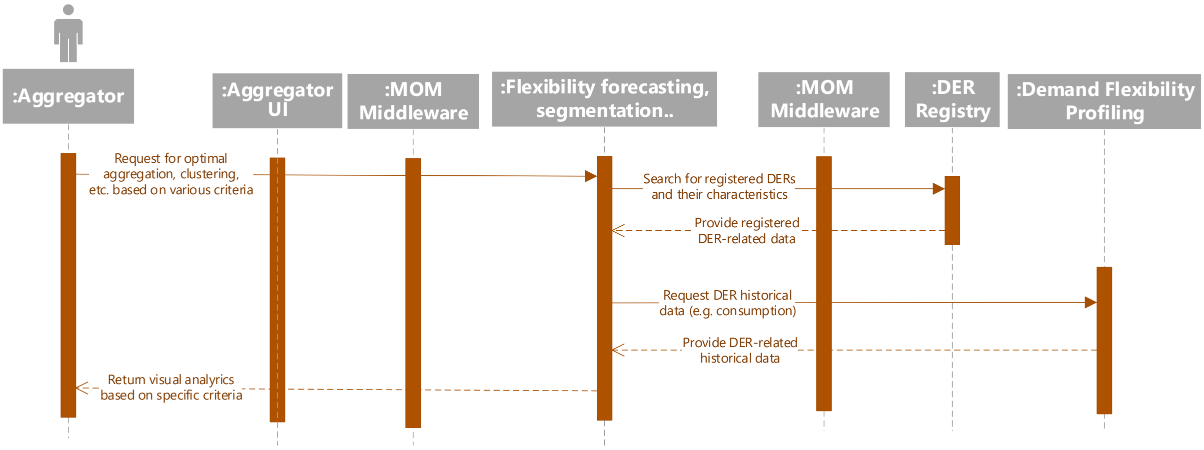


Figure 8: Use Case 3 - Sequence diagram

Use Case 3	Establishment of a data analytics framework for the optimal portfolio management
Pre – conditions	Availability of historical data (energy consumption, cost etc…) per asset. Definition of business models/ criteria that are of interest for Cooperatives (to further set the criteria for portfolio analysis)
Use Case Path	The stakeholder is selecting the criteria for the analytics process (selection of business objective and type of analysis). The data analytics engine (incorporating several analytics techniques is performing portfolio analysis towards the extraction of useful insights. These insights are further available through a User Interface (UI), to facilitate the Cooperative in the decision making process.
Post Condition	The extraction of analytics results will facilitate the formulation of spatio-temporal VPPs for the provision of better services to the customers and other market parties.

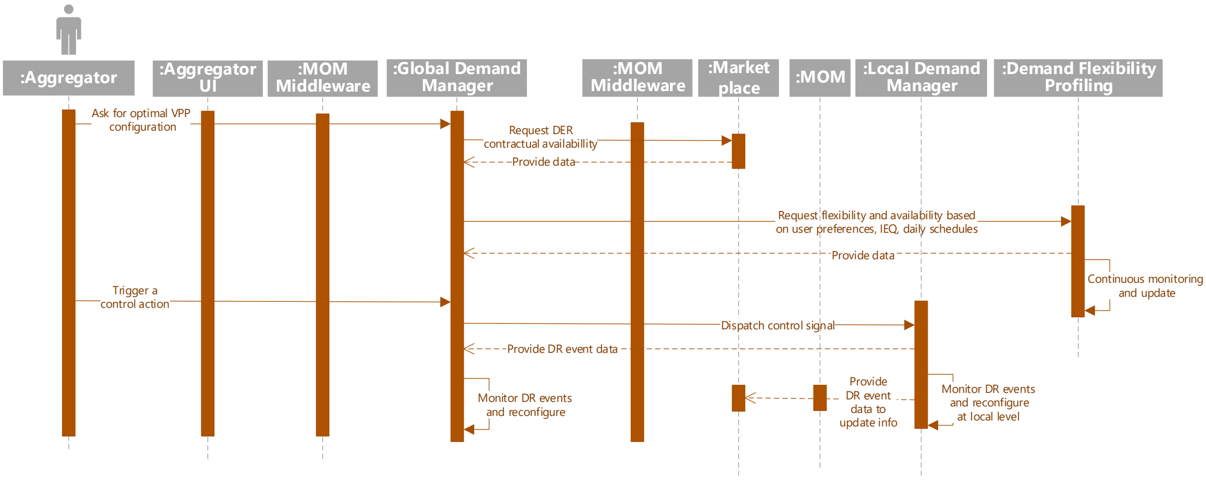


Figure 9: Use Case 4 - Sequence diagram

Use Case 4	Establishment of a real time fully automated DR triggering framework
Description (narrative)	The overall framework to promote personalized, human-centric and contract-safeguarded participation in demand response programmes in an automated way, preserving consumers' choice to opt-out at any time.
Pre – conditions	To identify flexibility and response capability of the consumers To define flexibility-based optimization algorithms for intra-building DR optimization The requirements of the Aggregator must be known in order to define these algorithms, including the possibility for consumers to opt-out of the programme at any time.
Use Case Path	Once the flexibility of the consumers and response capability have been identified, the framework will support on the decision making for the dispatch of personalized DR signals to them. This "decision making" framework is a continuous dynamic process that monitors the DR evolution to identify overrides or failures to respond and automatically revise the initially defined strategies so as to achieve the provision of the anticipated amounts of flexibility and in this sense, optimize business functions and energy transactions of all stakeholders involved.
Post Condition	To dispatch the corresponding control/signal actions to the loads, preserving comfort and indoor quality for the people

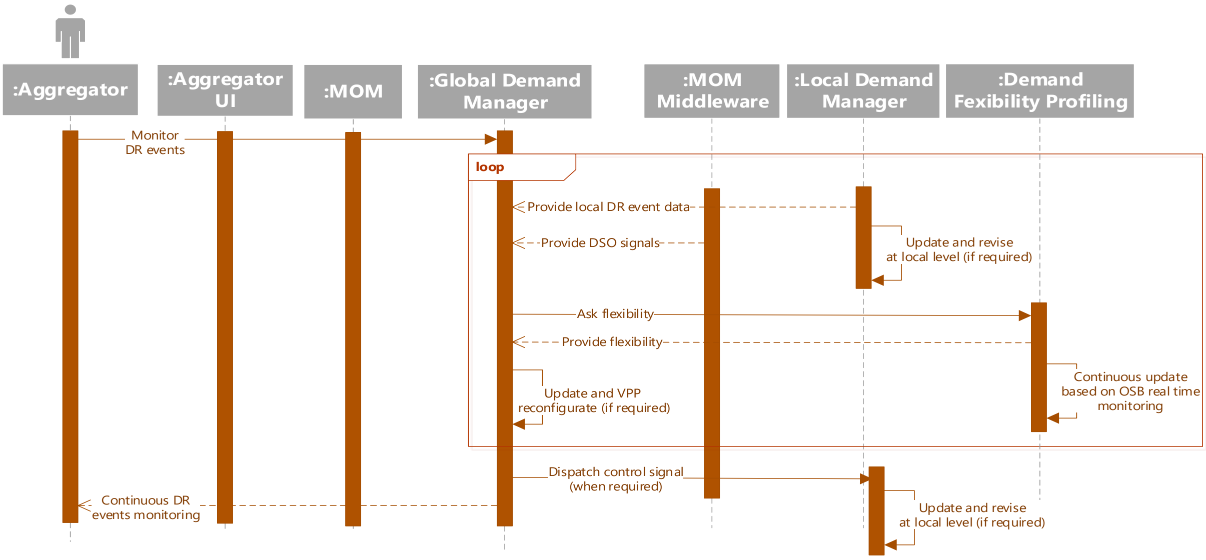


Figure 10: Use Case 5 - Sequence diagram

Use Case 5	Real time monitoring of DR strategies implementation and re-configuration of dynamic VPPs
Pre – conditions	Connection with the DSO is required for gathering the signals
Use Case Path	The demand/storage flexibility and information with the signals coming from the DSO are continuously analysed to decide the optimal configuration of demand-based VPP to timely respond and provide the required flexibility to the grid. Fully automated control signals will be dispatched, based on a bilateral negotiation process that considers real-time contextual information involving Consumers and Aggregators, avoiding any kind of penalization for both sides. Nevertheless, the evolution of each Demand Response event will be continuously monitor to identify overrides of the deployed strategies or failures, so it will be possible to revise the initially defined strategies with the idea of achieving the flexibility, optimize business functions and energy transactions of all the involved stakeholders.
Post Condition	Automated control signals with the aim to utilize the flexibility of the Local Demand Managers

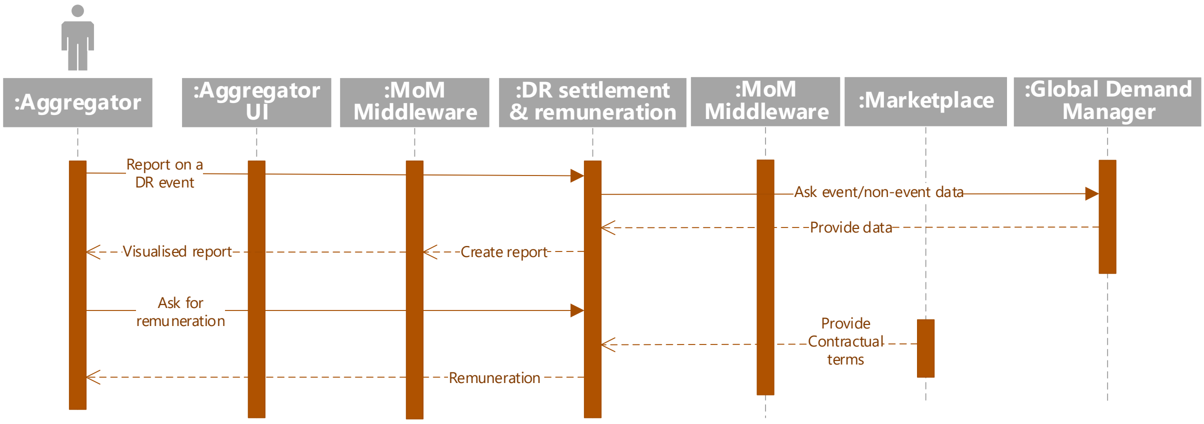


Figure 11: Use Case 6 - Sequence diagram

Use Case 6	DR settlement and Prosumer remuneration for DR participation
Pre – conditions	It is based on the FLEXCoop PMV methodology
Use Case Path	To define accurate baselines of energy performance/consumption of Prosumers and normalize it against varying occupancy patterns, energy uses and climatic conditions.
Post Condition	It will enable Aggregators to measure and verify the flexibility that has been activated by Prosumers during a DR event and accordingly remunerate them for their participation in DR.

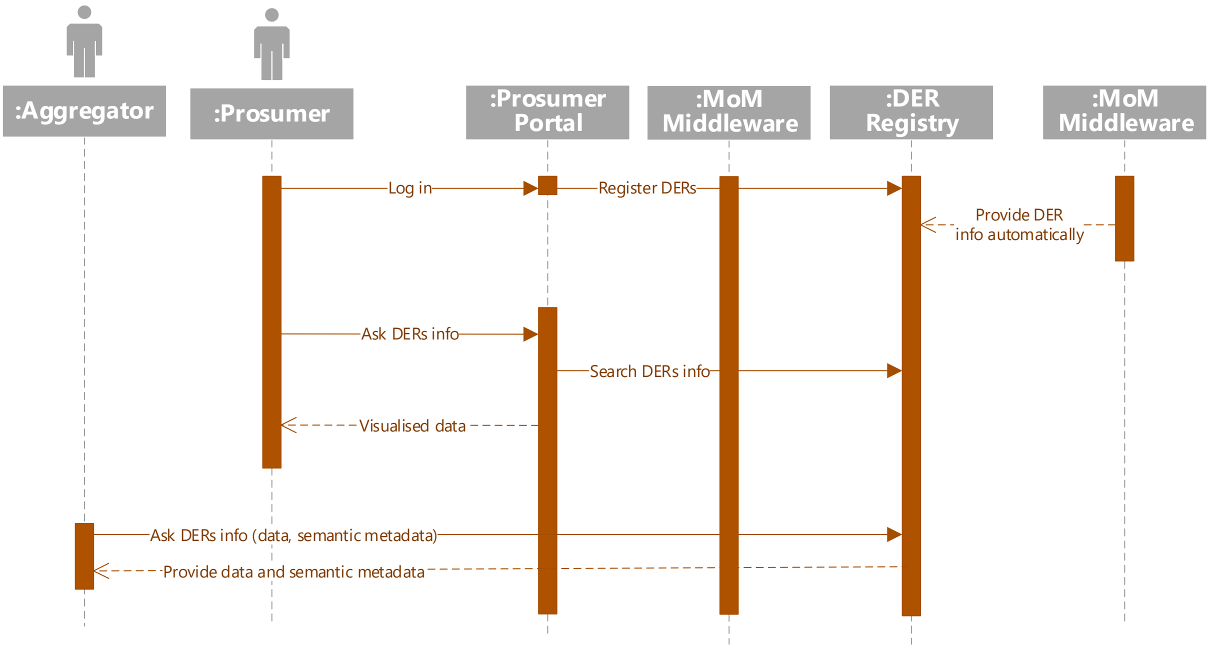


Figure 12: Use Case 7 Sequence diagram

Use Case 7	Establishment of DER Registry to ensure the openness of energy market to the final customers
Pre – conditions	The OSB has to be deployed and configured in the end users home.
Use Case Path	1. Prosumer signs in into the DER Registry via the FLEXCoop identity provider 2. Prosumer registers its DERs (smart device assets) in the DER Registry
Post Condition	Prosumer DERs are registered and discoverable by the Aggregators

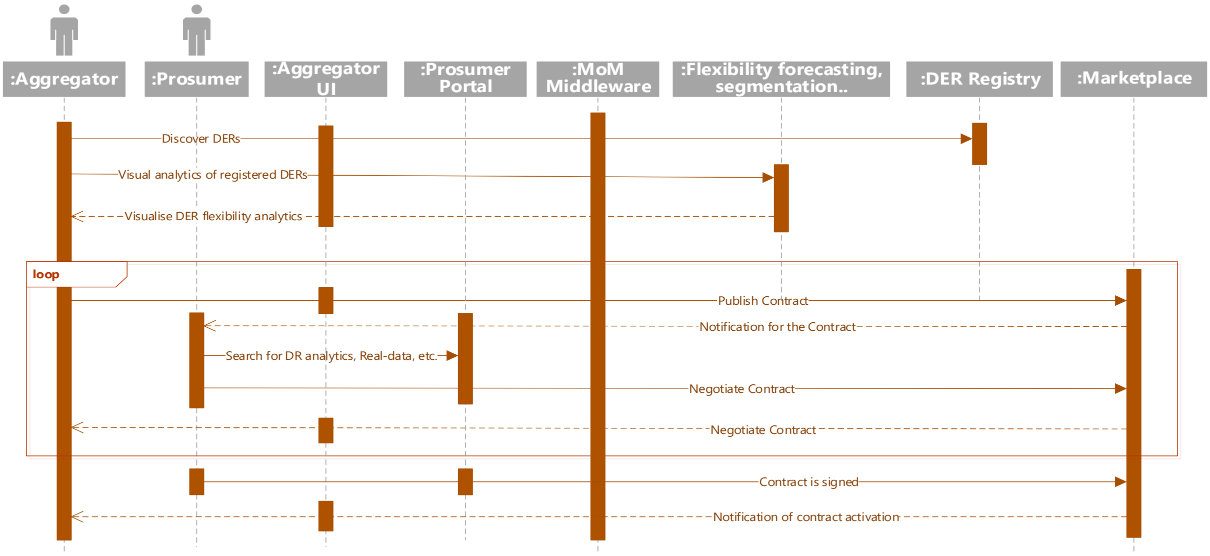


Figure 13: Use Case 8 Sequence diagram

Use Case 8	Establishment of Flexibility Pooling and Sharing Marketplace to promote end users empowerment in energy market
Pre – conditions	The OSB has to be deployed and configured in the end users home. The end user and its device assets have to be registered in the DER registry.
Use Case Path	1. Aggregators take full responsibility for their imbalances and become BRPs 2. Aggregators check for end user flexibility via DER Registry 3. Aggregators controls end user device assets in order to fulfil energy balancing needs
Post Condition	End users can participate in the energy market and can receive remuneration.

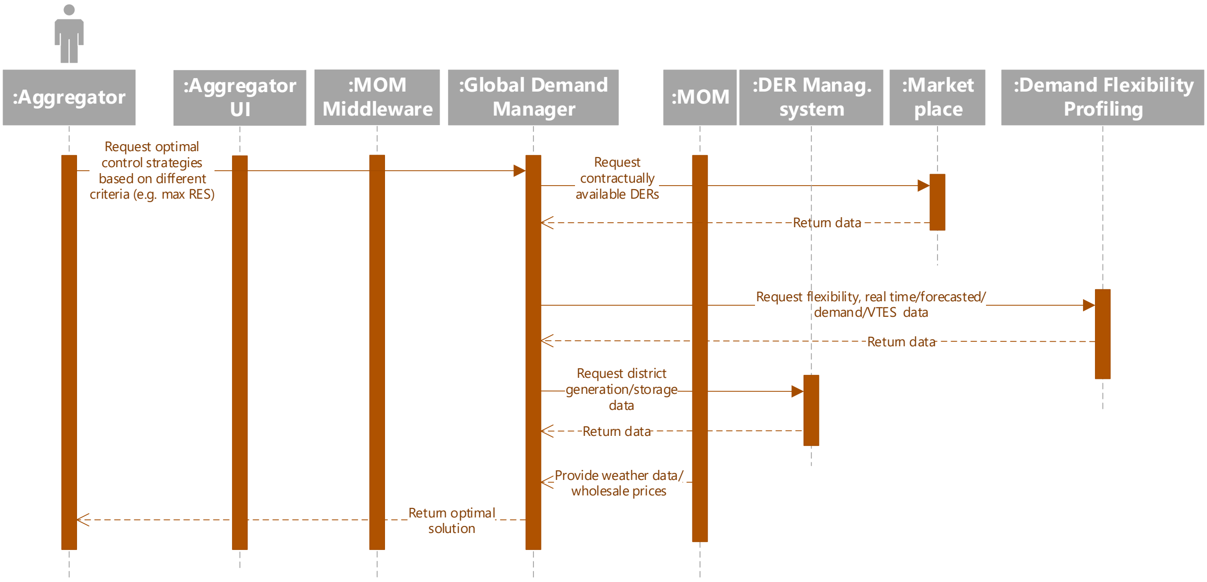


Figure 14: Use Case 9 Sequence diagram

Use Case 9	Promotion of self-consumption concept for maximizing consumption from local generation units
Pre – conditions	Availability of local RES and storage/ Integration with the overall framework Real time data (consumption, generation) from assets and controllability over the different asset types
Use Case Path	Real time and forecasted data about local demand, generation and storage are considered for the analysis. Data about flexibility (controllability) of each asset is available. The Dispatch Service System (DSS) engine will take into account these parameters (real time conditions and potential flexibility) towards the selection of the best control strategies that ensure the maximum of local RES exploitation.
Post Condition	Implementation of meso-term control strategies to ensure the maximization of self-consumption

The following two use-cases can be mapped in the same sequence diagram as above (Figure 14) considering that the only thing changed is the optimisation that the Global Demand Manager need to perform.

Use Case 9 bis	Promotion of system efficiency concept for optimising energy purchase on wholesale market based on price or CO_2 content
Pre – conditions	Availability of day ahead wholesale prices Availability of a CO2 index for wholesale electricity Real time data (consumption, generation) from assets and controllability over the different asset types
Use Case Path	Real time and forecasted data about local demand, storage and wholesale market prices are considered for the analysis. Data about flexibility (controllability) of each asset is available. The DSS engine will take into account these parameters (real time conditions and potential flexibility) towards the selection of the best control strategies that ensure the cheapest energy is bought from the market
Post Condition	Implementation of meso-term control strategies to ensure the optimization of energy purchase

Use Case 10	Cooperative-operated microgrids towards the establishment of independent entities in distribution grid
Pre – conditions	Availability of local RES and storage/ Integration with the overall framework Real time data (consumption, generation) from assets and controllability over the different types of assets
Use Case Path	The end user is monitoring the status of the local grid. Real time and forecasted data about local demand, generation and storage are considered for the analysis. Data about flexibility (controllability) of each asset is available. The DSS engine will take into account these parameters (real time conditions and potential flexibility) and also some grid related KPIs (e.g. voltage/power level) towards the extraction of abnormal grid conditions and further triggering of corrective control actions.

Post Condition	Implementation of real time control actions to minimize or even eliminate the number of critical grid events in local grid.

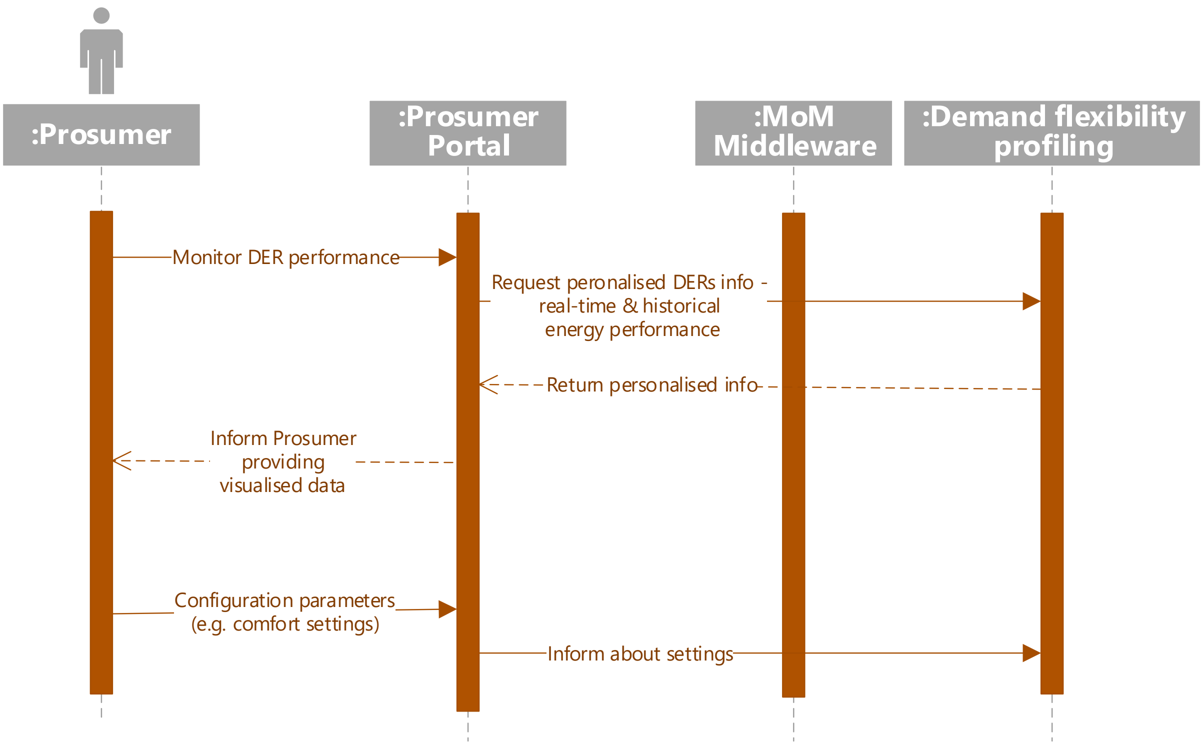


Figure 15: Use Case 11 sequence diagram

Use Case 11	Prosumers awareness and knowledge about their consumption patterns- Engagement to the overall framework
Pre – conditions	Availability of WSN in premises to monitor the current (real time) operational status. Availability of (personalized) profiling information. Active enrolment of consumers in cooperative monitoring services.
Use Case Path	The end user, through the UI, has access to personalized information. Information about (real time & historical) energy performance should be available. Complementary to the monitoring functionality, the consumer should be able to define several configuration parameters (e.g. comfort settings)
Post Condition	The end user (customer) is getting a better (personalized) understanding of his performance.

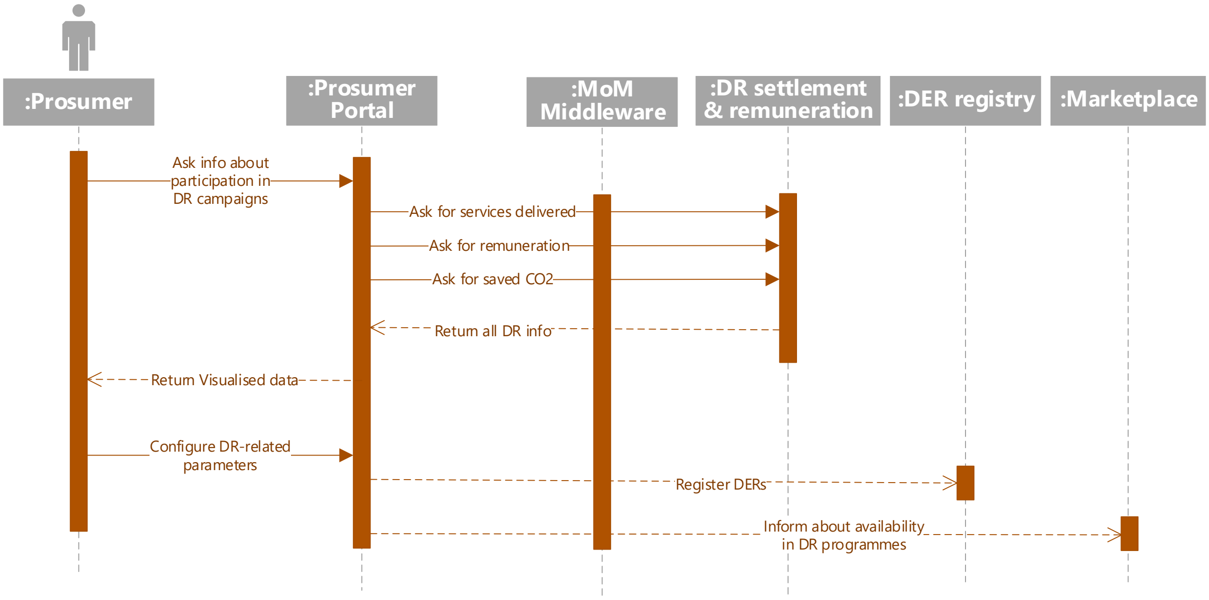


Figure 16: Use Case 11 bis sequence diagram

Use Case 11 bis	Prosumers on delivered services and resulting remuneration – Transparency of the overall framework
Pre – conditions	Availability of WSN in premises to monitor the current (real time) operational status. Availability of (personalized) profiling information. Active enrolment of consumers in cooperative market programmes.
Use Case Path	In addition, to personalized consumption information, information related to the participation in DR campaigns (or any marketplace) should be depicted through the UI. Complementary to the monitoring functionality, the consumer should be able to define several configuration parameters related to DR activities (availability in DR programmes, DER registration in a central registry etc...)
Post Condition	The end user (customer) increase his awareness about participation in DR and other market campaigns.

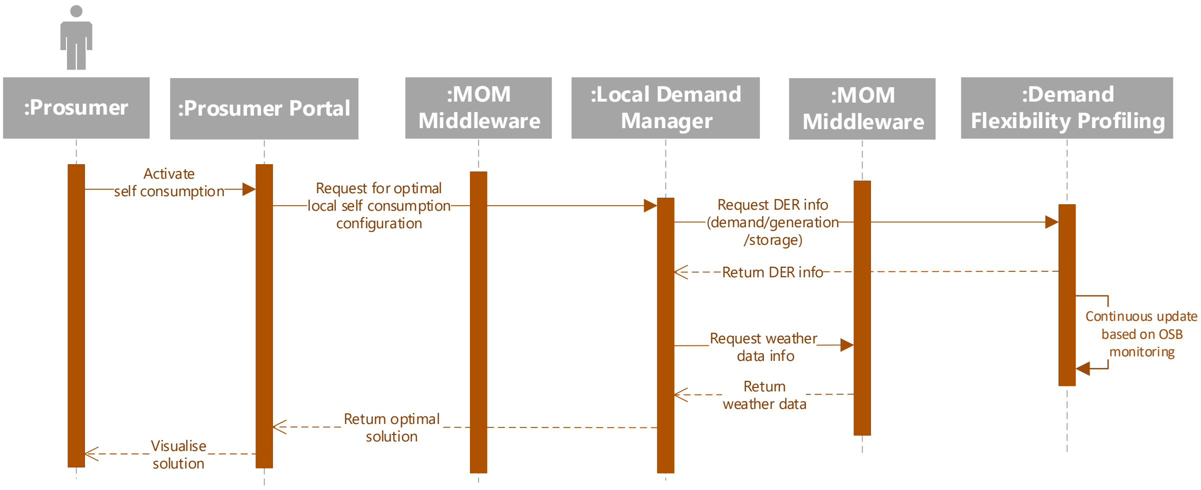


Figure 17: Use Case 12 Sequence diagram

Use Case 12	Prosumer level self-consumption to maximize energy consumption when high RES generation
Pre – conditions	Availability of local RES and storage solutions. Availability of WSN in premises to monitor the current (real time) operational status. Availability of (personalized) profiling information. Active enrolment of consumers in individual self-consumption programmes.
Use Case Path	The end use is activating the self-consumption functionality. An agent running in (local) gateway is taking into account real time and forecasted data about consumption/ storage/ generation towards the selection of local level control strategies that maximize the self-consumption level. A Local Demand Manager running in premises is the core element of this use case, incorporating the business logic for self-consumption maximization
Post Condition	Establishment of a Prosumer level self-consumption framework to ensure the maximization of local RES generation. Provision of self-consumption as an energy service from the local cooperative.

3. Modules Functional and Technical specification

For each component, we present a component view diagram providing information on component and its sub-components organisation including their interfaces. Specific attention has been paid for the layers/sub-components that communicate with external systems/components. The UML notation has been used.

3.1.1. Visualization Aggregator Toolkit

3.1.1.1. GDM View

<u>Description of design / functionality</u>

As part of the Visualization and End-User Toolkit Layer, Aggregators will receive a friendly interface for optimal portfolio management, flexibility segmentation, classification and clustering, dynamic VPP formulation, self-consumption (within microgrid boundaries) and DR strategies implementation monitoring.

This web application tool will be used by Aggregators to perform the basic operations with their portfolio of customers at three different levels: individually, as a cluster of users, and for the whole portfolio. Among other operations, Aggregators will have the capability to create new clusters according with some criteria, to manage them, and compare a user of it against the entire cluster to detect unexpected behaviours.

In addition to the management of users, this tool can also be used for monitoring the strategies and for receiving real time DR event alerts with some information about them.

Description of architecture

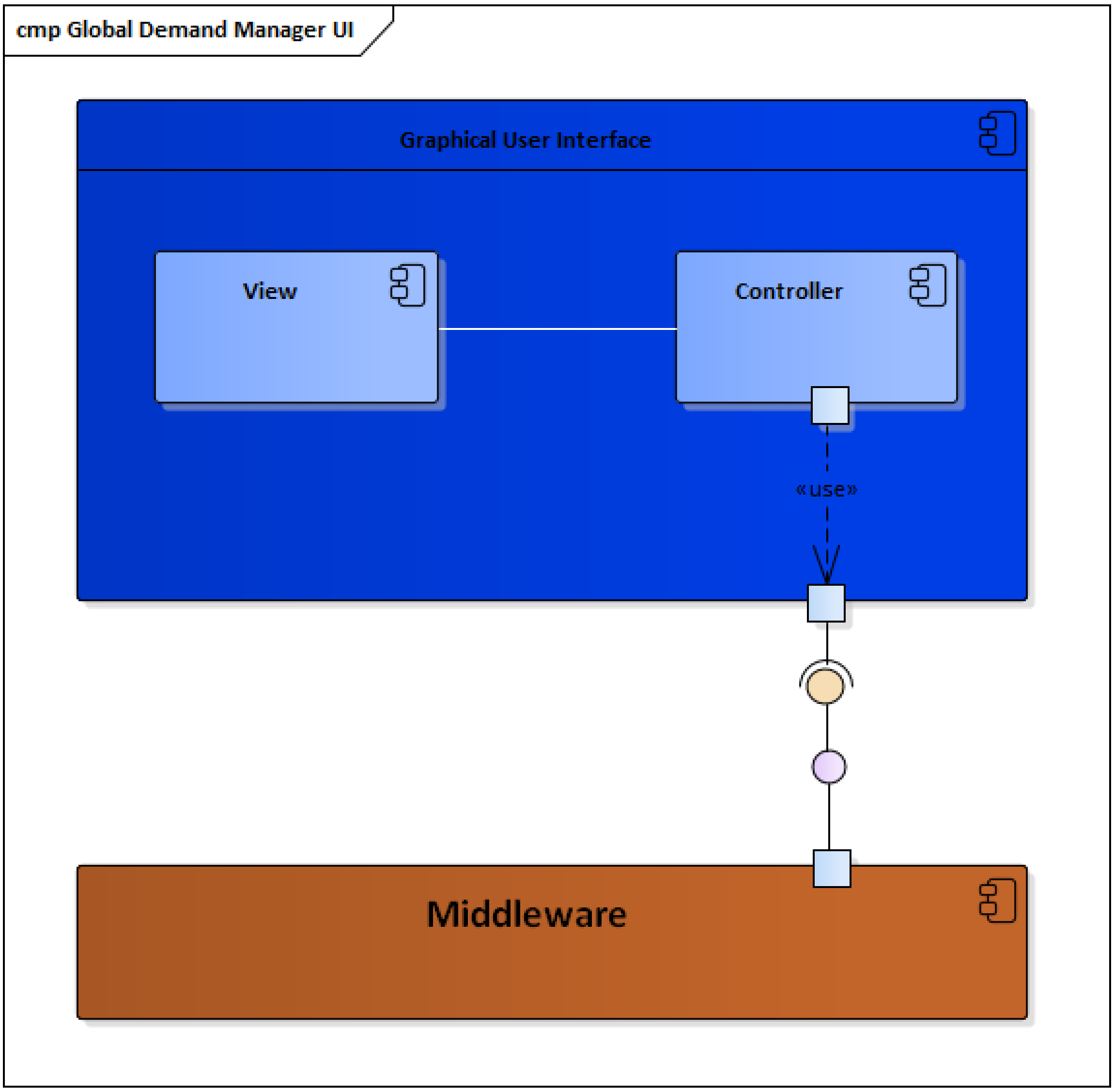


Figure 18: Global Demand Manager (GDM) UI CMP diagram

This web application tool will be used by Aggregators to perform the basic operations with their portfolio of customers at three different levels: individually, as a cluster of users, and for the whole portfolio. Among other operations, Aggregators will have the capability to create new clusters according with some criteria, to manage them, and compare a user of it against the entire cluster to detect unexpected behaviours.

In addition to the management of users, this tool can also be used for monitoring the strategies and for receiving real time DR event alerts with some information about them.

The architecture of the User Interface module will follow a Model-View-Controller approach:

- View: All the graphical components and the structure of the web site. This is the part that will see the final user and that will allow him to interact with the system.

- Controller: This will remain transparent to the user, and here will be defined all the behaviour of the application and how it responds to the user interactions.
- Model: On this case, the model won't be handled by the GUI application. Instead of that it will communicate with the **Message Oriented Middleware**, who is in charge of dealing with the other components and provide responses to every petition done via the User Interface.

Description of component interaction

Although the **GDM View** module will need functionalities from several components of the FLEXCoop architecture, all the communications will be done directly interacting with the **Message Oriented Middleware**, so that will be the only interaction with other modules of the architecture.

Description of deployment

The application will be deployed in the cloud in a Docker container, making it accessible to every user with the URL of it. It will be deployed on a 3rd party server specialized in hosting web applications, that means that the services will always be available due to the redundancy of the services and data if something goes wrong.

3.1.1.2. DER Registry View

Description of design / functionality

Energy cooperatives / Aggregators should be able to access the DER registry to facilitate DER discovery and VPP formulation and allow for successful provisioning and acquisition of specific and dedicated services from DERs.

Description of architecture

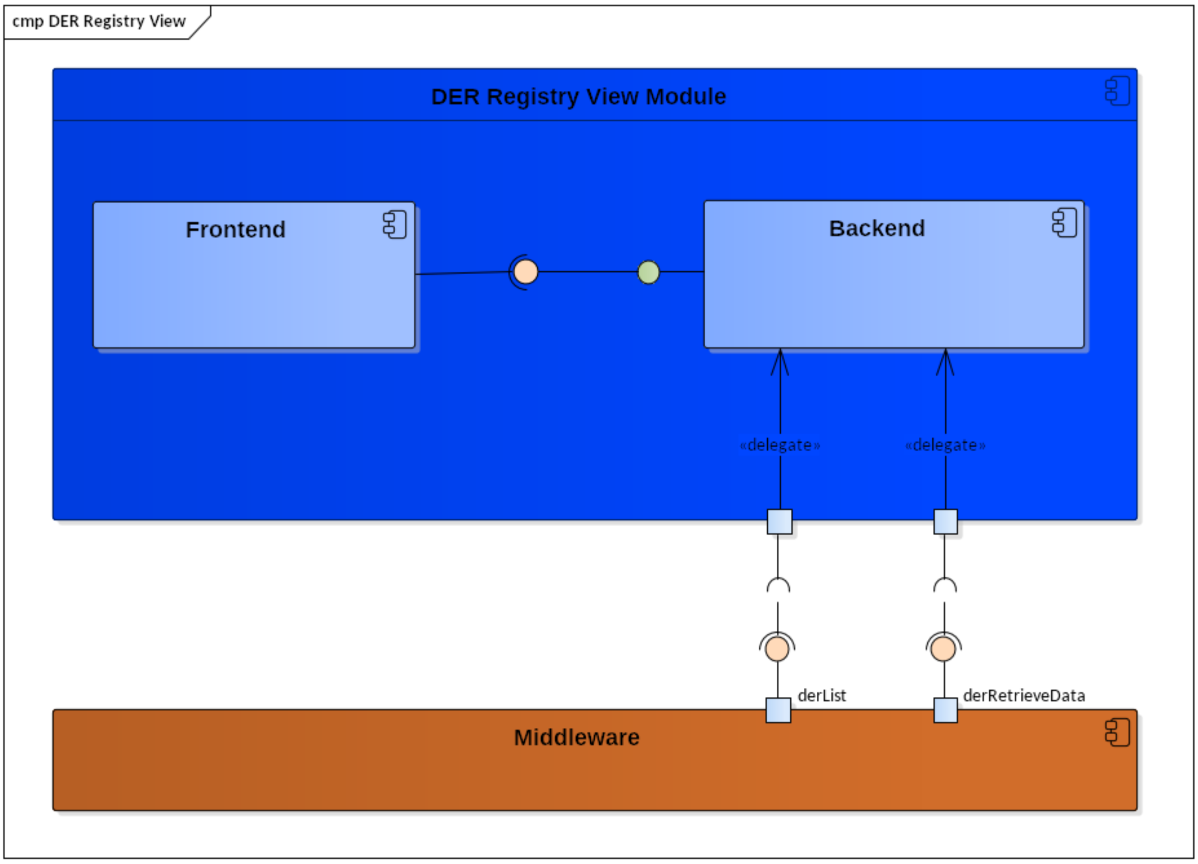


Figure 19: DER Registry View Module CMP diagram

The DER Registry View consists of two components:

- a frontend component using HTML, CSS and JavaScript in the Aggregator's web browser showing the interactive Graphical User Interface, and
- a backend component written in Java that serves the web pages for the frontend

The frontend uses web sockets for data update messages from the backend. If the frontend has to visualize complex data like graphs or charts, a JavaScript visualisation toolkit like D3 is used.

The backend implements an authentication mechanism of the frontend and backend towards the middleware, the OpenADR protocol for communication with the middleware, and a REST interface for the communication with the frontend. The backend uses the OpenADR Push mode to receive data from the message-oriented middleware.

Description of component interaction

If the Aggregator opens the DER Registry View frontend, he has to authenticate to the middleware via the backend. The current list of DERs (with its stored properties) and their Aggregator contracts are queried by the backend via OpenADR over the middleware from the

DER Registry, prepared for visualisation, and sent to the frontend. Changes to the list in the registry are updated.

Description of deployment

The DER Registry View backend is deployed as a Java 7 runtime in a Docker container.

3.1.1.3. Open Marketplace View

Description of design / functionality

Complementary to the Aggregator DER Registry view, the Aggregator marketplace view will enable Aggregators to publish their offers to attract consumers and engage them in demand response services. Alternative contract types and remuneration methods (both for standby and activated DERs) will be offered to Prosumers, who will be given the opportunity to discover and select DERs with desired characteristics to address their flexibility requirements, to negotiate and customize their contractual relationship with Aggregator. Once an agreement is established, then the Aggregator is provided access to the respective DERs published in the FLEXCoop registry

Description of architecture

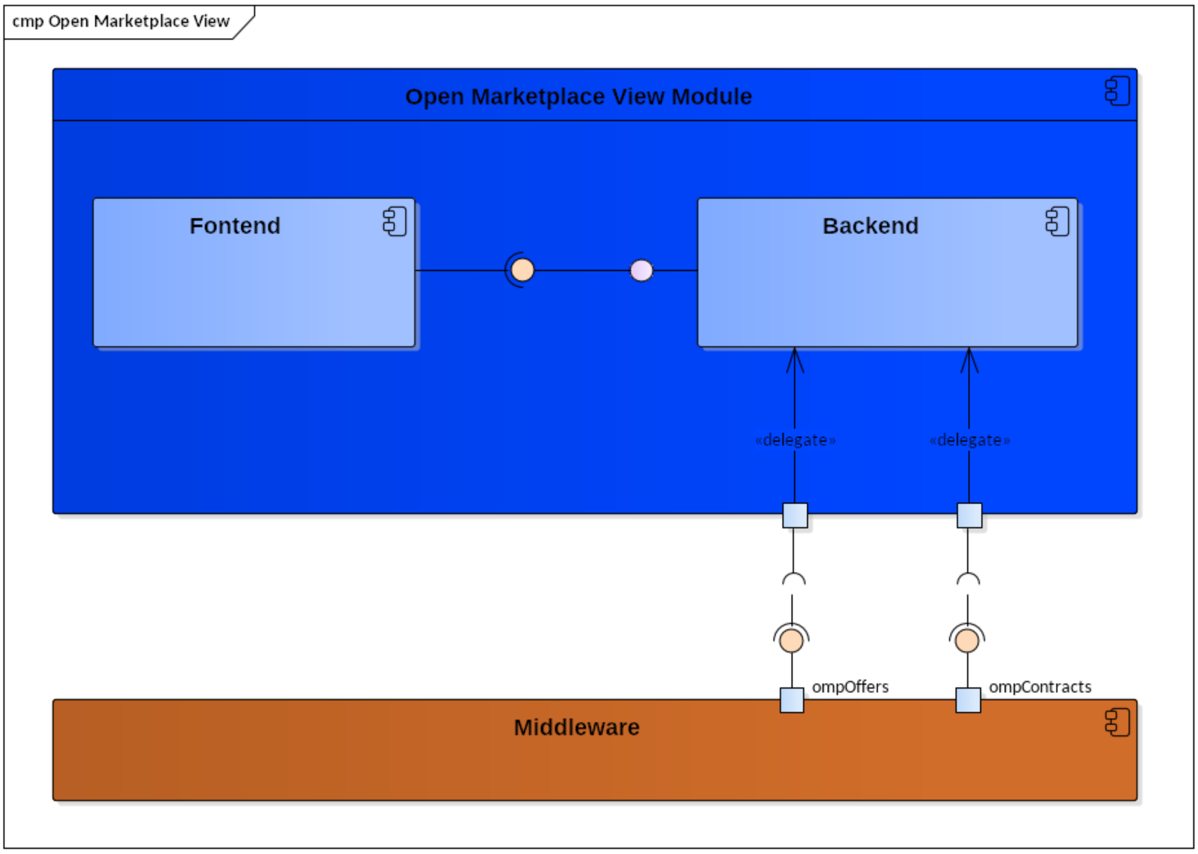


Figure 20: Open Marketplace View CMP diagram

Similar to DER Registry View, the Marketplace View consists of two components:

- a frontend component using HTML, CSS and JavaScript in the Aggregator's web browser showing the interactive Graphical User Interface, and
- a backend component written in Java that serves the web pages and marketplace data for the frontend

The frontend uses web sockets for data update messages from the backend. If the frontend has to visualize complex data like graphs or charts, a JavaScript visualisation toolkit like D3 is used.

The backend implements an authentication mechanism of the frontend and backend towards the middleware, the OpenADR protocol for communication with the middleware, and a REST interface for the communication with the frontend. The backend uses the OpenADR Push mode to receive data from the message-oriented middleware.

Description of component interaction

If the Aggregator opens the Open Marketplace View frontend, he has to authenticate to the middleware via the backend. Aggregators can publish offers and negotiate them with individual customers via the Open Marketplace. The DER owners can search for offers, start a negotiation about a contract and sign the contract. The offers and contracts of Aggregators and DER owners are stored in a database. The DR Settlement and Remuneration module is informed about signed contracts.

Description of deployment

The Open Marketplace View backend is deployed as a Java 7 runtime in a Docker container.

3.1.2. Visualization Prosumer Toolkit

Description of design / functionality

The role of this software component is to act as the interface application for Prosumers to enable efficient monitoring of real-time demand data and Demand Response events triggered by Aggregators towards their active enrolment in the FLEXCoop environment. The main features and functionalities supported by the application are in line with the different business scenarios as examined in the project, namely: (a) enriched visualization (energy demand monitoring and analytics, demand forecasts as extracted from analytics services, increasing awareness around energy market), (b) DR & self-consumption campaigns participation (notifications of demand response events triggered by Aggregators and verification of compliance with contracts), (c) Marketplace Participation (financial and economic management of contractual agreements).

The latest is actually the main innovation of the tool as will enable the active participation of end users/consumers in different energy markets following negotiation towards the establishment of contractual agreements with the different market operators (DR Aggregators in FLEXCoop).

Description of architecture

The definition of Prosumer Toolkit internal architecture is in line with the functional analysis and is presented in the following schema. The component diagram for this software module is presented.

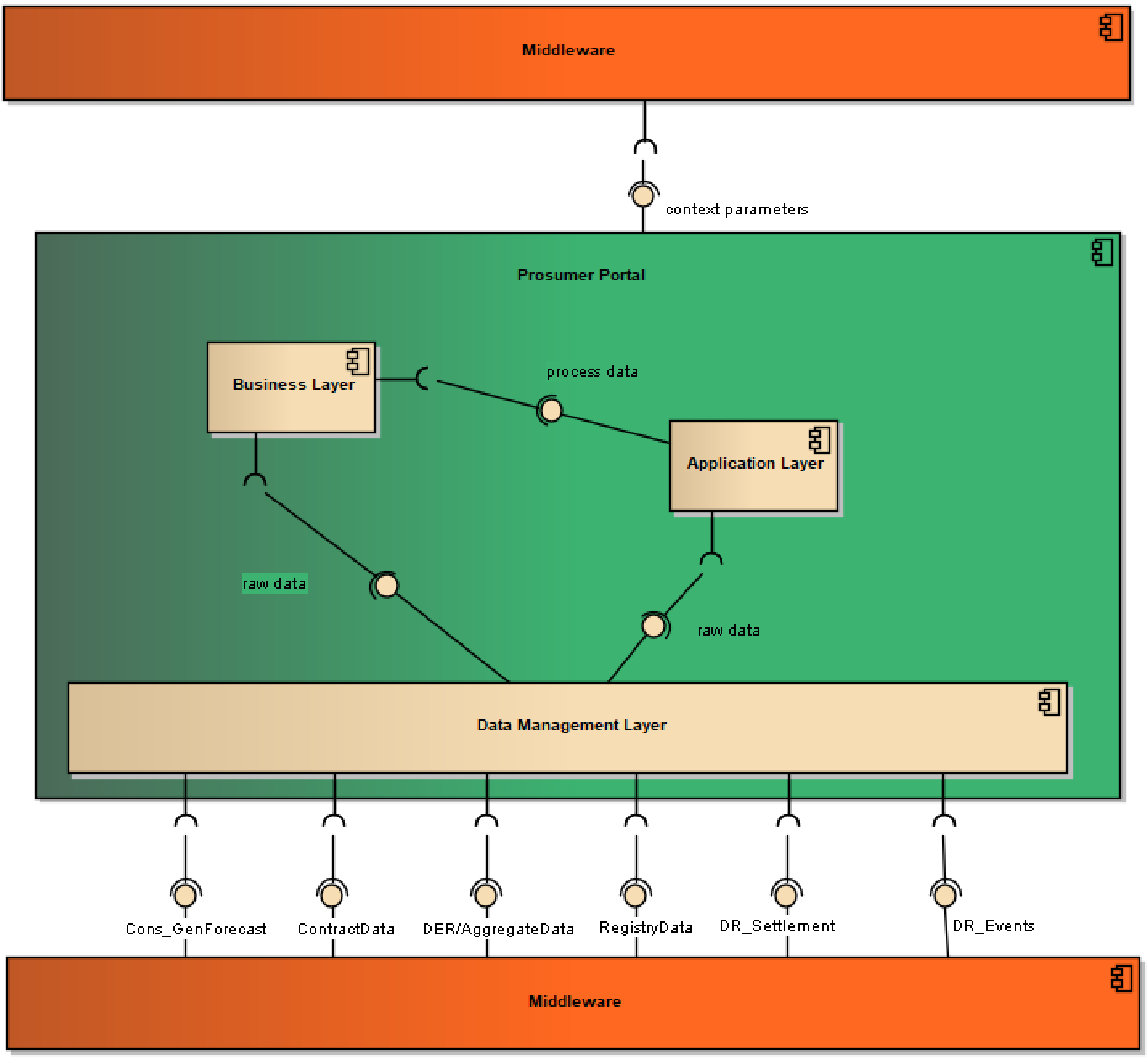


Table 3: Visual Prosumer Toolkit CMP diagram

A typical 3-tier architecture is considered and the different modules are briefly described:

- **Data Management Layer**: The main role of this module is to define the wrappers for interfacing with FLEXCoop Middleware. While the Prosumer Toolkit will not store the dynamic logs of information required for visualization, a minimum of configuration/settings information along with business/market related parameters will be stored internally in the Prosumer application data management layer.
- **Application layer**: This is the analytics layer of the application to support the different functionalities as defined in the list of requirements:

    - Enriched Visualization and Awareness: (near) real time information visualization, historical Reports visualization, DER registry information visualization
    - Demand Response & self-consumption campaigns management toward the active enrolment of end users in energy market schemas as examined in the project
    - Marketplace participation: contractual agreements negotiation and management to ensure the active enrolment of end users in energy markets
- **Business layer**: This is the front-end layer for rendering and presentation of information to the associated end user devices.

It is evident that the architecture of the Prosumer portal application will rely on the Model View Controller (MVC) principles, which is commonly used for developing user interfaces. The reason for this decision was to ensure the modularity of the application, by separating the data management from the analytics and visualization layers.

Therefore, in the proposed software architecture RESTful services to external components will be defined in order to exchange the associated information (via Middleware). A PostgreSQL DB will be configured to store local model parameters and Django framework (supporting the MVC pattern for software development) will be adopted for the development of the application layer (as a Python application) coupled with state-of-the-art JavaScript libraries (and HTML5 principles) for enriched data visualization.

A more detailed presentation of the overall framework and the details about the development process will be reported in FLEXCoop deliverable D6.3 – Prosumer Application – Preliminary Version.

Description of component interaction

Towards presenting through the Prosumer application the aforementioned information, interfaces with FLEXCoop Message Oriented Middleware will be defined (on the basis of FLEXCoop data model definition) while the requested information for visualization is available from different FLEXCoop software components:

- Demand Flexibility Profiling Module & DER registry for retrieving home environment metering and sensing data
- DER Management System enhanced with functionality about consumption& generation forecasting
- Global Demand Manager & DR settlement and remuneration module; subscribing to receive DR events and asset related information and further requesting information about DR participation (business related parameter)
- DER Registry for initial registration of a user with an asset and further searching registered DERs and their characteristics (including availability to participate in the m market)
- Open Marketplace for Market participation; setting contracts and negotiation contractual agreement

As mentioned above, RESTful services to Message Oriented Middleware will be exposed to support the functionality of the module. Additionally, a client to the FLEXCoop publish-subscribe mechanism (i.e. AMQP, MQTT) will be also supported for communication with Message Oriented Middleware. The detailed specifications for the aforementioned interfaces will be defined in WP4 (Interfaces definition) though an overview of these interfaces is part of the sequence diagram analysis which presented in this document.

Description of deployment

Taking into account the consortium approach for a non-centralized deployment, the aim is to deploy the service as a cloud application ensuring that way its reliability and stability. The deployment of the Visualization Prosumer Toolkit will be hosted in a WSGI server (for hosting Python software apps) in association with an Apache or NGINX which is used as the web server.

The complexity of the software system and the size of each pilot are considered for the definition of the hardware requirements towards the deployment of the Visualization Prosumer Toolkit. An indicative set up with an 8-core CPU, 8GB of RAM, 80GB storage and a 64-bit Linux OS is prescribed though a more specific analysis of deployment requirement will be defined in FLEXCoop D6. 3 – Prosumer Application.

3.1.3. Open Marketplace

Description of design / functionality

An **Open Marketplace** is the software layer to facilitate smooth negotiation of Demand Response contractual relations and the establishment of transparent DR activation and remuneration rules between the two sides: Aggregators and Prosumers. The **Marketplace** allow Aggregators to publish standardized contract templates and negotiate them on-the-fly with individual Prosumers. Through the marketplace Aggregators will be given the opportunity to select DERs with desired characteristics to address their flexibility requirements (whether they are part or not of their portfolio) and negotiate DR agreements with the DER owners/ Prosumers.

- Facilitate Prosumer in order to participate in the market
- Aggregators can publish offers
- Contracts for Prosumers
- Prosumer integration via Open Smart Box (OSB)

Description of architecture

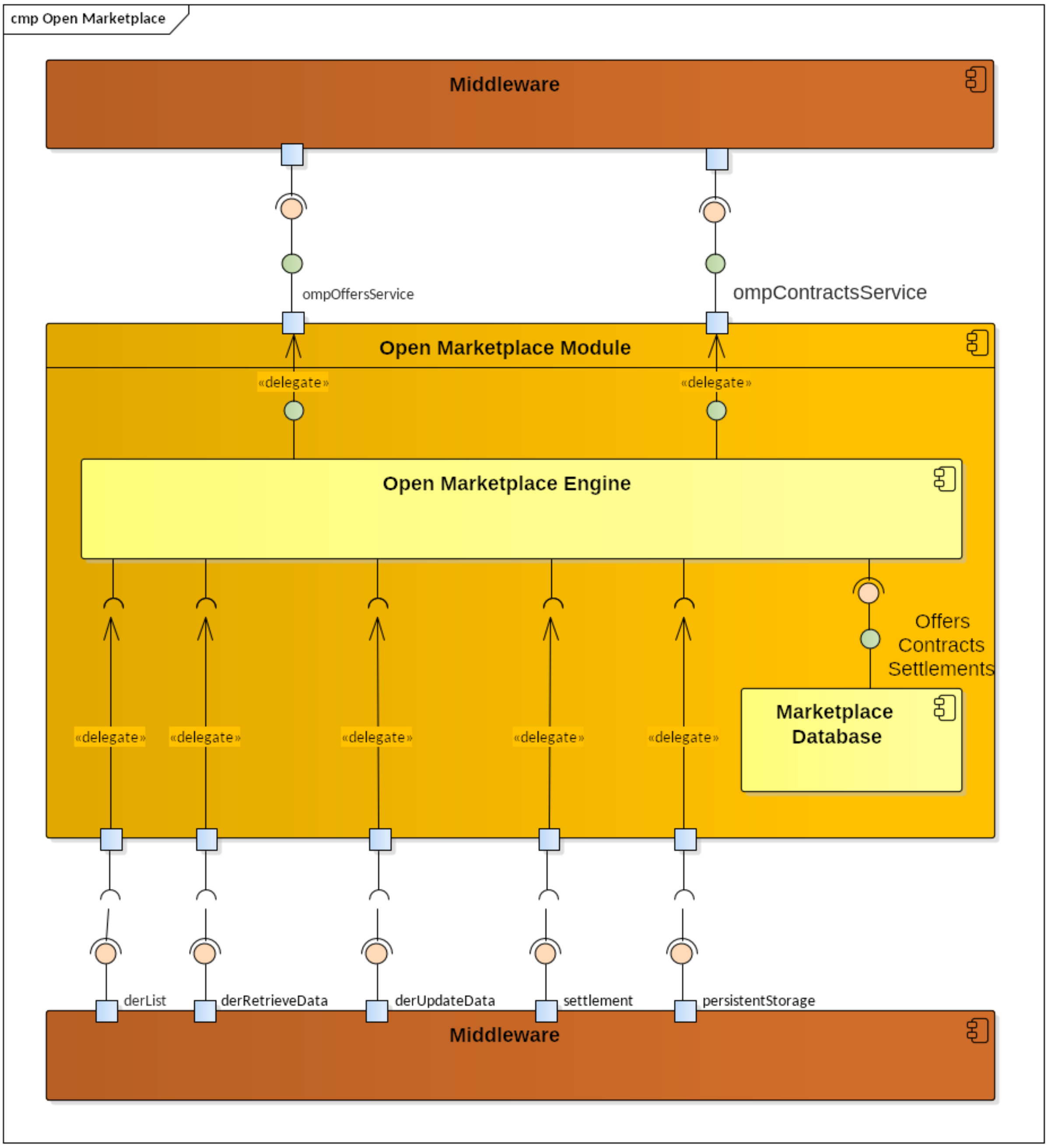


Figure 21: Open Marketplace CMP diagram

The **Marketplace** is a backend component. This means it will not provide an UI by itself but provides the data to different front-end applications as well as other components that make use of the data of the **Marketplace.** It is planned to implement the **DER Registry** with Python3 as main programming language. The architecture will follow the MVP pattern. If possible, code specific to the data of the **Marketplace** will be implemented in a library to be reusable for other components. It is planned to use either the Django[4] or flask[5] framework to implement the necessary APIs.

The **Marketplace** will use persistent storage function provided by the **Middleware** to store the following information:

- Details of all contracts, offers and settlements
- If not provided by other components, information on the market place participants

The **Marketplace** will use an in memory DB like redis[6] to hold states and other temporary data. It may also be used to cache data before pushing it to the persistent API provided by the Middleware. The communication with the **Middleware** will be done by using OpenADR.

Description of component interaction

The Marketplace interacts with the **DER Registry** via the Middleware via OpenADR.

Therefore the Marketplace will implement an OpenADR VEN

Description of deployment

The deployment will be done using Python's packaging mechanism. To manage the configuration in the live deployment either Docker[7] or Ansible[8] can be used. This will be aligned with other components that have dependencies that are more complex.

3.1.4. Local Demand Manager

Description of design / functionality

This is the DSS layer at building level. The role of this component is to monitor in real time the context and operational conditions, and by combining information coming from the Holistic Context-Aware Flexibility framework to select the optimal DR control strategies, as triggered by the Aggregator.

Local optimization will be a dynamic process since the algorithmic framework will enable continuous monitoring of the DR event evolution to break down global flexibility requirements communicated by the Aggregator into individual flexibility that can be offered per load at the consumer side and dispatch the corresponding control signals/ actions to the loads (following a human-centric approach preserving comfort and indoor quality). In addition, the role of the component is to continuously monitor the progress of control actions to identify overrides of the implemented strategies or failures to respond and automatically revise the initially defined

---

[4] https://www.djangoproject.com/

[5] http://flask.pocoo.org/

[6] https://redis.io/

[7] https://docker.com

[8] https://www.ansible.com/

strategies so as to achieve the provision of the anticipated amounts of flexibility even at local level.

<u>Description of architecture</u>

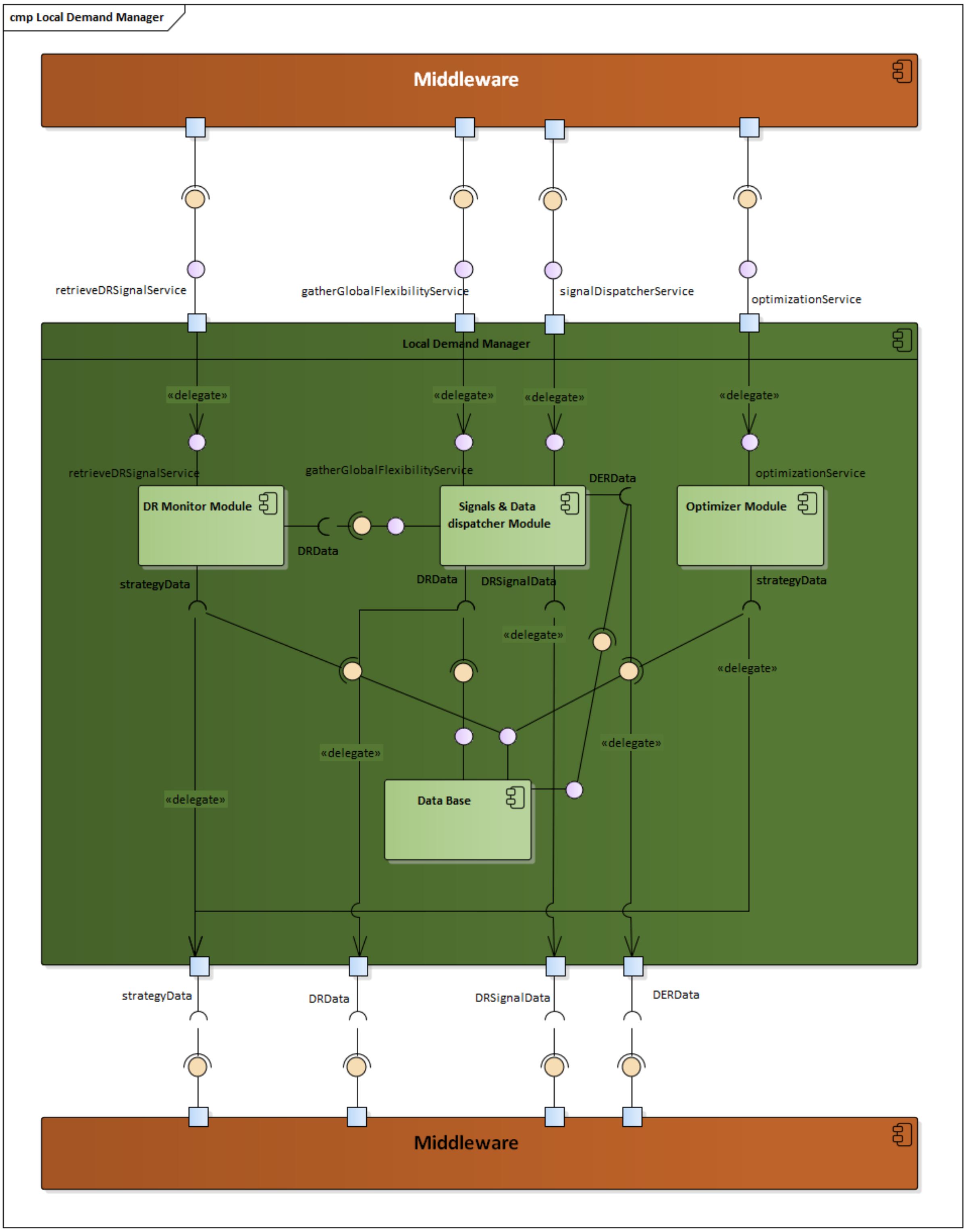


Figure 22: Local Demand Manager CMP diagram

The **Local Demand Manager** will contain 3 modules, running in parallel each time any of their functionalities is required.

The **DR Monitor** submodule will work as a background process constantly analysing every DR data received and revising the initially defined strategies if it detects any failure or overrides.

The **Signals & Data dispatcher** submodule will be in charge of two tasks, a) the first task will be breaking down global flexibility requirements into individual flexibility and dispatching DR signals, b) the second one will be providing data after every DR signal received, or periodically for keeping the information on the DER Registry updated.

Lastly, the **Optimizer** will be in charge of the optimization process for selecting at every moment the best strategies to be applied.

A storage system will be used for storing and retrieving needed information for accomplishing the functionalities of the three submodules.

The module will be developed using Node.js technology, making it accessible as a cloud service by using HTTP REST interfaces. Having the services allocated in the cloud, by a third-party server, guarantees the access to those services.

Description of component interaction

Communication between **Local Demand Manager** and the rest of components will be done via the **Message Oriented Middleware**.

The **DR Monitor** submodule will need interaction with the **Global Demand Manager** for reacting with every dispatch control signal received and checking if everything goes as expected. As a reaction to every DR event, this module will send that DR event information to the **Signals & Data dispatcher** module, so it can react and update the Marketplace data and communicate to the **Global Demand Manager** the response to that control signal.

In addition to the aforementioned, **Signals & Data dispatcher** submodule will periodically communicate with the **DER Registry** for sending it the data and keeping it updated.

On the other hand, **Optimizer** will need to gather information from the **Demand Flexibility Profiling** for getting the flexibility information to calculate the optimizations.

Finally, Prosumers could interact with the functionalities of this module interacting with the **Prosumer Portal**, which is the final User Interface available to this kind of users.

Description of deployment

This module will be deployed on a Docker in the cloud, offering API interfaces for accessing to some of its functionalities It will be deployed on a 3rd party server specialized in hosting web applications, that means that the services will always available due to the redundancy of the services and data if something goes wrong.

3.1.5. Global Demand Manager

Description of design / functionality

The role of this module is to develop the FLEXCoop **Global Demand Manager (GDM)**, a module to continuously analyse demand/ storage flexibility, along with signals coming from the DSO side and on-the-fly decide about the optimal configuration of demand-based dynamic Virtual Power Plants (VPPs) to timely respond and provide the required flexibility to the grid.

The **Global Demand Manager** will be responsible for dispatching appropriate fully automated control signals (Direct Load Control on the basis of standardized Service Level Agreements) to Local Demand Managers (following an agent-based approach) with the aim to utilize their flexibility, following a bilateral negotiation process. The Global Demand Manager will continuously monitor the evolution of each Demand Response event, to identify overrides of the deployed strategies or failures to respond and will revise the initially defined strategies so as to achieve the provision of the anticipated amounts of flexibility (dynamic VPP re-formulation) and optimize business functions and energy transactions of all stakeholders involved.

Description of architecture

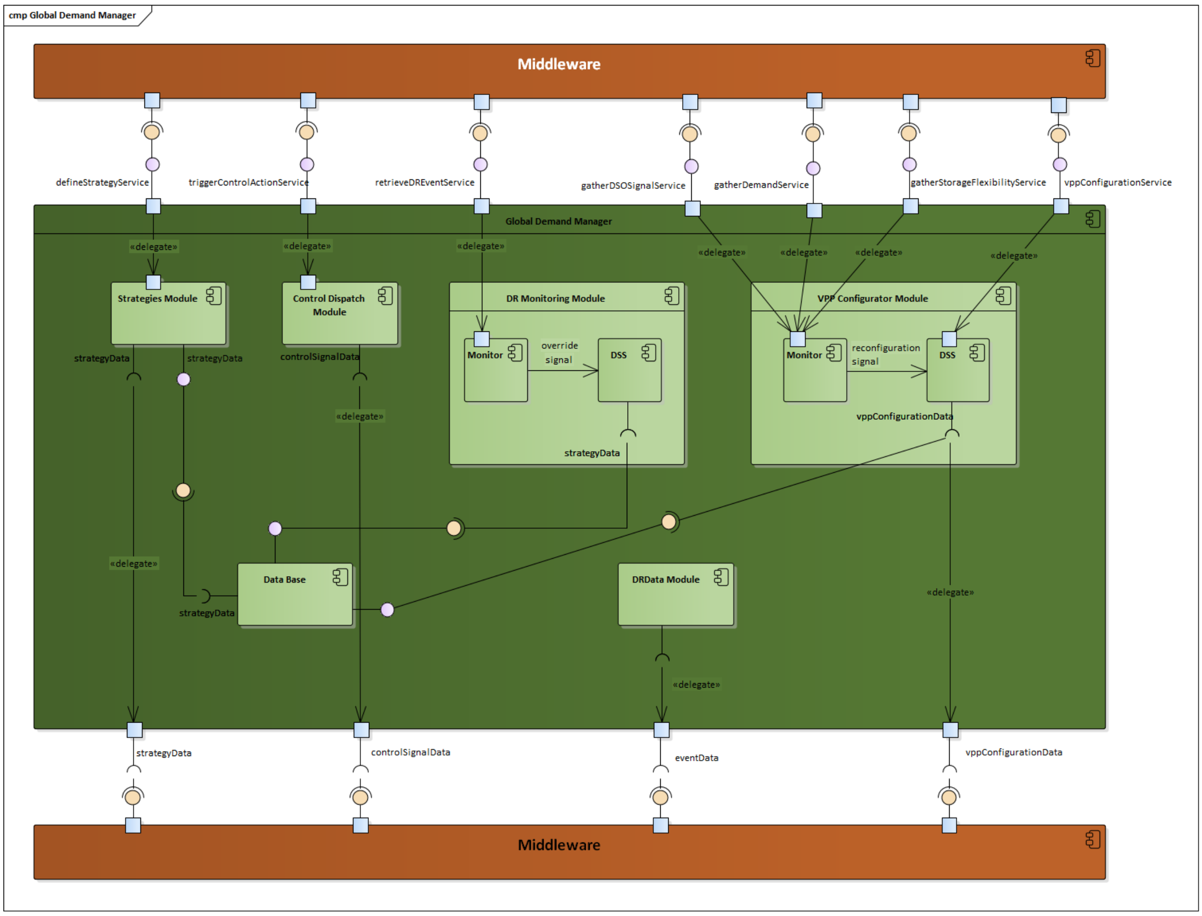


Figure 23: Global Demand Manager CMP diagram

The **Global Demand Manager** will be based in three main submodules, each one of them in charge of a different set of related tasks.

The **Virtual Power Plant (VPP) Configurator** submodule will contain two different components: the **Monitor** and the **Decision Support System (DSSy)**. The **Monitor** could be seen as a background process constantly receiving and analysing the signals and data coming from other modules, and triggering functionalities from the **DSSy** module only if they are needed. This module will ensure the optimal configuration of the VPPs at every moment.

The **Demand Response Monitoring** submodule will also contain the same two different aforementioned components, but on this case they will be in charge of ensuring the optimal strategy to be used at every moment by analysing every Demand Response event.

The third submodule, the **Control Dispatch Module**, will be in charge of dispatching control signals to the Local Demand Managers.

A fourth component will be used by those three modules for storing and consulting the needed data (the **Data Base**).

Due to some functionalities from the three components and from the storage system will be need to be accessible to some users, an interface will be offered to guarantee access to those services. The implementation of the whole module will be done using *Node.js* technology, offering some of their functionalities as cloud services accessible by using the interfaces of a HTTP REST API. The rest of functionalities will be automatic processes that will be launched at a concrete date (i.e. the recalculation of VPPs).

<u>Description of component interaction</u>

The Global Demand Manager will need to communicate with some others components of the FLEXCoop architecture for being able to realize its tasks.

Internally to the **VPP Configurator**, the Monitor process will communicate with the DSSy process if it detects any change that affects to the current VPP configuration, so the DSSy will have to reconfigure it only when the Monitor notifies it to do so. This monitor process needs the following signals coming from other FLEXCoop modules through Middleware:

- Demand Response event data from the **Local Demand Manager**.
- DSO signals coming from the **Message Oriented Middleware**.
- Flexibility coming from the **Demand Flexibility Profile**.

Meanwhile, the **Demand Response Monitoring** module will have a similar behaviour than the already commented module. The Monitoring process will be internally connected with the DSSy process, triggering it only when it is detected that the strategies must be revised. The input from this module will come from the Demand Response events triggered by the **Local Demand Managers**.

Lastly, the **Control Dispatch Module** will be sending control signals to the **Local Demand Managers** when a trigger action is required.

Some functionalities will be offered to the Aggregator user, who will be able to use them interacting with the **Aggregator UI**. The communication between this **Aggregator UI** (or any other module of the FLEXCoop solution) and the **Global Demand Manager** module will be done via the **Message Oriented Middleware**.

Description of deployment

All the services of this module will be packed on a single Node.js application deployed on a Docker in the cloud, making some of them accessible by using the public interfaces of the API. It will be deployed on a 3rd party server specialized in hosting web applications, that means that the services will always be available due to the redundancy of the services and data if something goes wrong.

3.1.6. Flexibility forecasting, segmentation and aggregation

Description of design / functionality

This component is responsible for the multidimensional analysis, correlation and efficient management of Prosumer profiles, flexibility and response capacity to DR signals. This component will enable the dynamic and spatio-temporal segmentation, classification and clustering of consumer flexibility for the selection of appropriate aggregated demand side-based VPPs to provide specific DR functions (considering flexibility profiles and their suitability to provide different grid services, e.g. balancing, frequency response or voltage regulation). The module will constantly interact with the DER repository of the Aggregator to discover the best DERs for each service request. The results from this analysis will further facilitate the implementation of DR strategies from the Global Demand Manager.

This module will embed all functionalities pertaining to the tool chain for collecting demand flexibility profiles of consumers, managing them in order to establish optimal VPP composition for the delivery of grid services. An indicative list of features includes: visualization, clustering based on historical data, trend analysis, insights extraction and further support on VPP management.

Its main innovation will be that rather than matching the demand profile of a specific building to a generic class and then extracting demand flexibility estimations, the FLEXCoop tools will cluster and segment buildings based on their actual, locally estimated demand flexibility incorporating detailed information about building infrastructure and use by occupants.

Description of architecture

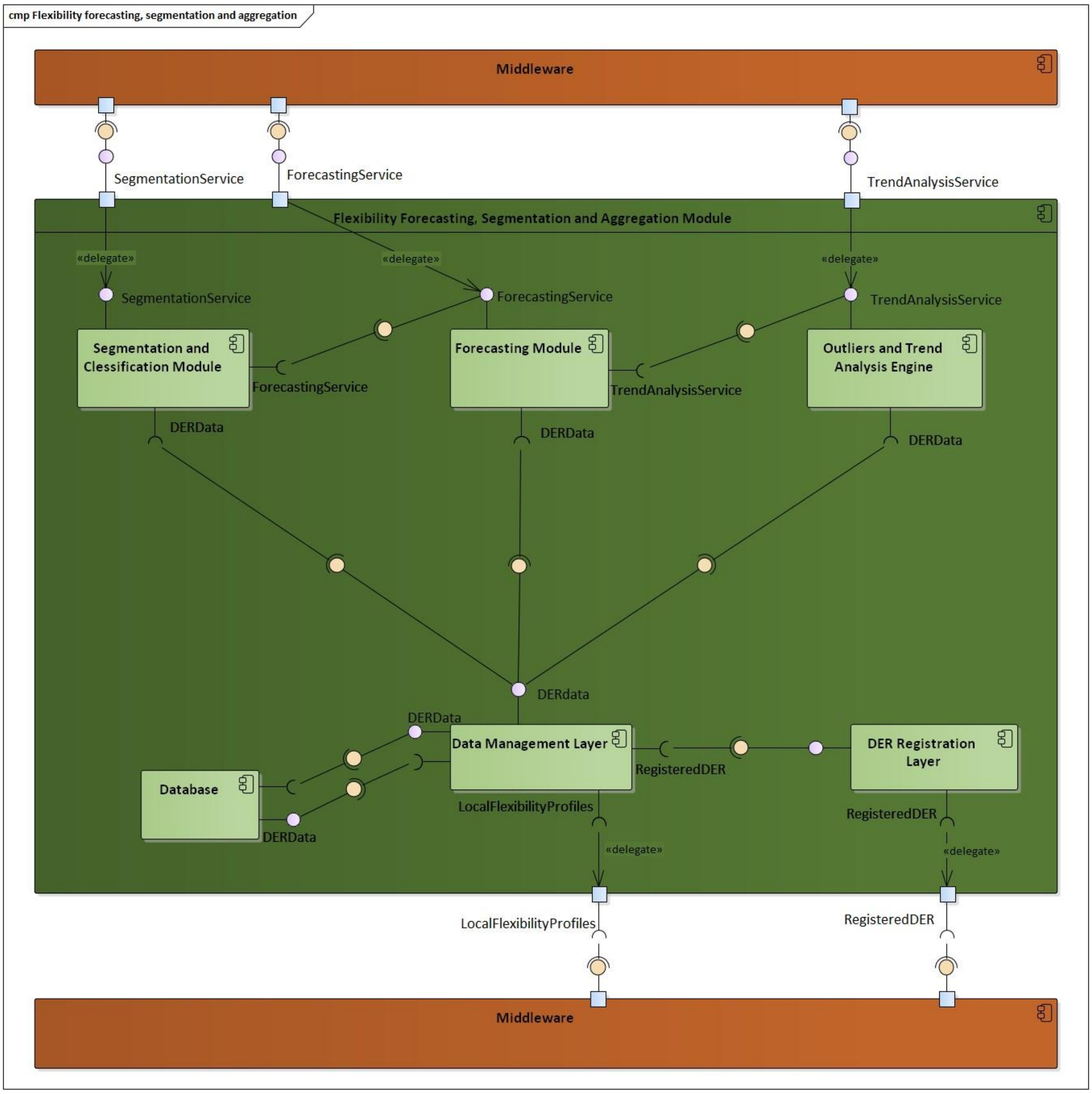


Figure 24: Flexibility forecasting, segmentation and aggregation CMP diagram

The Flexibility forecasting, segmentation and aggregation module consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent DER related data
- The DER registration layer that maintains the updated list of registered DERs and
- The different modules required to provide the actual services of the Flexibility forecasting, segmentation and aggregation module i.e. segmentation and classification, forecasting, trend analysis, etc.

The development of the core application is planned to be based on Java and the whole framework to consist a J2EE application. A MySQL database is planned to be installed to manage model parameters, information and data retrieved from the Message Oriented Middleware.

The software architecture will rely on the MVC (Model View Controller) pattern, which is commonly used for developing user interfaces. In principle, it separates out an application into three interconnected layers promoting modularity and ease of collaboration as well as reusability. To this end, the Spring MVC framework will be used providing the aforementioned architecture pattern as well as ready components towards developing a flexible and loosely coupled application. Hibernate ORM (version 5) is the java framework proposed to be used for mapping the object-oriented models to the relational database. For the ***View layer*** of the MVC architecture we will use JSP (Java Server Pages) to generate dynamically web pages based on HTML in conjunction with the JavaScript framework. In addition, multiple JavaScript libraries such as d3.js, GDC (Google Visualization Console) visualization API will be used to offer visual analytics on forecasting, segmentation and aggregation of consumer flexibility. Analytics consist also of a business logic layer, enhanced with the WEKA library, which is a collection of machine learning algorithms for data mining tasks.

The proposed software architecture ensures the modularity of the application. This way, modifications of any part of the application will not affect the overall development process. Furthermore, Spring MVC supports the design, development and exposure of RESTful services which can be used for the communication with external components, as further described below.

Description of component interaction

In the proposed software architecture, RESTful services to external components can be exposed to support the functionality of the flexibility forecasting, segmentation and aggregation module. Additionally, a publish-subscribe mechanism (i.e. AMQP, MQTT) for communication with 3rd parties will be supported. This way, the flexibility forecasting, segmentation and aggregation module will interface through the Message Oriented Middleware:

- With Aggregator UI for data visualisation
- With DER registry and Open marketplace for searching registered DERs and their characteristics (including contractual availability)
- With the Demand Flexibility Profiling module for providing DER-related historical data (e.g. consumption)
- With the Global Demand Manager to facilitate the real time DSSy optimization and selection of Prosumers to participate in DR campaigns (VPP Configuration, DR monitoring and dispatch module)

Description of deployment

The objective is to deploy flexibility forecasting, segmentation and aggregation module as a cloud application, ensuring that way its reliability and stability. The deployment of the flexibility forecasting, segmentation and aggregation framework will be hosted in an Apache Tomcat server (open source implementation of the Java Servlet, JavaServer Pages and Java

WebSocket technologies) operating as an application server. In addition, Apache or NGINX server is planned to be used as a web server.

The minimum hardware requirements for the deployment of the flexibility forecasting, segmentation and aggregation module, as it has been estimated so far, are an 8-core XEON (or equivalent) CPU, 32GB of RAM, 20TB hard disk and a 64-bit Linux OS.

3.1.7. DR Settlement / Remuneration

<u>Description of design / functionality</u>

Aggregators will be offered a DR Settlement and Remuneration (DRSR) application to the realization of three objectives: a) to perform an objective and accurate baselining of current energy performance/ consumption of prosumers and normalize it against varying occupancy patterns, energy uses and climatic conditions, b) to measure and verify the flexibility (and relevant economics) that has been delivered by prosumers during a DR event, and, c) to estimate the short/mid-term impact of project activities through a socio-economic analysis. The role of the component is complementary to the role of VPP Configurator, DR Monitoring and Control Dispatch Module towards the a-posteriori remuneration of consumers for participation in Demand Response Campaigns.

Description of architecture

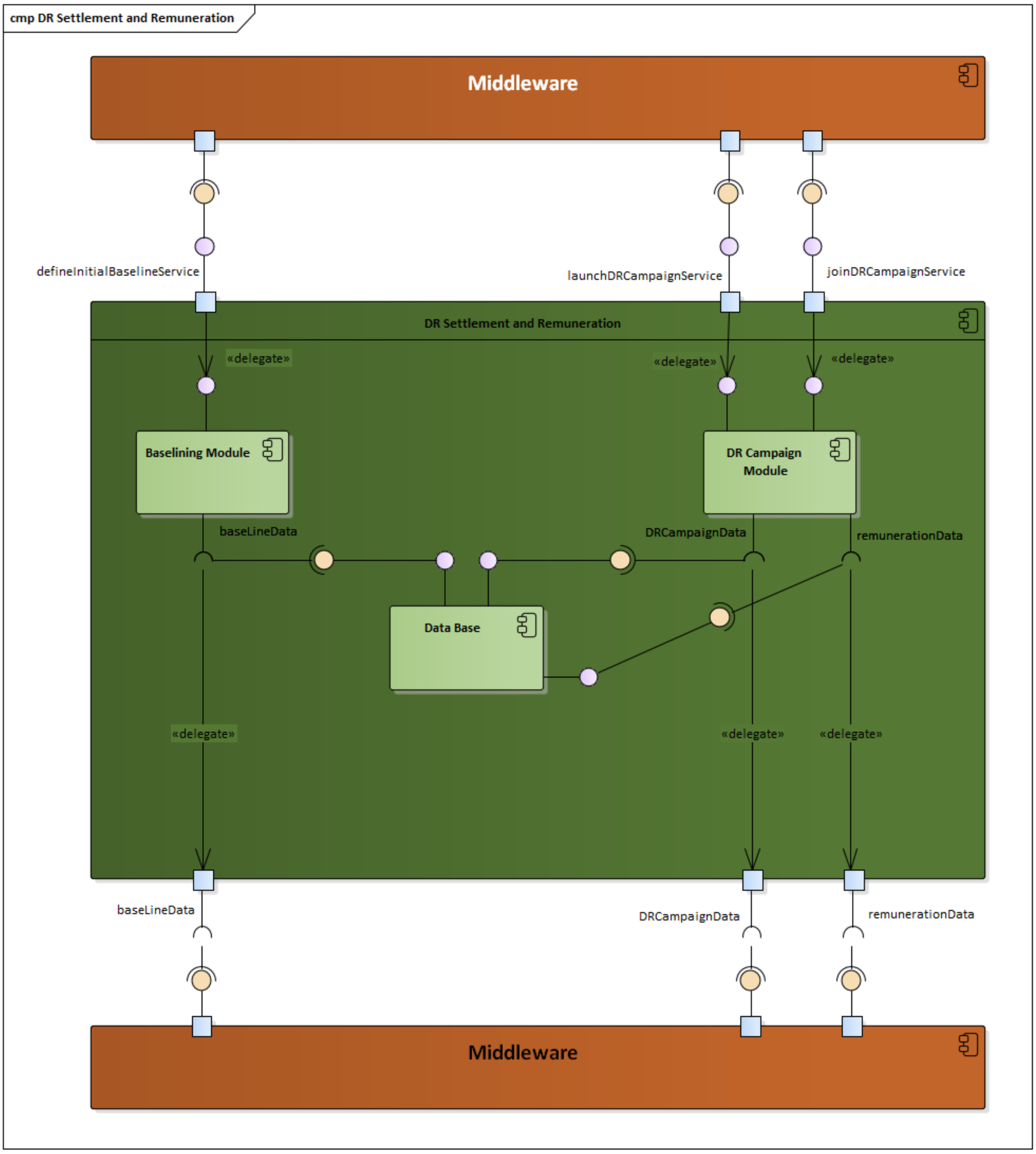


Figure 25: DR Settlement and Remuneration CMP Diagram

The **DR Settlement and Remuneration** module will be split in two main parts. The **Baselining** sub-module will be focused in the calculation of the baselines of the Prosumers, defining the initial baseline for the new clients and updating it at the end of the day (or when required). If a new Prosumer with no historical data joins the system, it won't be possible to create an initial baseline until having some information about him/her.

The second sub-module (**DR Campaign**) will be the entry point for every new Demand Response campaign launched by the Aggregators, and all the functionalities related with DR campaigns and DR events occurring during its duration. For the Prosumers, this module will be used for getting all the information about the ongoing DR campaigns (and joining them if it is desired) and the historical data of all the campaigns where they participated. Lastly, at the end of the campaign this sub-module will analyse the contractual agreements with the Prosumers and remunerate them if required.

Both modules will be supported by a storage system (i.e. a Data Base) containing historical data needed for the calculation of the baselines and keeping them updated. The different information about the demand response campaigns and the behaviour of the Prosumers will also be here.

The implementation of this component will be done using *Node.js* technology, offering some of their functionalities as cloud services accessible by using the interfaces of a HTTP REST API. The rest of functionalities will be automatic processes that will be launched at a concrete date (i.e. keeping updated the baselines) or depending of the finalization of other events (i.e. the remuneration task).

Description of component interaction

The module itself needs to communicate with the **Global Demand Manager** and the **Marketplace** in order to being able to calculate the remuneration for the different Prosumers involved in the campaigns. This communication won't be direct, it will be done interacting with the **Message Oriented Middleware**, and this module will be in charge of delivering the proper messages to all the affected modules.

The public services will be accessible for the Aggregators and Prosumers via the UI tools (the **Aggregator UI** for Aggregators, and the **Prosumer Portal** for Prosumers), being the Message Oriented Middleware in charge of communicating every component among them.

Description of deployment

All the services of both sub-modules will be packed on a single Node.js application deployed on a Docker in the cloud, making some of the accessible by using the public interfaces of the API. The final application of this module will be hosted in a 3rd party server in the cloud, guarantying on this way the secure communication and the total availability of the services.

3.1.8. Demand Flexibility Profiling

Description of design / functionality

FLEXCoop Demand Flexibility Profiling Framework will enable user-specific continuous & correlated monitoring of ambient /occupancy conditions and occupants' actions to extract context aware user preferences and understand comfort (dis)satisfaction zones, while considering health boundaries and set-points.

The Demand Flexibility Framework is composed of the following modules/engines that are further detailed below.

- ***Prosumer energy behaviour and comfort analysis engine***

  The engine will provide process of identifying user's actual preferences in terms of Heating, Ventilation and Air Conditioning (HVAC), Lighting and water heating loads operation, considering also health constraints imposed during building operation using data streams coming directly from ambient sensors and control actuators used by the building occupants during their everyday "interaction" with the respective devices. The engine will use the comfort models defined in T3.2.
- ***Prosumer context-aware flexibility profiling engine***

  This engine will provide the potential amount of demand flexibility from controllable devices, namely HVAC and lighting using the DER models defined in FLEXCoop T3.1.
- ***Virtual Thermal Energy Storage Module (VTES)***

  The VTES module will enable the definition and profiling of flexibility offered by virtual energy storage components using the respective profiling models defined in T3.3. An intelligent thermal storage software component will be developed applicable to various building types independent of their size, use and construction characteristics. FLEXCoop will provide the means (models, techniques and control algorithms) to enable efficient satisfaction of building occupants' thermal needs and preferences, even along the duration of DR events, by utilizing the thermal storage capabilities and flexibility of HVAC and water heating loads. The module will allow the deployment of highly efficient DR strategies by either guiding consumers or directly controlling the aforementioned loads to shift their demand in time (pre-heating/ cooling) and simultaneously reduce energy costs, maximize RES output absorption (self-consumption) and minimize grid problems.
- ***EV Flexibility Profiling Module***

  This module will define flexibility profiles of Electric Vehicles (EVs) considering a variety of parameters such as driving needs and distances (driving patterns), battery state of charge, discharge rate, time-of-day. The module will extract the capacity of EVs to act either as demand or storage assets (based on the models defined in the T3.4) in order to be introduced in the holistic context-aware and human-centric Demand Response framework
- ***Local Flexibility Analysis and Forecasting***

  The role of this module is to coordinate the different flexibility layers defined above.

The ultimate aim of this framework is to deliver holistic context-aware demand flexibility profiles, reflecting real-time demand and storage flexibility as a function of multiple parameters, such as availability of renewable energy, time, device operational characteristics, environmental context/conditions, user daily schedules and comfort preferences as well as health/ hygienic constraints. Such Demand Flexibility profiles will constitute dynamic constructs that will be continuously updated based on real-time data, thus providing a robust framework for monitoring and analysing energy related behaviour and corresponding flexibility features, to be further considered towards the implementation of DR strategies.

Although the Demand Flexibility Profiling consists of a number of modules/ engines as briefly described above, our plan is to be provided as a unified framework. Thus, for all these modules/engines the same software architecture principles, languages, libraries and frameworks as well as deployment environment and technologies are planned to be used. To this end and for avoiding repetition, for this point forward, we will treat the Demand Flexibility Profiling as a whole. However, in order for its sub-components to be clearly defined and their functionality to be clarified the UML component views are provided in the following section separated for each engine/ module composing the Demand Flexibility Profiling component of the FLEXCoop solution.

Description of architecture

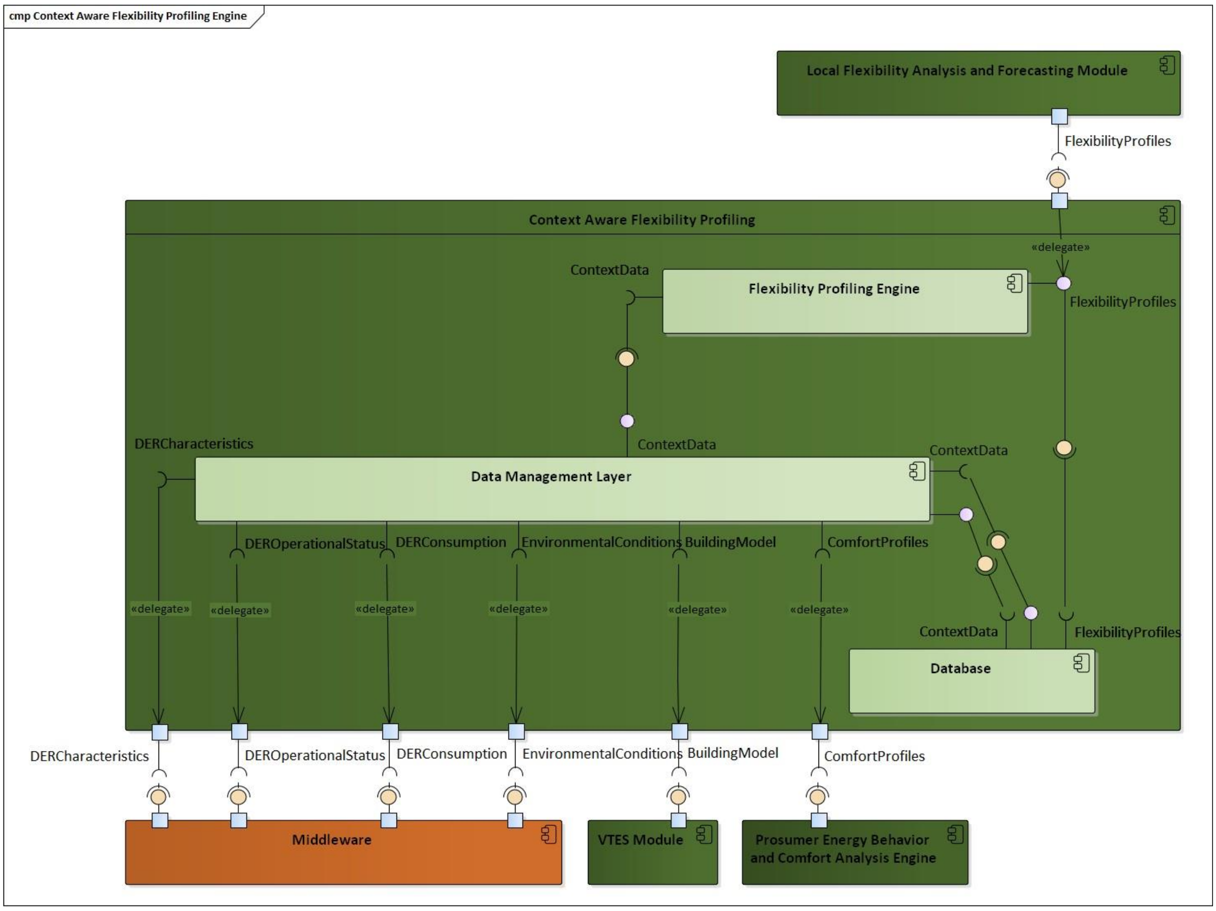


Figure 26: Context Aware Flexibility Profiling Engine CMP diagram

The Context Aware Flexibility Profiling Engine consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent context data and flexibility profiles
- The flexibility profiling engine that incorporates the algorithmic framework towards the extraction of flexibility profiles

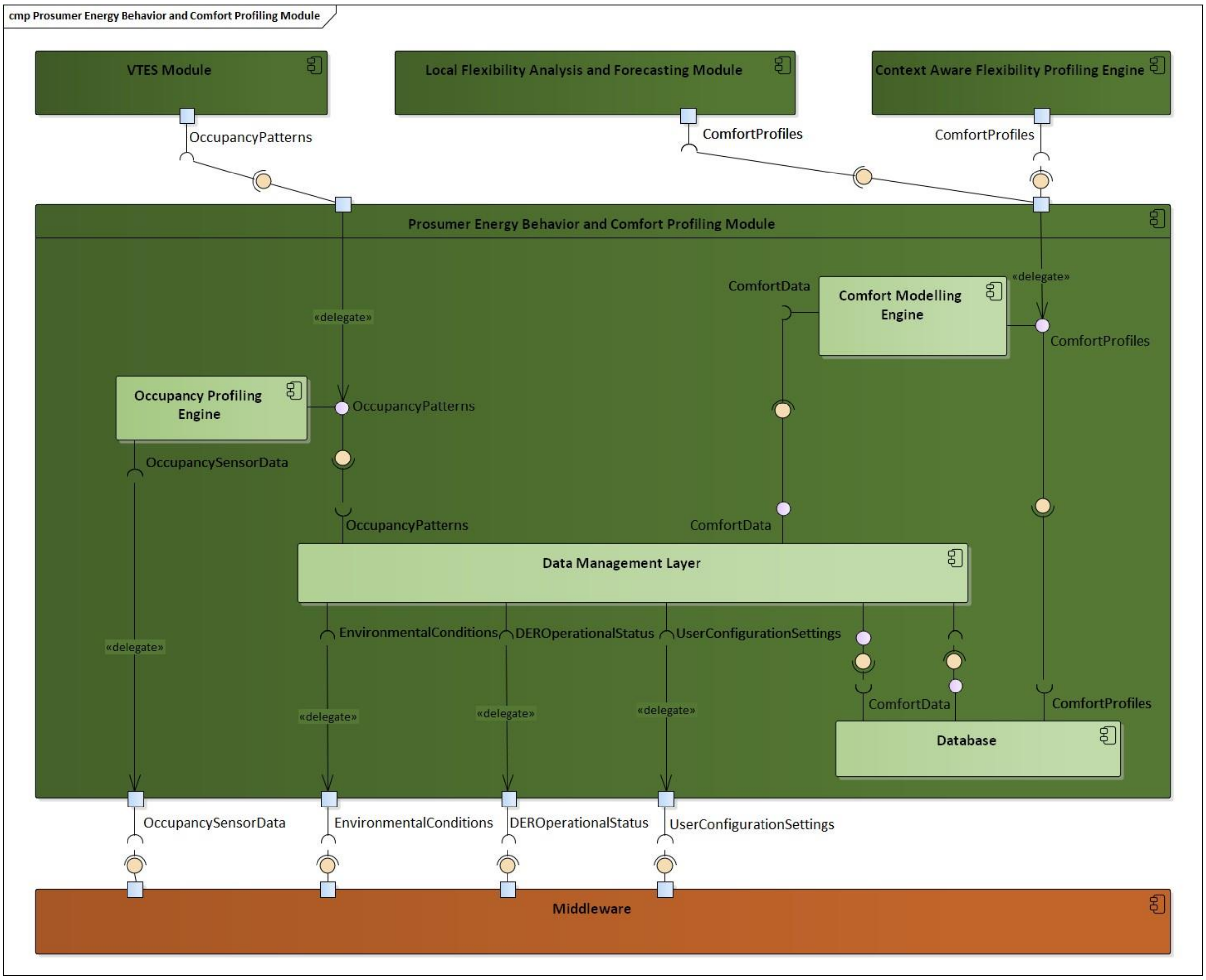


Figure 27: Prosumer Energy Behaviour and Comfort Profiling Module CMP diagram

The Prosumer Energy Behaviour and Comfort Profiling Module consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent comfort data and confort profiles
- The occupancy profiling engine that incorporates the algorithmic framework towards the provision of occupancy patterns
- The comfort modelling engine that incorporates the algorithmic framework towards extracting visual and thermal comfort profiles

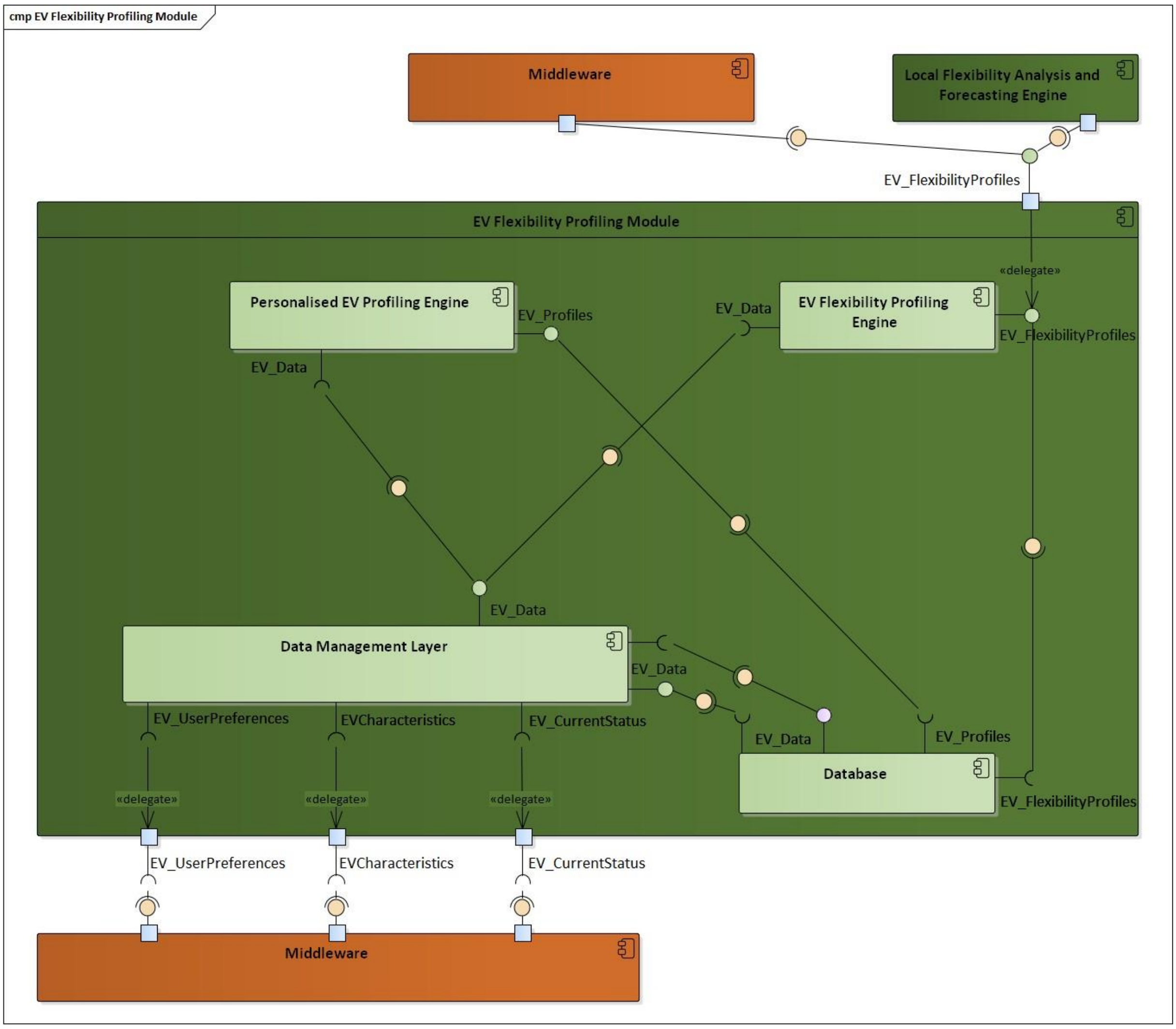


Figure 28: EV Flexibility Profiling Module CMP diagram

The EV Flexibility Profiling Module consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent EV-related data, personalised profiles of EV usage and EV flexibility profiles
- The personalised EV profiling engine that incorporates the algorithms required to extract patterns in EV usage and
- The EV flexibility profiling engine that incorporates the algorithmic framework used for extracting EV flexibility profiles

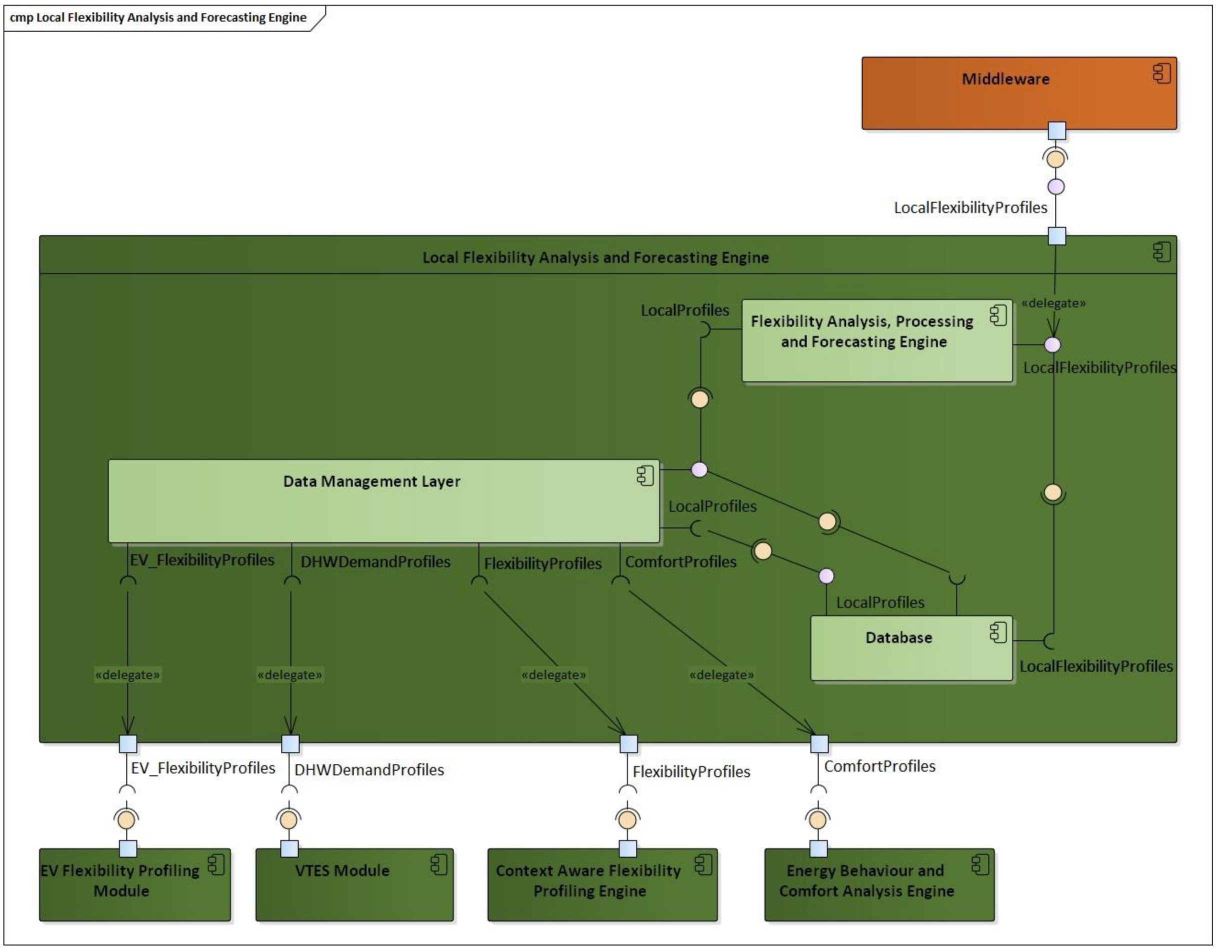


Figure 29: Local Flexibility Analysis and Forecasting Engine CMP diagram

The Local Flexibility Analysis and Forecasting Engine consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent local flexility, comfort and demand profiles
- The flexibility analysis, processing and forecasting engine that incorporates the algorithms required to extract local flexibility profiles

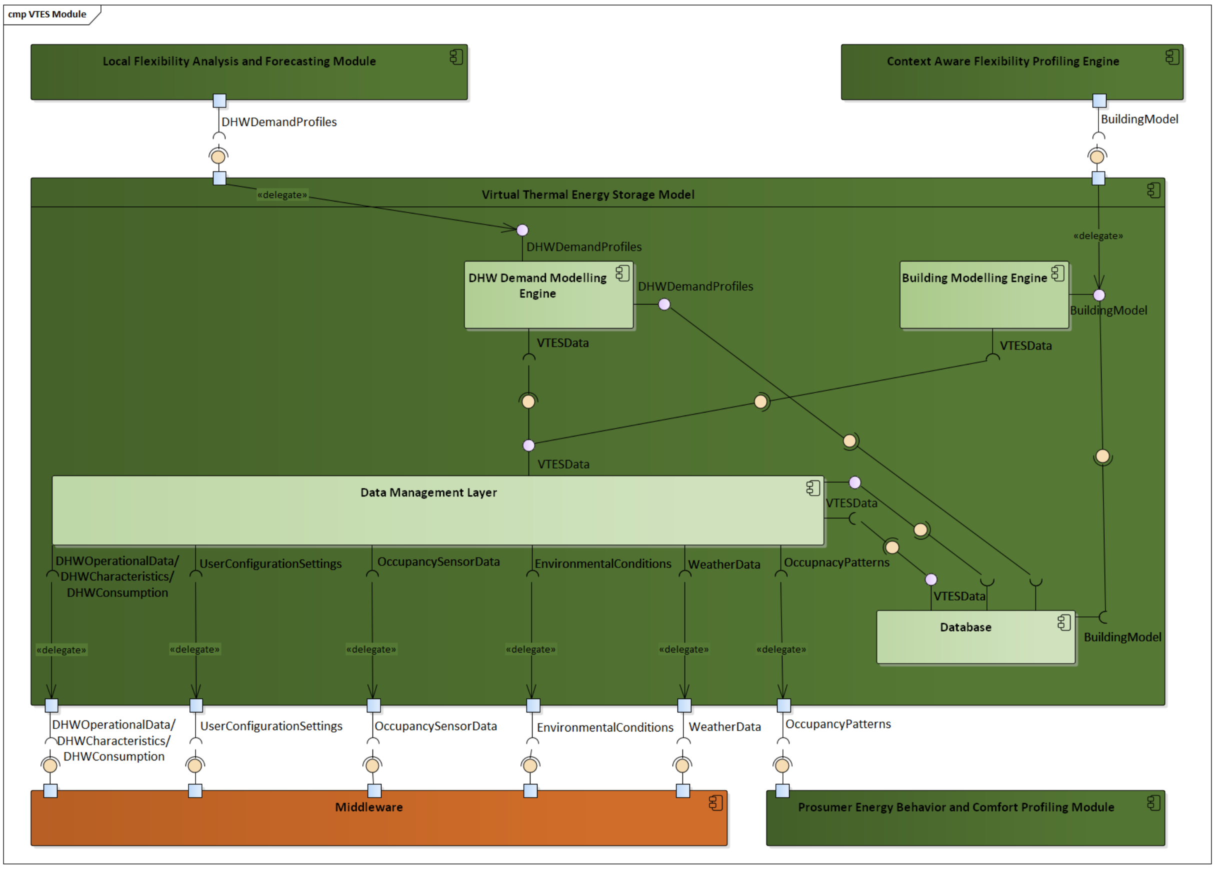


Figure 30: VTES Module CMP diagram

The VTES module consists of:

- A data management layer responsible for the data retrieval and orchestration
- A database to store persistent VTES related data building models
- The Domestic Hot Water (DHW) demand modelling engine to extract patterns in DHW usage
- The building modelling engine that incorporates the algorithmic framework that is used to provide building – thermal space models

The development of the Demand Flexibility Profiling component is planned to be based on Java/J2EE using a MySQL database to manage model parameters, information and data collected through the Message Oriented Middleware.

The software architecture will rely on the MVC pattern to provide modularity, ease of collaboration and reusability (see also Section 3.1.6) for more details). In particular, the Spring MVC framework will be used to provide the MVC pattern as well as a number of ready components towards developing a flexible and loosely coupled component. Hibernate ORM (version 5) will be used to map the object-oriented models to the relational database. WEKA library will be used for data mining.

The proposed software architecture ensures the modularity of the component while the Spring MVC supports the design, development and exposure of RESTful services that will be used for the communication with external components, as further described below.

Description of component interaction

In the proposed software architecture, RESTful services to external components can be exposed to support the functionality of the Demand Flexibility Profiling component. Additionally, a publish-subscribe mechanism (i.e. AMQP, MQTT) for communication with 3rd parties will be supported. This way, the Demand Flexibility Profiling component will interface through Message Oriented Middleware:

- With OSB towards receiving real time data from pilot premises and triggering control commands to DERs
- With Aggregator UI towards information gathering on real-time (& short-term forecasts) of context-aware demand flexibility profiles per building/OSB
- With the Local Demand Manager for DR monitoring and reconfiguration
- With the Flexibility Forecasting, Segmentation and aggregation module for providing DER historical data (e.g. consumption)
- With the Global Demand Manager towards providing data on demand flexibility and DER availability based on user preferences
- With Prosumer portal for configuration of the demand flexibility profiling component as well as for providing DER-related data, real-time and historical energy performance, etc.

Description of deployment

The Demand Flexibility Profiling component will be a cloud-based reliable and stable application responsible to perform analytics over historical data towards the extraction of comfort, DER, thermal storage and EV profiles. The deployment of the Demand Flexibility Profiling component will be hosted in an Apache Tomcat server (open source implementation of the Java Servlet, JavaServer Pages and Java WebSocket technologies) operating as an application server. In addition, Apache or NGINX server is planned to be used as a web server.

3.1.9. Middleware

Description of design / functionality

This is the data management layer of Aggregator Applications. Based on CIMNE's beedata platform, enabling (i) the realization of a Message Oriented Middleware (ESB), deployed at the district level and ensuring fine-grained and interoperable communication and data exchange with a variety of heterogeneous sources (buildings, district DER management systems, DR actors' proprietary systems, weather data sources and wholesale energy prices); and (ii) mass treatment and storage of huge volumes of heterogeneous data sets. The platform will employ appropriate techniques for data cleaning, normalization, curation and semantic annotation to ensure high performance and ability to adapt to the requirements and needs of the DR value chain stakeholders.

To ensure interoperability with external systems, the FLEXCoop Big data management platform will comprise in an open platform and application software framework to establish seamless, transparent and homogeneous standards-based (OpenADR 2.0b, USEF) interfaces to all integrated FLEXCoop components. Furthermore, it provides appropriately defined semantic virtual entities (semantically enhanced components) thus incorporating the necessary semantics for the efficient management of the various information streams.

The BigData Storage and Data Analysis within the middleware is implemented using Apache Hadoop, providing a secure and reliable distributed long term data storage, and allowing the execution of Data Analysis using MapReduce technology. Beside this, to provide fast access to the data, a short term data storage and cache is implemented using MongoDB.

Description of architecture

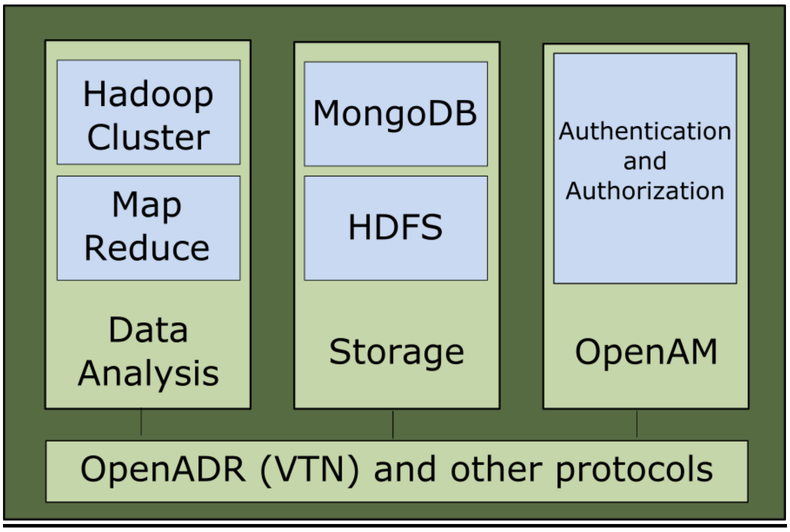


Figure 31: Middleware CMP diagram

The development of the Middleware component is planned to be based on Python 3, providing an OpenADR 2.0b implementation, and it will be used as a main connection bridge between all other FLEXCoop components.

OpenADR is a server-client communication protocol, where the server implements a Virtual Top Node (VTN) and the clients a Virtual End Node (VEN), the communication always happens between VTN and VEN.

To comply with the OpenADR protocol, VTN must implement two low level communication protocols HTTP and XMPP, while the VEN will only implement one (VEN using different protocols can communicate to the same server).

The Middleware will be in charge of all the communication among the FLEXCoop components, allowing the secure communication using the OpenADR standard and providing access to the BigData Storage and Data Analytics.

The middleware will also provide user authentication and authorization implementing a centralized authentication service using OpenAM[9]. This platform will ensure the access to all the different FLEXCoop Components with a single log-in to the platform.

The BigData Storage within the middleware is implemented using Appache Hadoop[10], providing a secure and reliable distributed data storage, and allowing the execution of Data Analysis using MapReduce technology.

Description of component interaction

The Middleware component interacts with all the FLEXCoop components in order to allow the communication between them. All the other components shall implement an OpenADR Virtual End Node (VEN) and register to the Middleware in order to start the communication.

It is also contemplated the implementation of other communication protocols if required by some components, in order to create a highly accessible platform.

Description of deployment

The Middleware will be deployed in a cloud-based server, accessible for all the other components through HTTP or XMPP as required by the OpenADR protocol.

The deployment will be hosted by using Python Flask server. The deployment will be hosted by using Python Flask[11] server. To ensure the server is working in a reliable way, it will be managed by the Unix task manager Supervisor[12] to monitor the server process. Finally, the middleware will be configured in a NGINX server to allow server cache and load balancing.

3.1.10. DER Registry

Description of design / functionality

The **DER Registry** will enable consumers/Prosumers to publish DERs belonging to their control portfolios and semantically enhance them in order to advertise and highlight their capabilities (as well as availability, geographic location, etc.), provide data for DER discovery and VPP formulation and allow for successful provisioning and acquisition of specific and dedicated services from DERs.

The **DER Registry** will be implemented as a database, which hold the state of all DERs connected to the FLEXCoop infrastructure. Fraunhofer FOKUS is the responsible partner of its implementation. DERs register on the registry when they connect to the system and update their status in the registry on any changes. The registry needs to keep track of all DERs, which includes actively monitoring keep alive messages from the DERs and mark timed out DERs as

---

[9] https://backstage.forgerock.com/docs/openam/13.5

[10] http://hadoop.apache.org/

[11] http://flask.pocoo.org/

[12] http://supervisord.org/

such. For accounting and monitoring reasons, the registry keeps a log of all events passed through the registry. In the context of FLEXCoop we refer to the **OSB** as a DER.

Description of architecture

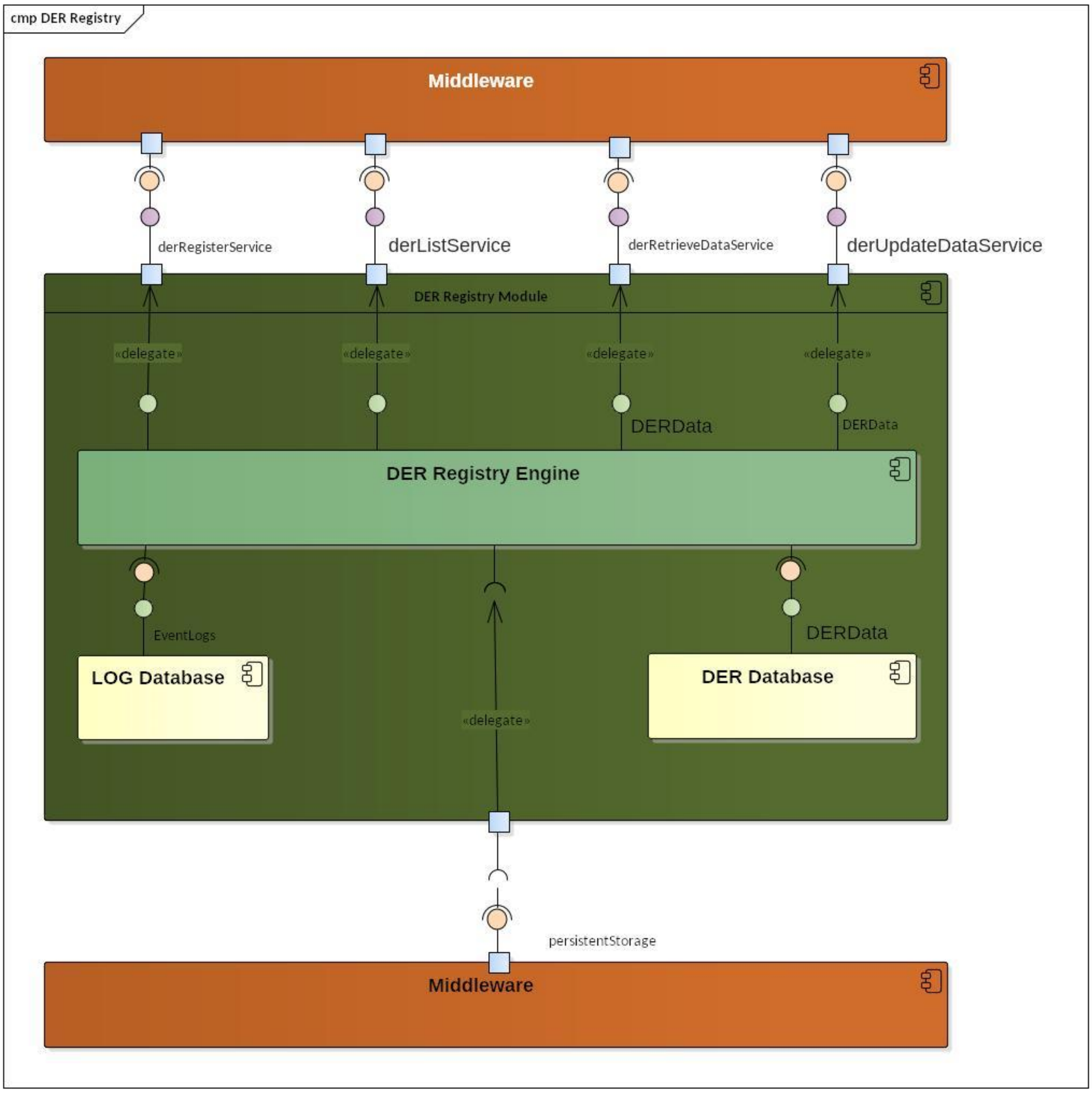


Figure 32: DER Registry CMP diagram

We plan to implement the **DER Registry** with Python3 as main programming language. The architecture will follow the MVP pattern. As the registry will not need a UI but the **DER Registry View** as well as the **Open Market Place** will need such, these components will share code and data structures. Therefore, common code fragments and models will be implemented as a library to be reusable. We plan to use either the Django[13] or flask[14] framework to implement the necessary APIs and UI components.

The DER Registry will use persistent storage function provided by the **Middleware** to store the following information:

- Details of all DERs known to the registry
- Information on the state of the currently registered DERs to provide this information to other components.
- The history of events monitored by the registry

The registry will use an in memory DB like redis[15] to hold states and other temporary data. It may also be used to cache data before pushing it to the persistent API provided by the Middleware.

Description of component interaction

The DER Registry interacts with the following Components:

The storage API provided by the **Middleware** will be used to store all persistent information. The same API will be used to read information and configuration options.

The **Middleware** will be used to communicate with the DERs. Therefore, the **DER Registry** will implement an OpenADR VEN to communicate with the **Middlewares** VTN. Event will be passed to the **Middleware** this way.

Description of deployment

The **DER Registry** deployment will be based on Python Package. This is the most basic and most robust way to deliver a Python Application. It can be used to be deployed via different package manager and by used in Docker or Ansible based deployments. These deployment frameworks can then be used to deploy the external dependencies like database server for the Registry.

3.1.11. Open Smart Box

Description of design / functionality

FLEXCoop Open Smart Box (OSB)

The OSB is a real time monitoring sensor/ actuator node. It is the hardware component of the architecture enabling the realization of the FLEXCoop concept for human-centric demand

[13] https://www.djangoproject.com/

[14] http://flask.pocoo.org/

[15] https://redis.io/

response deployment. This is the in premises gateway (GW) acting as the local agent of the Aggregator in the building environment.

It consists of a modular communication and sensing/ control smart system with a threefold role: a) integration of a wide range of sensors, such as luminance, occupancy, temperature, humidity, air quality, b) setting of the control interfaces to monitor/control the operation of specific devices, namely lights, water heaters and heat pumps and c) setting of the interfaces to measure the electricity consumption of these specific devices.

Description of architecture

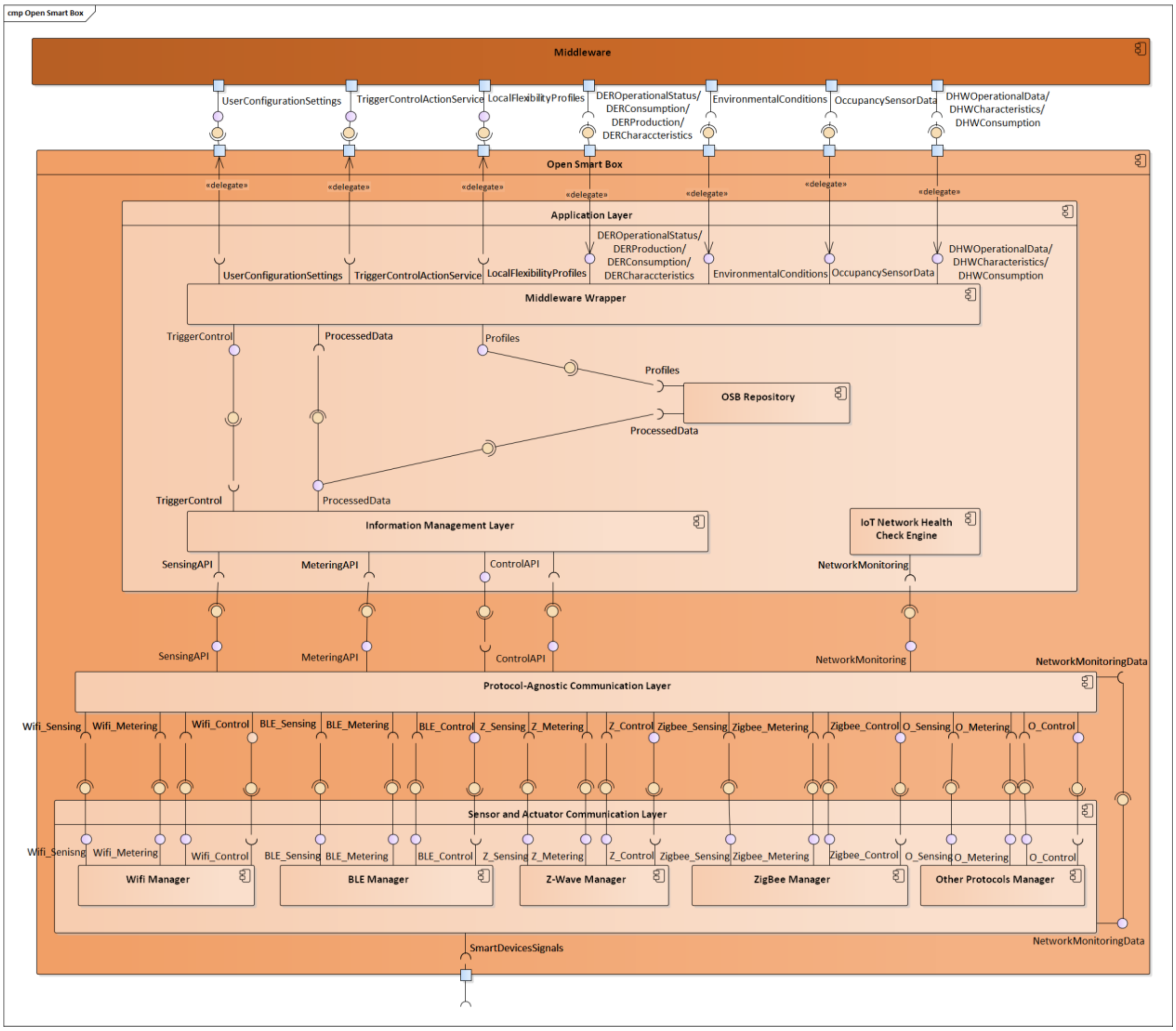


Figure 33: Open Smart Box CMP diagram

The OSB will be composed of three main interconnected layers providing modularity, reusability and ease of maintenance. Following a bottom – up presentation of the OSB, we briefly present the different layers below.

- ***The Sensor and Actuator Communication Layer***
  This layer is responsible for the intercommunication among the gateway and the smart devices (sensors and actuators) that are installed in the home environment and are intended to be controlled by the OSB. This layer embeds all the communication drivers required for supporting different communication protocols such as Wi-Fi, z-wave, BLE, ZigBee, etc.
- ***Agnostic Communication Layer***
  This layer is responsible for the integration of all the available functionality provided by the connected devices through different protocols in order to be collected and converted into a generic API. To this end, OpenHAB automation software is planned to be used as a protocol-agnostic platform providing scalability and unlimited compatibility with the gateway.
- ***Application Layer***
  This layer is responsible for the basic functionality of the OSB including: control dispatching, managing of sensing and energy consumption data, data normalization (if required), storage of demand flexibility profiles, interchanging information with the Middleware through web services, etc. The application layer will be developed in Java.

Description of component interaction

To interchange data with the Middleware, the Jersey framework will be used for developing RESTful web services that support data exposure in a variety of representation media types and abstract using a client-server communication. Thus, the application layer of the OSB will also support a publish-subscribe mechanism (i.e. AMQP, MQTT) for communication with 3rd parties as well as the OpenADR standard. This way, the OSB will interface through the Middleware:

- With the Demand Flexibility Profiling engine for providing real-time sensing and energy consumption data
- With the DER registry for registration of the DER devices connected to the OSB
- With the Local Demand Manager for receiving signals for controlling connected devices as well as for providing real-time context – environmental conditions
- With the Global Demand Manager for providing real-time context – environmental conditions and demand flexibility status
- With the Prosumer portal for providing real-time sensing and energy consumption data, RES generation and storage information, etc. and for receiving settings for connected devices (e.g. comfort settings)

Description of deployment

The objective is to deliver OSB as an in-premises local agent ensuring reliability and stability. OSB may be hosted in a single-board computer Raspberry Pi 3 Model B, which covers the minimum hardware requirements. Each OSB will be delivered with an OpenVPN setup to facilitate the maintenance/monitoring phase.

Docker platform will also be installed in OSB smoothing the operation between different modules that may be deployed in Raspberry Pi, regarding hardware resources. In parallel docker will provide interfaces for standardize and automate the way that OSB applications may be built, deployed, managed and secured.

3.1.12. IEC 61850 Server/ DER Management System

Description of design / functionality

The IEC 61850 is an international standard that was originally designed to define communication protocols for intelligent electronic devices at electrical substations, but the IEC 61850 series of protocols in practice go beyond the electrical substations and currently represent a *lingua franca* specification for the automation architecture. This makes the IEC 61850 protocol very highly relevant for FLEXCoop and making the FLEXCoop solution compatible with IEC 61850 makes it easily implementable. In other words, IEC 61850 is a practical requirement for the actual equipment that has to be operated directly, and this includes but is not limited to the local DERs In this context, the DERs are primarily district generation and storage systems and not the in-house DERs that would be managed by the OSB component. Numerous vendors and integrators of the district-level generation and storage systems already utilize and support the IEC 61850 protocol.

In the shortest possible form – the IEC 61850 server of the FLEXCoop solution will serve as a gateway. This gateway will receive the relevant internal FLEXCoop messages being passed through the Message Oriented Middleware, and then activate the translation from the internal FLEXCoop messages towards the 61850 side. On that side, where the actual message payload (e.g. the execution of a command to set the operating point) will be activated. Similarly, the gateway will generate messages in the message-oriented middleware (e.g. to notify about inactivity or problems on the district DER side).

By using this gateway, the FLEXCoop solution can rely on the IEC 61850 protocol to perform commands and actually operate with the local equipment, or it can resort to only using the IEC 61850 protocol to push the measurement and reporting values. The gateway makes FLEXCoop solution fully support the IEC 61850 protocol-based communication, and the successful usage with the actual DERs will only be a matter of gateway configuration. Furthermore, the IEC 61850 series of standards determines the description of the devices (primarily in electrical substations) and the exchanged information between the devices, both at runtime and at configuration time.

In this project, KONČAR will for the most part take advantage of the existing and proven IEC 61850-server software component, which was developed within the scope of OS4ES project. The IEC 61850 communication stack will be adapted on the internal side to satisfy the FLEXCoop architecture, i.e. so that the IEC61850 server can properly communicate with the message-oriented middleware. The OS4ES IEC61850 software stack supports the IEC 61850 communication over XMPP protocol, including client and server functionality. This software stack will be used in the gateway. If required in the pilots, legacy OSI-based MMS carriers for 61850 protocols, as well as other legacy protocols can be supported by means of a KONČAR-developed protocol converter device installed at the pilot sites.

The IEC 61850 data model is a hierarchical, function object oriented model.

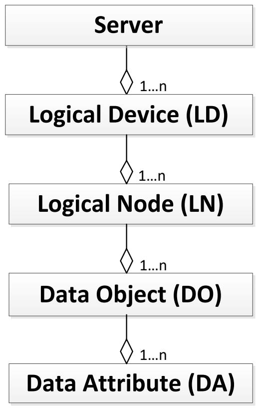


Figure 34: Object hierarchy in the IEC 61850 data model

Each physical Intelligent Electronic Device (IED) can perform several functions. IEC 61850 ed. 2.0 provides provision for logical devices within a single physical device (a *server* in this context). Within each of the logical nodes, there is a number of data, some of which is deemed mandatory. This data can be subdivided into:

- **Common data** relevant to the logical node (which is independent from the actual dedicated function represented by the logical node).
- **Status information** – either the status of the process or the status of the function allocated to this particular logical node.
- **Settings** – information relevant for the functioning of this logical node.
- **Measured values** – analogue data either measured from the process or calculated from the actual values in the functions. and
- **Controls** – data changed by commands.

The IEC 61850 model relies on abstract communication service interfaces (ACSIs). The IEC 61850 allows both client/server interfacing as well as peer-to-peer interfacing, i.e. it allows vertical and horizontal communication, as well as additional services such as time synchronization and file transfer. The received data can either be spontaneous, by request or by subscription, and sending can also be by request or by subscription. The vertical ACSI (Abstract Common Service Interface) maps to client/server communication. Horizontal ACSI conforms to the publish/subscribe model. The following table depicts the ACSI services.

Service model	**Description**	**Services**
Server	Represents the external visible behaviour of a device. All other ACSI models are part of the server.	GetServerDirectory
Application association	Provision of how two devices can be connected. Provides different views to a device: restricted access to the server's information and functions.	Associate Abort Release
Logical device	Represents a group of functions; each function is defined as a logical node.	GetLogicalDeviceDirectory
Logical node	Represents a specific function of the substation system, for example, overvoltage protection.	GetLogicalNodeDirectory GetAllDataValues
Data	Provides a means to specify typed information, for example, position of a switch with quality information, and timestamp.	GetDataValues SetDataValues GetDataDefinition GetDataDirectory
Data set	Allow to group various data together.	GetDataSetValues SetDataSetValues CreateDataSet

		DeleteDataSet GetDataSetDirectory
Setting group control	Defines how to switch from one set of setting values to another one and how to edit setting groups.	SelectActiveSG SelectEditSG SetEditSGValues ConfirmEditSGValues GetEditSGValues GetSGCBValues
Reporting and logging	Describes the conditions for generating reports and logs based on parameters set by the client. Reports may be triggered by changes of process data values (for example, state change or deadband) or by quality changes. Logs can be queried for later retrieval. Reports may be sent immediately or deferred (buffered). Reports provide change-of-state and sequence-of-events information exchange.	Buffered RCB: Report GetBRCBValues SetBRCBValues Unbuffered RCB: Report GetURCBValues SetURCBValues Log CB: GetLCBValues SetLCBValues QueryLogByTime QueryLogAfter GetLogStatusValues
Generic substation events	Provides fast and reliable system-wide distribution of data; peer-to-peer exchange of IED binary status information. GOOSE means Generic Object Oriented Substation Event and supports the exchange of a wide range of possible common data organized by a DATA-SET	GOOSE CB: SendGOOSEMessage GetGoReference GetGOOSEElementNumber GetGoCBValues SetGoCBValues
Transmission of sampled values	Fast and cyclic transfer of samples, for example, of instrument transformers.	Multicast SVC: SendMSVMessage GetMSVCBValues SetMSVCBValues Unicast SVC: SendUSVMessage GetUSVCBValues SetUSVCBValues
Control	Describes the services to control, for example, devices or parameter setting groups.	Select SelectWithValue Cancel Operate CommandTermination TimeActivatedOperate
Time and time synchronization	Provides the time base for the device and system.	TimeSynchronization
File transfer	Defines the exchange of huge data blocks such as programs.	GetFile SetFile DeleteFile GetFileAttributeValues

Table 4: IEC 61850 Service models

The following table depicts the mapping between ACSI services and application-level communication functions in the 61850 protocol.

For the purposes of FLEXCoop, it is obvious that a class of application-level functions will not be handled within the DER Management System – the DER management system only performs functions relevant for a certain DER.

ACSI service / Application-level communication functions	Device registration	Device search	Contract management	Access rights management	Control command / Setpoint change	Status update / Measurement update	Data model management
Server							X
Application association							X
Logical device							X
Logical node							X
Data							X
Data set							X
Setting group control						X	
Reporting and logging						X	
Generic substation events						X	
Transmission of sampled values						X	
Control					X		
Time and time synchronization							
File transfer							

Table 5: ACSI service

For the FLEXCoop solution, as the gateway will be linked to the message oriented middleware, the most of communication will occur spontaneously, either indicated by the DER side or the message oriented middleware side.

Description of architecture

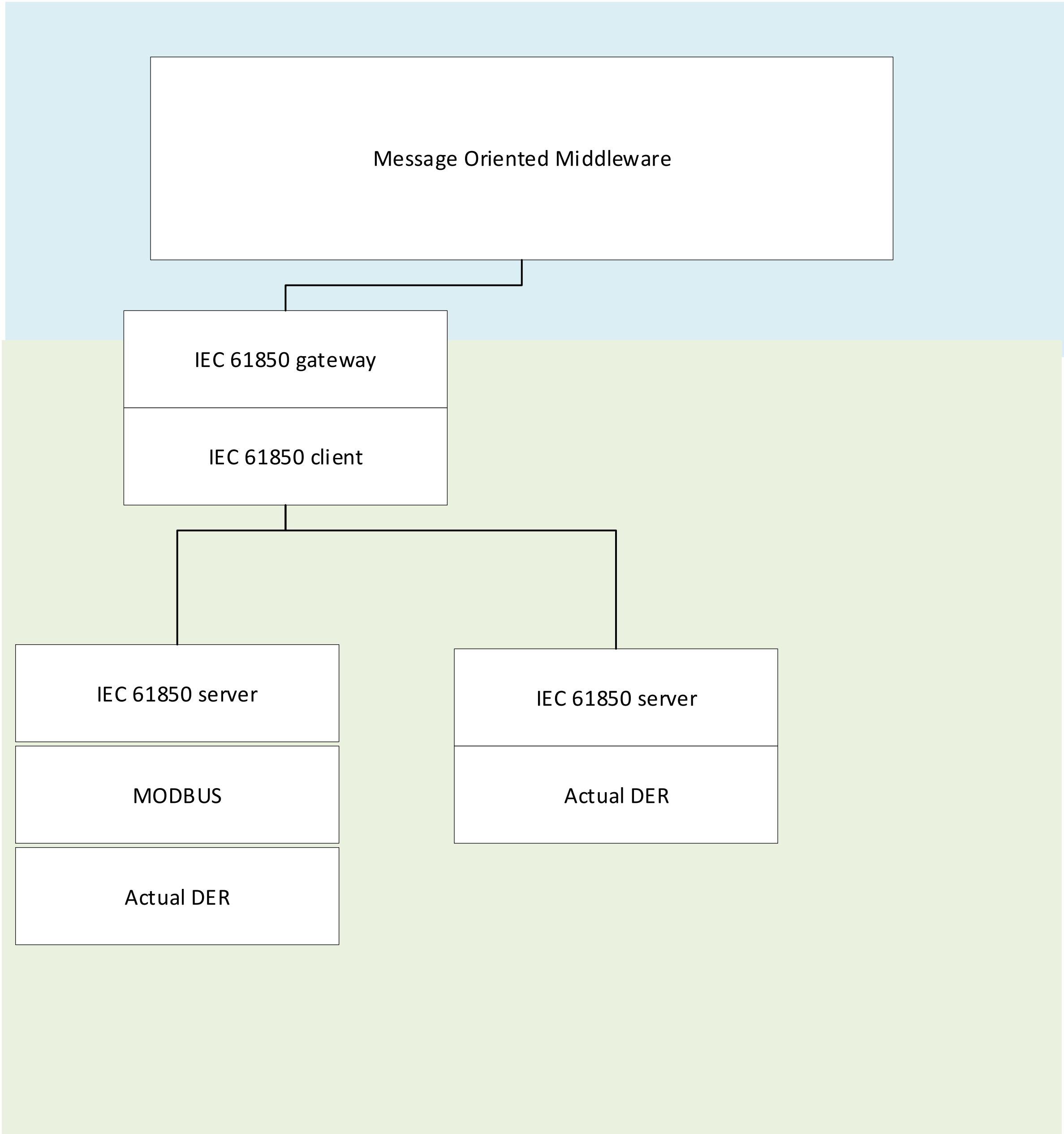


Figure 35: General Architecture of FLEXCoop DER Management System

The DER Management System consists of three principal components:

- IEC 61850 Gateway
- IEC 61850 communication client
- IEC 61850 communication server

In the FLEXCoop architecture, it is supported that there is a single DER manager for multiple DERs, as well as having multiple DER gateways, each of these having either a single DER under control or multiple ones. This allows a higher degree of spatial flexibility while maintaining the whole value chain in the communication.

Description of component interaction

The primary role of 61850 gateway is to communicate with the message oriented middleware and receive / retrieve the messages relevant for the particular instance of the DER manager, and then passes them downstream to the relevant DERs via IEC 61850 protocol. This is a subcomponent of the DER Management system that requires significant additional development for the FLEXCoop project, while the other two components capitalize on prior development for the OS4ES project. The 61850 gateway will, in accordance and coordination with the rest of developments, primarily of message oriented middleware, implement the OpenADR VTN and VEN functionalities in order to be able to communicate with the rest of FLEXCoop solutions.

As stated above, the IEC 61850 client and server components are already mostly developed within the scope of OS4ES project and will be used and configured here within the DER Management system. This design ensures the FLEXCoop solution will support numerous DERs in the field, as no DER communicates with anything else besides the IEC 61850 client, integrated with the gateway. Each DER instance either is advanced enough to support the IEC 61850 protocol on its own, or it only supports older legacy protocols such as MODBUS and then the FLEXCoop IEC 61850 server is needed as well to complete the communication channel.

Description of deployment

The libraries used in DER gateway are lightweight cross-platform libraries developed in Python 3. They have been successfully tested within the scope of OS4ES project in Python, and are completely compatible with Docker-based deployment used elsewhere in the project.

The DERs in the field may not support the communication protocol required by the IEC 61850 gateway. This can be mitigated by using the KONČAR protocol converter. This device, readily available, supports numerous legacy protocols. It is a Linux-based appliance, based on commodity hardware and supporting wide range of industry standard legacy protocols. By using this approach, the focus is kept on key FLEXCoop developments, while keeping it compatible with the situation in the field. For all the purposes of FLEXCoop project, focusing on XMPP-based protocol ensures the concept is sound and valid.

4. Deployment

The deployment of the FLEXCoop software components aims on reflecting the heterogeneous requirements defined in the sections above. Besides these requirements, the goal is to provide a stable and reproducible system which can be scaled in terms of resource usage, geographical distribution and redundancy.

The deployment will also reflect that the functional components are highly decoupled, which allows a distributed deployment as well as a centralised deployment, depending on the use case and available resources. As shown in Figure 5 the Middleware is the central point of communication. Therefore, only this component needs to be reachable for all other components which reduces the complexity of the distributed deployment.

During the project the different components will be deployed either in-house or on third party servers respectively on cloud provider infrastructure. This allows to work with deployment frameworks or strategies fitting to the specifications of the different component and development processes. It also enables the project to evaluate different strategies and find the best solution for each component as well as for the whole project.

Component specific deployment details, can be found in each component section of this document. As most components are not yet implemented at the time of writing this document, there can be changes on the details during the development.

The final implementation of the FLEXCoop architecture will be deployable on cloud infrastructure like kubernetes, AWS or Azure as well as on physical servers. As the energy related infrastructure is designed with a very long operation time in mind, it is important to utilise reliable resources for the FLEXCoop solution deployment.

5. Interoperability and Component Interactions

Interoperability is an important topic which strikes three aspects of FLEXCoop. The first aspect is the interoperability with devices in the Home respectively Smart Home area. The second aspect is interoperability with external systems based on standards from the energy domain e.g. IEC 61850 and OpenADR. The third aspect is interoperability with systems used by the different roles who interact with the FLEXCoop system.

By having a central component which is able to adapt different information types and pass it in a unified way to other components the FLEXCoop architecture is able to provide a simple way to add new interface to external systems easily. The IEC 61850 Server component is a perfect example how interoperability with an external system can be achieved.

As mentioned before the interaction between the components is designed to go through the Middleware. To provide another point of view on the communication architecture the following table will ignore the middleware and lists which components interact with each other from a logical viewpoint

	GDM View	DER Registry View (AGR / PROS)	Open Marketplace View (AGR/PROS)	Visual Prosumer Toolkit	Open Marketplace	Local Demand Manager	Global Demand Manager	Flexibility forecasting, segmentation and aggregation	DR Settlement / Remuneration	Demand Flexibility Profiling	Middleware	DER Registry	Open Smart Box	IEC 61850 Server / DER Management System
GDM View		--	--	--	--	--	DREvents VPPInformation	FlexibilityForecasting FlexibilityReports ClustersGeneralInformation ClustersUsersInformation	DRCampaignsInfo DRCampaignsRemuneration	DERConsumption		--	--	--
DER Registry View (AGR / PROS)	--		--	--	--	--	--	--	--	--		Device status and availability	--	--
Open Marketplace View (AGR/PROS)	--	--		--	Contracts (and contract template parameters) - Historical contractual parameters	--	--	--	--	--		--	--	--
Visual Prosumer Toolkit	--	--	--	--	Contracts (and contract template parameters) - Historical contractual parameters	--	DR Requests for DR events	Prosumers clusters (if needed)	DR related KPIs about DR settlement	Real time and historical information about energy consumption, generation, demand flexibility, etc..		Device static parameters as provided by OSB to DER Registry	--	Demand /Generation forecasting information
Open Marketplace	--	--	Contracts (and contract template parameters) - Historical contractual parameters	--		--	--	--	--	--		Device status and availability	--	--
Local Demand Manager	--	--	--	--	--		DRSignalData	--	--	LocalFlexibilityProfiles	WeatherData	RegisteredDER	--	--
Global Demand Manager	--	--	--	--	Contractually availble DERs	DREvent		--	--	LocalFlexibilityProfiles	DSOSignal WeatherData Wholesale prices	--	--	District generation/ Sorage data
Flexibility forecasting, segmentation and aggregation	--	--	--	--	--	--	--		--	LocalFlexibilityProfiles		RegisteredDER	--	--
DR Settlement / Remuneration	--	--	--	--	Contracts (and contract template parameters)	--	FlexibilityDelivered	--		DERConsumption EnvironmentalConditions OccupancyPatterns		--	--	--
Demand Flexibility Profiling	--	--	--	UserConfigurationSettings	--	--	--	--	--		WeatherData	--	DERCharacteristics DEROperationalStatus DERConsumption OccupancySensorData EnvironmentalConditions DHWOperationalData DHWCharacteristics DHWConsumption	--
Middleware														
DER Registry	--	Device status and availability	--	Device status and availability	--	--	--	--	--	--			--	--
Open Smart Box	--	--	--	--	--	--	--	--	--	--	--	--		--
IEC 61850 Server / DER Management System														

Table 6: Required Interface to be provided by the Middleware

6. Conclusion

The current state of the architecture for the FLEXCoop system as it is described in this document includes all components necessary to fulfil requirements resulting from FLEXCoop T2.5. We have provided a detailed description of the functionality, structure, interaction and deployment of the components. To achieve the best possible results the programming languages and frameworks that are planned to be used have been identified and analysed accordingly. The loose coupling of the components, enabled by the middleware, allows to use different approaches on component internal architecture and tooling without the risk of incompatibility.

References

[1] “CEN-CENELEC-ETSI Smart Grid Coordination Group - Smart Grid Reference Architecture.” Nov-2012.
[2] European Network of Transmission System Operators for Electricity, “The Harmonised Electricity Market Role Model.” European Network of Transmission System Operators for Electricity, 2015.
[3] CEN-CENELEC-ETSI Smart Grid Coordination Group, “SG-CG/ M490/F_ Overview of SG-CG Methodologies, version 2.0,” 2014. [Online]. Available: http://www.energynetworks.org/assets/files/electricity/engineering/Standards/SGCG%20Reports%20071014/SGCG_WGMethod_Sec0076_INF_ReportforComments(incl_annexes).pdf. [Accessed: 24-Sep-2018].
[4] “The TOGAF Standard, Version 9.2 - Definitions.” [Online]. Available: http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap03.html#tag_03_26. [Accessed: 24-Sep-2018].
[5] Rolf Apel, Siemens Infrastructure and Cities, “Smart Grid Architecture Model - Methodology and practical application,” presented at the EPCC, 12th International Workshop, Bedford Springs, PA, 2013.