GOFLEX

Generalized Operational FLEXibility for Integrating Renewables in the Distribution Grid (GOFLEX)

D2.1 Automatic Trading Platform Requirement & Interface Specification

April 2017

Imprint

Contractual Date of Delivery to the EC: **30 April 2017**
Actual Date of Delivery to the EC: 28 April 2017

Author(s): **Laurynas Šikšnys (AAU), Bijay Neupane (AAU), Sašo Brus (INEA), Gregor Černe (INEA)**
Participant(s): **Marinšek Zoran (INEA), Torben Bach Pedersen (AAU)**
Reviewer(s): **Seshu Tirupathi (IBM)**

Project: Generalized Operational FLEXibility for Integrating Renewables in the Distribution Grid (GOFLEX)

Work package: **WP2**
Confidentiality: Public
Version: **1.2**

Contact: **Laurynas Šikšnys – siksnys@cs.aau.dk**
Website: **www.goflex-community.eu**

Legal disclaimer

The project Generalized Operational FLEXibility for Integrating Renewables in the Distribution Grid (GOFLEX) has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731232. The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the Innovation and Networks Executive Agency (INEA) or the European Commission (EC). INEA or the EC are not responsible for any use that may be made of the information contained therein.

Copyright

Executive Summary

This report provides a detailed description of the functional and non-functional requirement of the Automated Trading Platform (ATP) in the GOFLEX project. The primary aim of the report is to have a clear view on what ATP should do and what functionalities it should have to serve the purpose. The report outlines the overall architecture of the ATP along with a description of input/output data, dependencies, and functionalities for each sub-system of the ATP. This document comprises project deliverable 1 of Work Package 2 (WP2).

Document History

Version	Date	Status	Author	Comment
1	03 Mar 2017	For internal review	Laurynas Šikšnys	First draft
1.1	20 Apr 2017	Pre-final version	Bijay Neupane	Fixes based on reviews
1.2	21 Apr 2017	Final version	Laurynas Šikšnys	Fixes based on reviews

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

Abbreviation	Definition
ATP	Automatic Trading Platform
CA	Consortium Agreement
DOMS	Distribution Observability and Management System
DR	Demand Response
DSO	Distribution System Operator
EMS	Energy Management System
FMAN	Flex-Offer Manager
FMAR	Flex-Offer Market
FOA	Flex-Offer Agent
FOG	Flex-offer Generator
GA	Grant Agreement
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
MQTT	MQ Telemetry Transport
SP	Data Service Platform
TSO	Distribution System Operator
WP	Work Package
xEMS	One of the energy management systems used in GOFLEX
XMPP	Extensible Messaging and Presence Protocol

1 Introduction

1.1 Purpose

This document is the first deliverable (D2.1) of the work package No 2 (WP2) of the GOFLEX [GOFLEX - 2016] project. WP2 will develop a so-called Automatic Trading Platform (ATP), which is a decentralized automatic demand-response trading platform encompassing all active demand-response (DR) providers (prosumers), intermediaries (aggregators, VPPs), and users (BRPs, DSOs, TSOs). The platform will offer different DR activation mechanisms (indirect and direct device-control) and monetization schemas derived from the category of dynamic pricing (e.g. flexibility contracts, market-based trading), which are suitable for different DR trading scenarios (environments) and use-cases. ATP will be developed in several major iterations/generations, addressing the following key objectives within WP2:

- Integrate the individual existing KIBERnet [KIBERnet - 2017], KIBERnet FLEX, and TOTALFLEX [TotalFlex - 2012] solutions into a single integrated DR trading platform, offering a common toolset, which allows maximizing the value of flexibility in different DR trading scenarios and use-cases, e.g., requiring different flexibility services/trading environments and prosumer types (including EVs, storage systems).
- Align the platform to the requirements from different demonstrator cases of GOFLEX
- Develop generalized interfaces for integrating the platform with a variety of energy management systems (e.g., EV EMSes) of different actors of the European Electricity Market system.
- Develop interfaces for integrating with the cloud-based data provisioning system (WP5).

The figure below sketches the TOTALFLEX and KIBERnet solution integration targets, as well as additional improvements and extensions that will be implemented in WP2 for ATP.

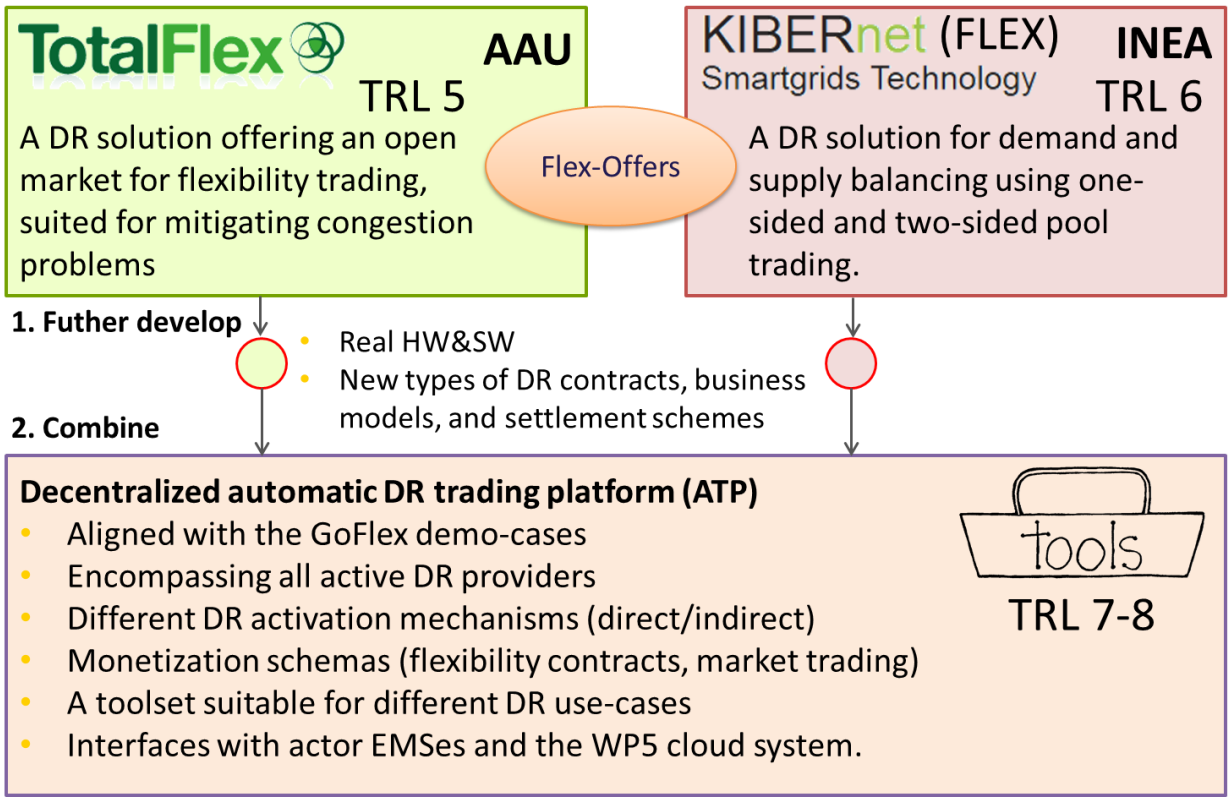

Figure 1: Solution Integration Diagram

In the following sections, we first describe the most relevant concepts to be utilized in ATP, including the concept of Flex-Offer. Then, we provide overviews of the TOTALFLEX and KIBERnet family solutions. Finally, we point to the most relevant document and introduce the structure of this document.

1.2 Flex-Offer Concept

The concept of the Flex-Offer was first proposed in the MIRABEL [MIRABEL, 2010] project and then further developed in the TOTALFLEX project. It is also central in ATP, and therefore described next.

Flex-Offer offers a robust and generic way to describe flexibility in electricity consumption and production of various DERs. An advantage of Flex-Offer is that it explicitly specifies available DER flexibility in a generalized way. Later, a number of Flex-Offers can be efficiently aggregated and disaggregated across various dimensions, e.g., different classes of prosumers. A single Flex-Offer typically includes:

- **Energy profile**, having a number of discrete slices, specifies electricity consumption and production options over a device's active period of operation, typically in 15min. time resolution;

- **Time flexibility interval** specifies a time period in which device's operation (profile) can be advanced or retarded.
- **Default profile** specifies a preferred / locally optimal consumption profile (a baseload)
- **Price data** specifies (discomfort) prices, e.g., associated to deviations from the default profile.

When using a Flex-Offer, no specific knowledge about the underlying DERs is needed, whether the electrical loads comes from heat pumps, EVs, cold stores, etc. An example of a simple Flex-Offer is shown in Figure 2.

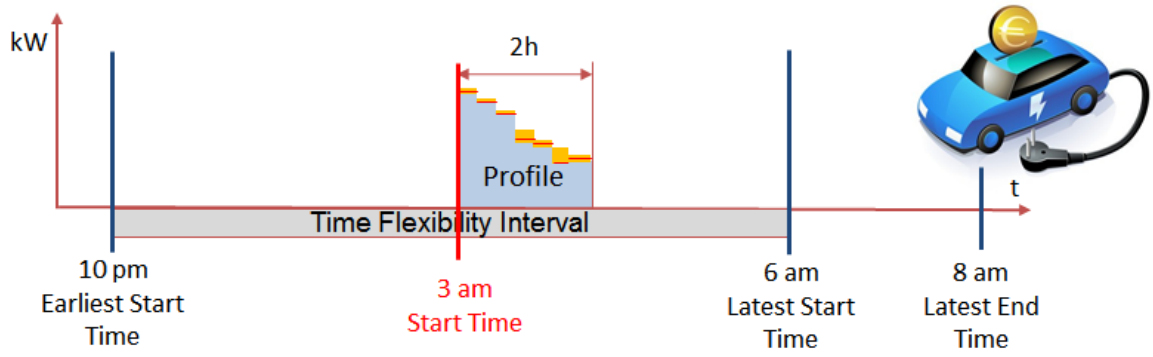

Figure 2: Flex-offer example for charging an electrical vehicle, EV

Figure 2 illustrates an instance of a Flex-Offer for charging an electric car. In this case, the car owner expresses the flexibility by specifying that the vehicle is available for charging from 10 PM until 6 AM, additionally providing its charging curve (which can be automatically obtained by the house gateway). If needed, a price for flexibility can be associated with a Flex-Offer. Specifically, this price can be expressed as a cost paid for 1kWh of energy amount deviation with respect to the reference (baseline) schedule. This yields linear price curves and constraints for DR buyers and sellers, as shown in the figures below.

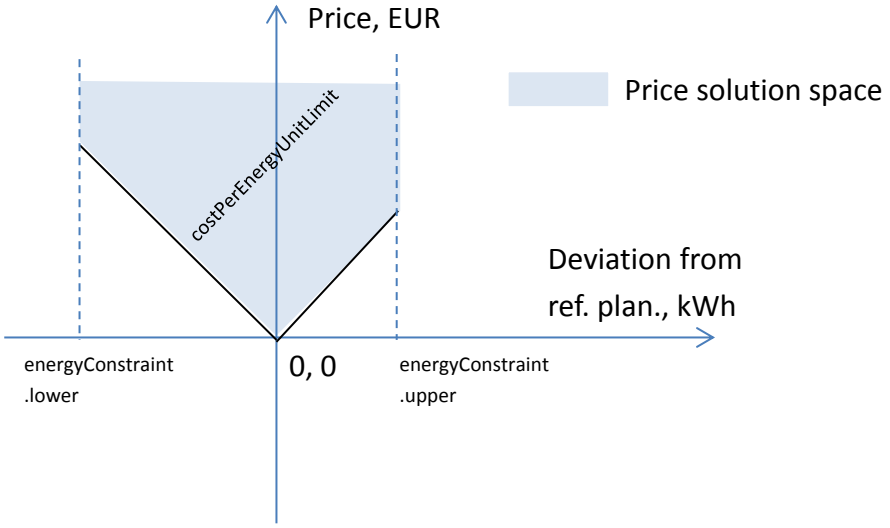

Figure 3: DR seller (e.g., Prosumer) linear price curve / constraint associated to a Flex-Offer slice

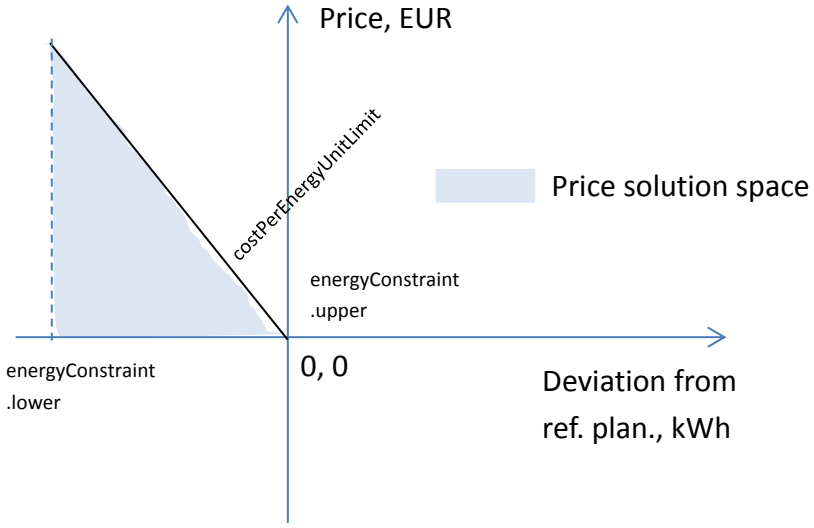

Figure 4: DR buyer (e.g., DSO) linear price curve / constraint associated to a Flex-Offer slice

Flex-Offers from individual Prosumers (e.g., heat pumps, electric vehicles) most often do not represent large flexible loads. Thus, a single such Flex-Offer has low impact and is of little interest for electricity trading. At the same time, optimizing energy loads based on large numbers of Flex-Offers is a computationally hard problem, which requires dealing with many decision variables and constraints originating from many Flex-Offers. By utilizing *aggregation*, flexibilities from individual appliances can be combined and thus offered in a more useful and efficient aggregated form. Such aggregated flexibility can again be represented as Flex-Offers – but with much larger energy amounts and flexibility margins. After aggregation,

schedules are typically assigned to the aggregated Flex-Offers (e.g., based on energy sold on the market). By respecting all inherent aggregated Flex-Offer constraints, a schedule specifies an exact start time and aggregated energy amounts be assigned to a number of underlying Prosumers. Such schedules are disaggregated to a number of schedules for each individual Flex-Offer it is composed of. This operation is denoted as Flex-Offer disaggregation. Disaggregated schedules are finally forwarded to the Prosumers who initially offered flexibility. This Flex-Offer aggregation, scheduling, and disaggregation process is illustrated in Figure 1.

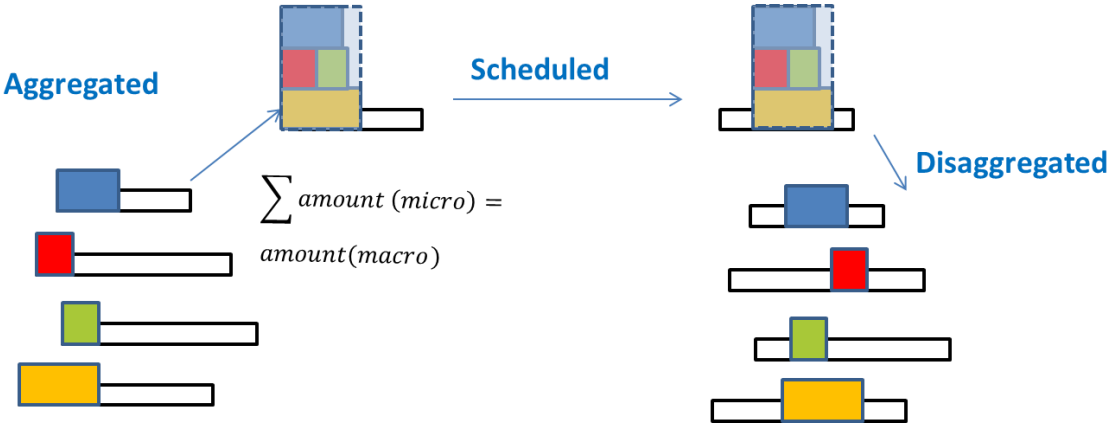

Figure 5: Aggregation/Disaggregation process

1.3 Flex-Offer Lifecycle and Communication Protocol

After Flex-Offer generation at the Prosumer-side, the Flex-Offer is typically sent to a receiving party, potentially, some utility company, BRP, or Aggregator, where is takes part in flexibility negotiation, planning, control, and billing processes, shown in Figure 6.

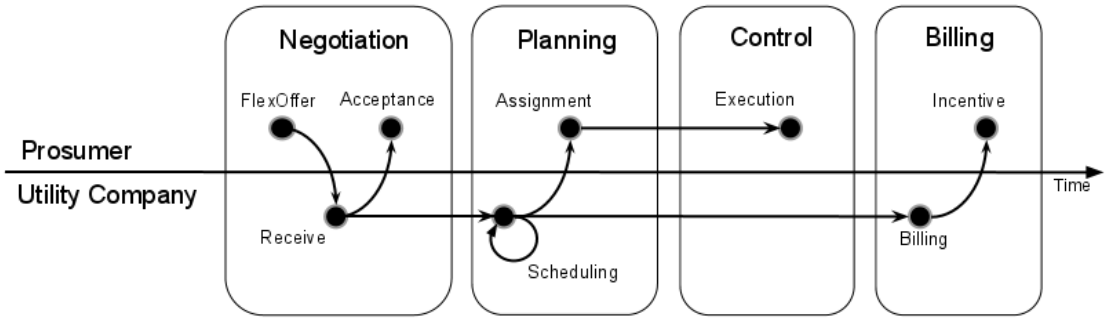

Figure 6: Flex-Offer lifecycle

- **Negotiation process** The Flex-Offer can be accepted, e.g., if all of its attributes are valid and offered flexibility is valuable for the receiving party. On the other hand, the Flex-Offer can be rejected, e.g., due to some validation errors, which then requires updating and resending

the Flex-Offer or, simply, operating Prosumer processes under the default profile (baseload).

- **Planning process** As mentioned earlier, the Flex-Offer can be decomposed into a number of decision variables and constraints, and used in actor-specific optimization and planning process. This results into one or more Flex-Offer schedules, i.e., assignments, which respect all Flex-Offer constraints and can be executed by the Prosumer.
- **Control process** Each Flex-Offer schedule (assignment) sent to a Prosumer is executed, starting at the given *starting time*, such that prescribed energy amounts are consumed or produced at subsequent time intervals.
- **Billing process** Prosumer is rewarded by the flex-offer receiving party for its offered flexibility (Flex-Offers).

Typical message exchange between the Prosumer and Flex-Offer receiving party, covering the negotiation and planning processes, is depicted in Figure 7.

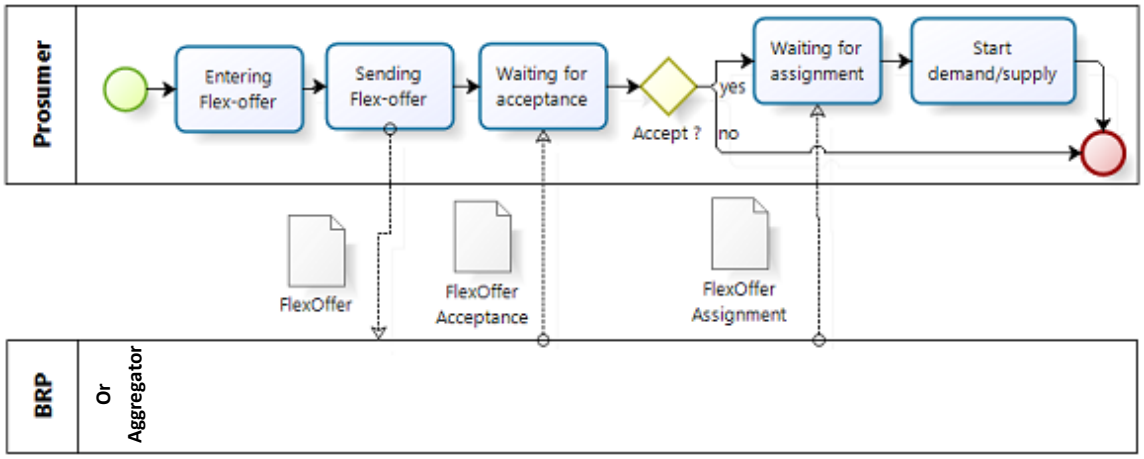

Figure 7: Flex-Offer message exchange

Note, the presented Flex-Offer concept and its representation were originally developed in the MIRABEL and TOTALFLEX projects. Later, in Section 7.1, we will propose additional extensions of the Flex-Offer concept (and accompanying representation), encompassing more advanced energy constraints that have emerged and should be considered in GOFLEX. In WP2, the support for all such Flex-Offer extensions will be considered, taking into account the loads available in the demonstration sites.

1.4 Overview of TOTALFLEX

TOTALFLEX is a project funded under the Danish ForskEL programme. It designs a cost-effective market-based electricity system [S. F. Barot et al. - 2015] where:

- Any size of flexible consumption and production is supported;
- Power customers and/or producers are rewarded by the highest bidder for the flexibility provided;
- The electricity system is better balanced, resulting in lower prices for system services;
- DSO can postpone grid reinforcement due to bottlenecks as congestion are mitigated;

To capture and exchange flexibility, TOTALFLEX uses the presented Flex-Offer concept and Flex-Offer exchange protocol. Flexibility (Flex-Offer) exchange between Prosumer and Aggregator, its activation, and pricing is governed by an open flexibility contract, which specifies how Prosumers are rewarded by Aggregators for their offered flexibility. Based on such a contract, Aggregators first calculate flexibility portfolio price (see Figure 8), calculate available flexibility (see Figure 9), and finally generate meaningful bids for trading aggregated flexibilities on various markets. In TOTALFLEX, a novel localized DSO-centric flexibility market has been developed, when flexibilities in the form of Flex-Offers with associated linear prices curves (see Section 1.2) are traded. Finally, individual flexible Prosumers are rewarded, typically, based on total time and amount flexibilities, as well as total quantity of flexible energy (in kWh) offered to the Aggregator. Thus, in TOTALFLEX, Prosumers passively trade flexibilities by delegating all trading activities to an Aggregator. Therefore such a flexibility trading mode is denoted as delegated, as it will be supported by ATP.

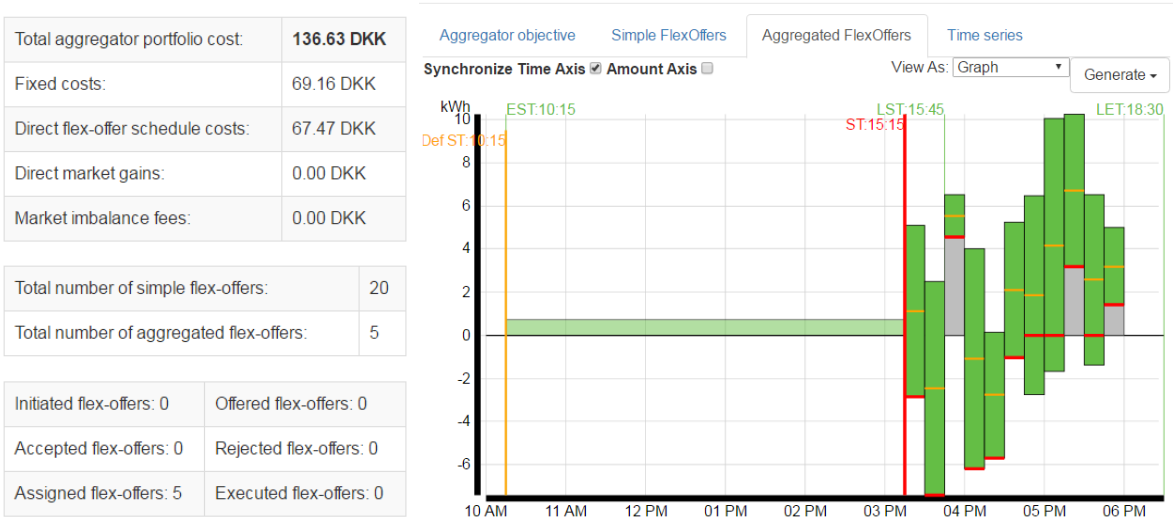

Figure 8: An examples of the TOTALFLEX aggregator management console

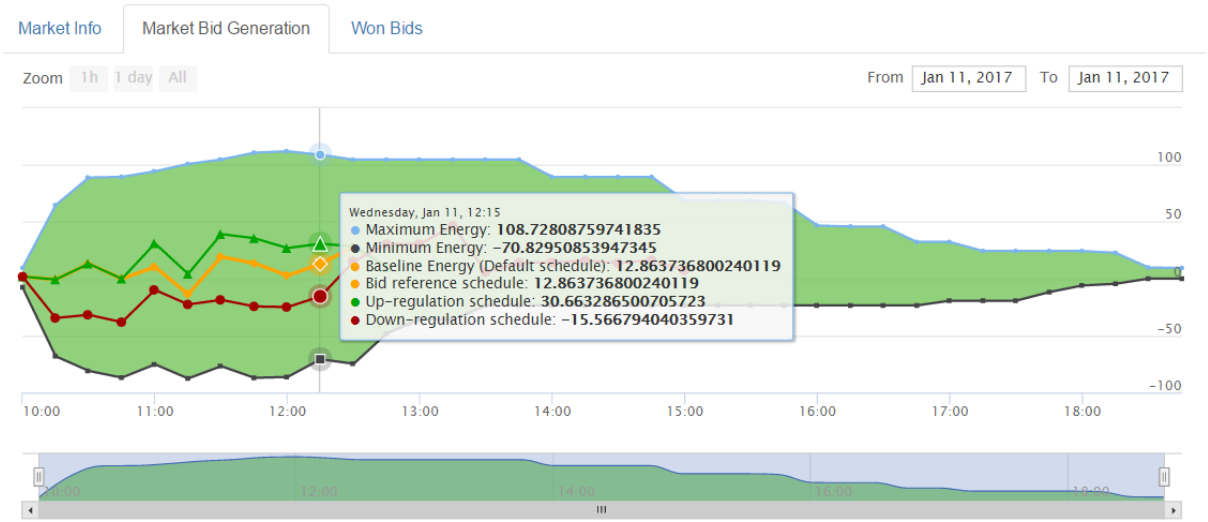

Figure 9. An examples of the TOTALFLEX aggregator's flexibility overview window

Techniques developed in the TOTALFLEX project will be adopted, further enhanced, practically demonstrated, including the techniques for data collection, pre-processing, and Flex-Offer generation.

1.5 Overview of KIBERnet

KIBERnet system is a realization of the virtual power plant whose task is to reduce the imbalances on the network. With centralized control of consumers and distributed producers on the distribution electricity network it enables the user (system operator) to adapt the energy flows and stabilize the electricity grid.

Within the project with the same name, cofounded by the European regional foundation, the system was realized in the prototype controlling 4 large industrial consumers with total of 20MW of installed consumption power and 5MW of the adaptation capacity.

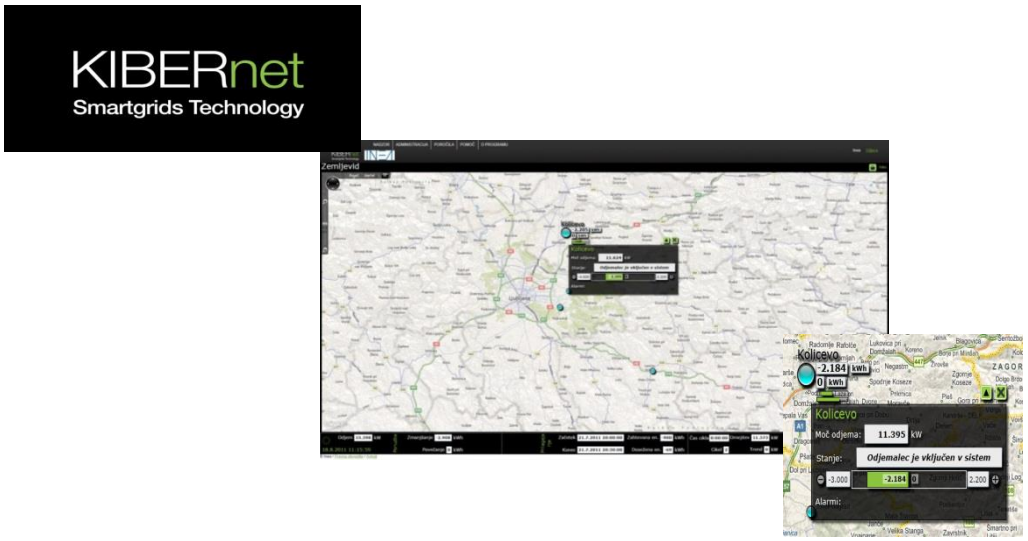

Figure 10: KIBERnet virtual power plant control center

1.5.1 System architecture

Architecture of the System KIBERnet consists of the two parts: 1) local control and monitoring system at prosumer and 2) demand response control center.

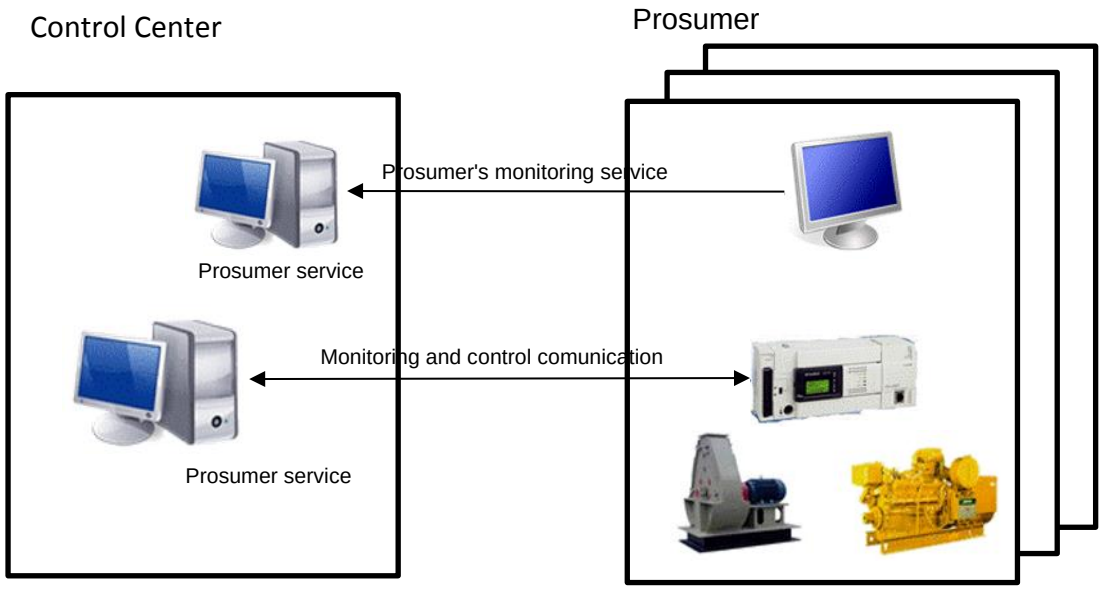

Figure 11: Architecture of the KIBERnet System

There are several prosumers' local controlling systems connected to the demand response control center. Each prosumer's system has its own controlling algorithm for the calculation of the adaptation capacity and execution of the demands. Each prosumer has access to the prosumer's service in control center for monitoring the service actions.

The control center controls all the prosumers on one side and offers the demand response services to the user on the other side. The user – TSO or DSO – uses the system to perform certain system services it is responsible to.

1.5.2 Control center – prosumer communication

There is a two way communication process between control center and prosumer. The prosumer according to its operation state calculates its adaptation capacity for a certain period ahead (1 hour) and sends it to the control center. The offer besides the energy capacity contains also the price information which should refund in the case of the intervention.

The control center aggregates the received offers and calculates total adaptation capacity of the "virtual power plant".

According to the grid situation and forecasted imbalance the user enters the necessary imbalance information which beside the imbalance schedule of energies contains the limit price, which is still acceptable for intervention. The optimization algorithm selects the available prosumers' offers, which are within required constraints and sends the demand schedule to the prosumer. After the confirmation of the demand the prosumer starts the adaptation.

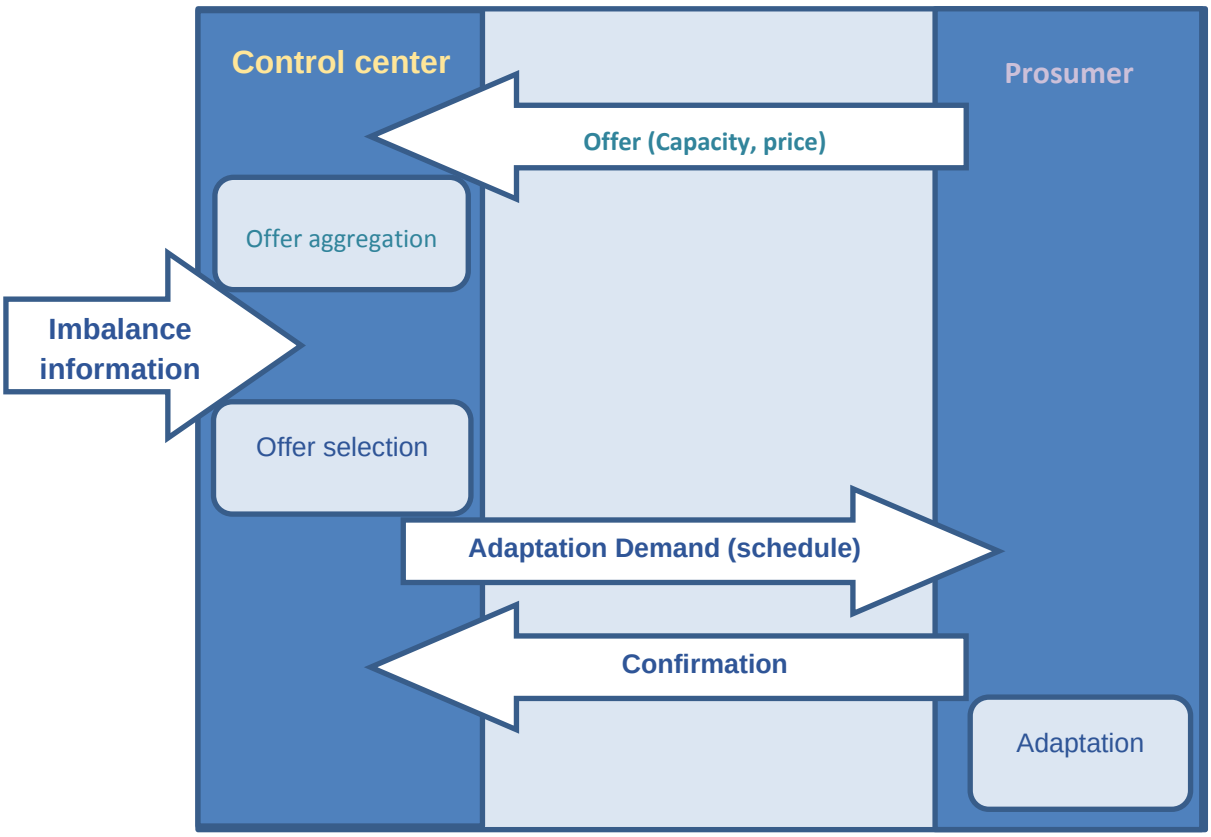

Figure 12: Communication between control center and prosumer

1.5.3 Offer aggregation and selection

Received adaptation offers are aggregated according to the price. Before the selection process the offers are sorted into the price-volume diagram. The entered user's imbalance information provides the requested volume and estimated cut-off price which triggers the selection algorithm.

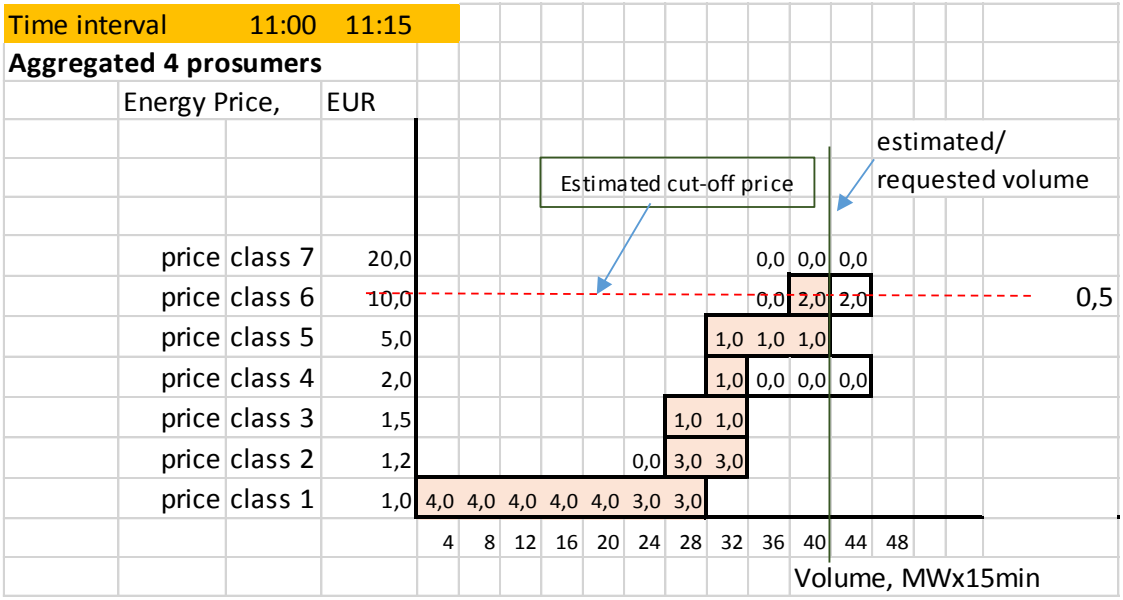

Figure 13: Sort of flex offers onto the price-volume diagram

The selected adaptation offers are disaggregated, transformed into the adaptation demands and sent to corresponding prosumers.

1.5.4 Adaptation evaluation

The evaluation of the executed prosumer's adaptation is based on the characteristic consumption. The control center measures the prosumer's total consumption. Based on past measurements the prosumer's most probable consumption for the time period of the intervention is calculated and compared with the measurement. The difference is taken as the adaptation realization.

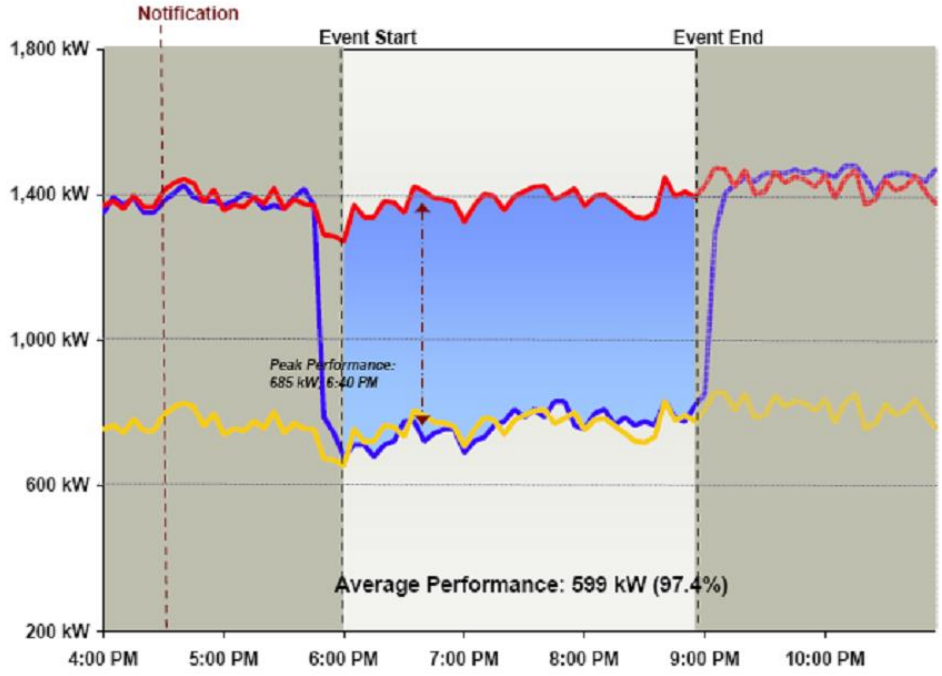

Figure 14: Adaptation evaluation: yellow – adaptation capacity, blue – measurement, red – consumption calculation if there is no adaptation, blue area – adaptation realization

1.6 Overview of KIBERnet FLEX

KIBERnet FLEX is an enhancement of the KIBERnet product, which additionally contains some of the key elements from MIRABEL project [7]. The enhancements are marked with red text in Figure 15.

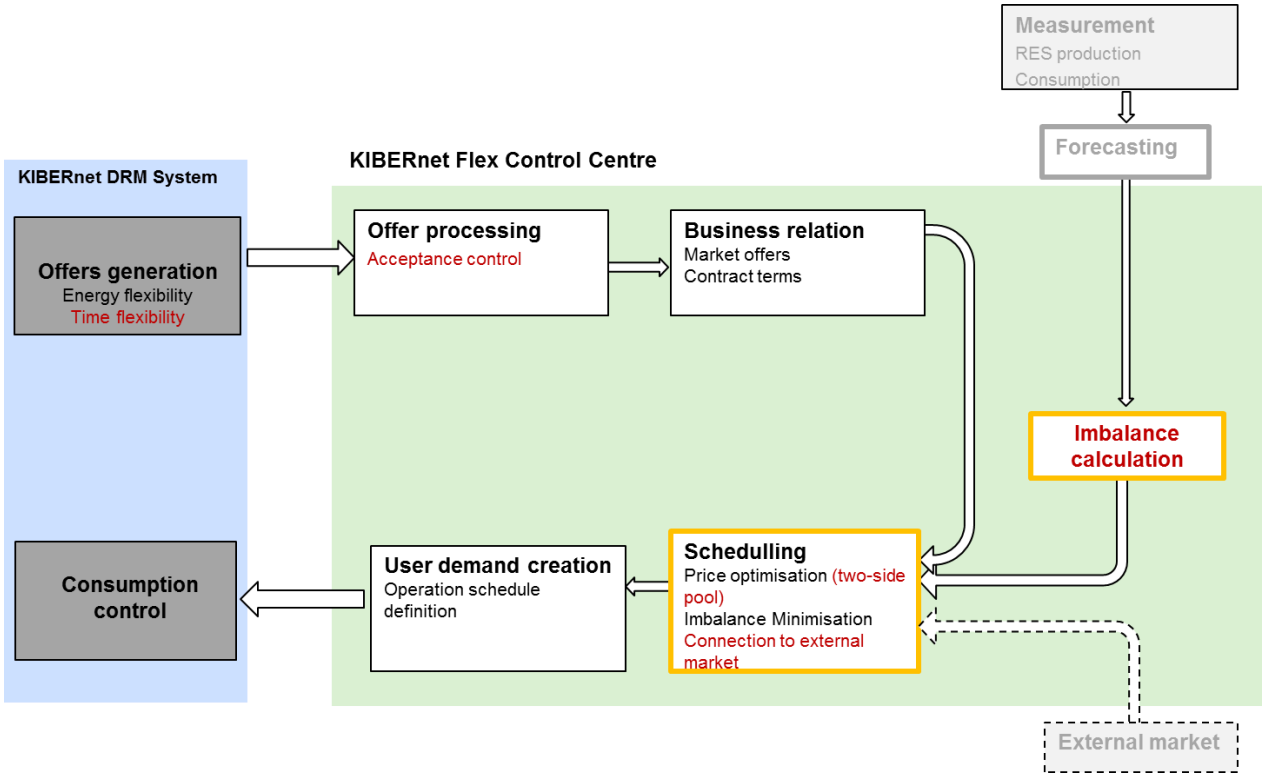

Figure 15: Elements of the KIBERnet and KiBERnet FLEX

- Flex-Offers were enhanced with additional time flexibility parameters like “start before”, “start after” and “assignment time”. These enable the explicit specification of time flexibility. The time span of the Flex-Offer was increased up to 24 hours.
- The acceptance control is added to the offer processing, for checking the consistency of Flex-Offers.
- The system enables automatic calculation of an imbalance which is an input and a trigger for the scheduling
- Scheduling algorithm uses the two-sided pool, which enables many-to-many matching of Flex-Offers.

All these extensions enable the connection and trading on the external market and allow comparing Flex-Offers with other types of bids on the organized markets in the economical terms.

1.7 Related Documents

The relevant documents for prosumer acquisition and integration are D7.1, D8.1 and D9.1. The uplink to the GOFLEX system is done through exchanging Flex-Offer and grid information with the service platform described in D5.1 and the Distribution Observability and Management System (DOMS) described in D4.1. The interaction between GOFLEX components is coordinated within D6.1. Further, the document D6.1 also contains the specification for xEMS and prosumers classification required by ATP.

1.8 Document Structure

In this document, we first overview the tasks of WP2. Then, we describe functionality and data to be provided to other GOFLEX sub-systems and actors, followed by the functionality and data to be consumed by ATP from other GOFLEX sub-systems. Then, we detail functional and non-functional requirements applicable to ATP, followed by initial design considerations. Lastly, we propose an implementation plan and explain how ATP will be used in the GOFLEX demonstrations.

2 Work Package Description

WP2 will develop ATP by following the methodology of the mixed sequential and iterative development, as shown in the work plan below.

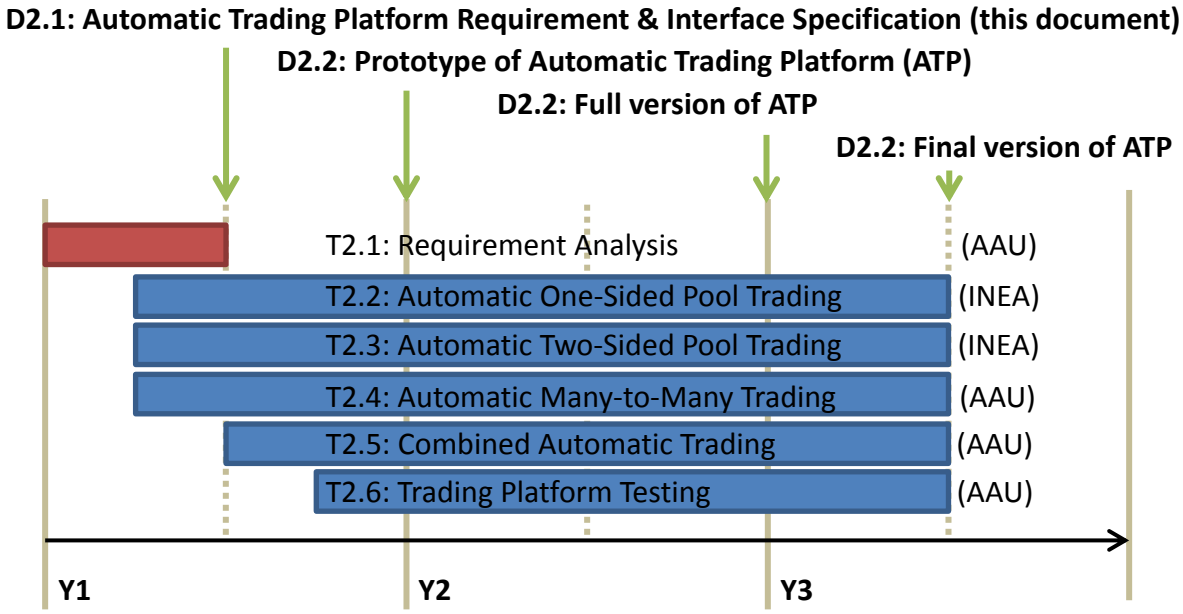

Figure 16: The work plan of WP2

First, ATP requirement analysis will be performed in Task 2.1, leading to D2.1 (this document). The purpose of this task is to clarify main concepts and open issues, as well as provide requirements and initial design consideration for the ATP.

The results of Task 2.1 will serve as input developing ATP instances for different GOFLEX trading cases in Tasks 2.2-2.4. All these tasks with focus on adapting the KIBERnet (FLEX) and TOTALFLEX solutions to the requirements defined in Task 2.1, developing the instances of ATP for the two aforementioned *direct* and *delegated* trading modes. In Task 2.5, the efforts of Tasks 2.2-2.4 will be combined, providing a single integrated ATP that offers the prosumer

a number of trading options, which allow maximizing the value of their flexibility in different trading scenarios.

Finally, Task 2.6 will perform extensive testing of the different ATP instances, to be developed in Tasks 2.2-2.5. The tests will be based on simulated DR providers, intermediaries, and users. Testing results (produced in this task) together with evaluation results from integrated real-world trials (WP6) will be carefully studied, producing new sets of requirements for the succeeding version (generations) of ATP.

This work package will deliver three software (combined with hardware) artefacts, namely *the prototype of ATP*, the *full version of ATP*, *the final version of ATP*. These correspond to different evolutions of the combined ATP to be developed in Task 2.5. These artefacts will be elaborated in Chapter 8.

3 Provided to other work packages / components

As described earlier, WP2 will combine and further develop individual TOTALFLEX and KIBERnet family solutions, producing ATP as an outcome of this work. ATP will be provided in the form of a system of systems (SoS), consisting of the following sub-systems: *Flex-Offer Agent*, *Flex-Offer Manager*, and *Flex-Offer Market*. These three independent sub-systems will ultimately be provided to other work packages either in one piece (the complete ATP) or one-by-one. In this chapter, we overview these ATP sub-systems and describe their exchanged data and offered functionality.

3.1 ATP Sub-system Overview

WP2 will develop ATP as three independent core sub-systems: *Flex-Offer Agent*, *Flex-Offer Manager*, and *Flex-Offer Market.*

Flex-Offer Agent (FOA) is a Prosumer-level ATP sub-system that exposes and makes all kinds of flexible loads available for DR or trading applications. The main function of FOA is to produce meaningful Flex-Offers (e.g., from the available xEMS data), handle their exchange with Flex-Offer Manager, and ensure proper execution of retrieved Flex-Offer schedules. FOA depends on an energy management system (EMS) which controls physical loads and provides all data needed to generate Flex-Offers. In the presence of an existing EMS, FOA is just a simple software and/or hardware component used as a plug-in or an extension of the EMS. However, in the case of no EMS, FOA might integrate the functionality of a basic predictive EMS and support both *delegated* and *direct* trading modes. FOA is expected to be highly extensible to integrate with the variety of different xEMSes and/or physical devices.

Flex-Offer Manager (FMAN) is an ATP sub-system to be, potentially, used by Aggregators, BRPs, MGRs, for managing (potentially large) collections of flexible loads. It integrates ad-

vanced Flex-Offer aggregation and disaggregation functionalities, optimization, as well as GUI-based analytics, which allows its users effectively and efficiently analyzing, trading, and shaping available flexibility in near real-time.

Flex-Offer Market (FMAR) is an ATP sub-system offering various flexibility trading options, potentially for the use by BRPs, MGRs, and third-party flexibility marketers. It provides functionalities of single-, two-sided pool trading, and many-to-many trading, suitable for different flexibility trading scenarios of demand-supply balancing, congestion management, etc. FMAR consumes a number of supply and demand bids, additional trading parameters, and finally allocates energy amounts in eco-socially effective and fair way. As it will be presented in Section 7.5, FMAR in practice will either take a form of a stand-alone service or be integrated as a library.

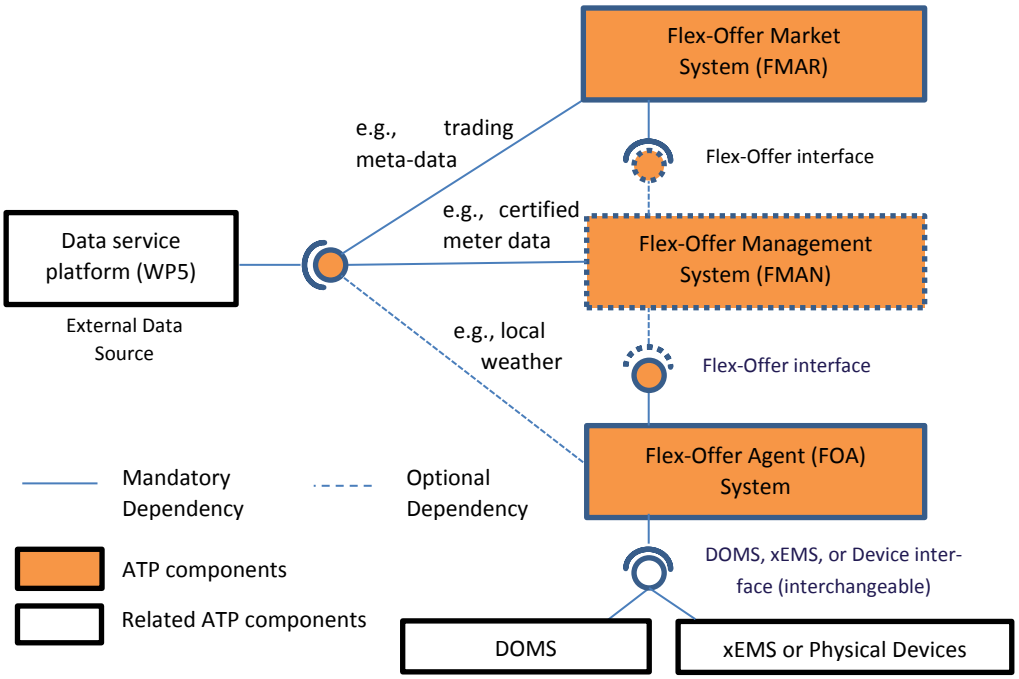

Figure 17: Dependencies and connections between ATP internal and external components

Figure 17 shows dependencies and connections between these three ATP sub-systems and the two most relevant external GOFLEX platform components – Data Service Platform (SP), Distribution Observability and Management System (DOMS), and xEMS. As it can be seen in the figure, at the in-bound side, FOA may connect to an existing DOMS or an xEMS system in order to obtain (most of the) data needed to generate meaningful Flex-Offers from flexibility demand and supply sides, respectively, and to schedule local execution of Flex-Offer schedules. Alternatively, as discussed previously, FOA may integrate basic EMS functionality and connect to a number of physical devices directly. For DOMS and all kinds of xEMSes and physical devices, FOA will be provided with a number or specialized extensions to handle a particular kind of load and/or a device protocol. At the out-bound side, FOA may connect to FMAR either directly or via FMAN, utilizing the generalized *Flex-Offer interface*. Typically,

FOA needs to be connected to FMAN in the cases of delegated trading or the direct trading when prosumer flexible loads are too small to enter the market directly and need to be aggregated. For larger flexible load offers or requests, FOA can be connected directly to FMAR. Additionally, FOA, FMAN, and FMAR depend on Data Service Platform (WP5) for acquiring external data, e.g., localized weather data, certified meter data, congestion points, and exporting summaries or anonymized Prosumer data, e.g., KPIs, sub-meter data. For that, ATP sub-systems comply with the interface requirements of the Data Service Platform (WP5). Data exchange between the sub-systems will be elaborated in Chapters 3-4.

3.2 ATP Sub-system Mapping to the Market Roles

The ATP subsystems FOA, FMAN, FMAR can be, potentially, used by an entity (company) playing one or more of the roles of European Energy Market. The mapping between ATP sub-systems and such roles is summarized in Figure 18.

FOA is to be used by the Prosumer and DSO roles. In practice, this means that FOA can be, potentially, installed locally at Prosumer or DSO premises or simply run and process Prosumer local data externally, e.g., in the cloud. The physical deployment of FOA is highly dependent on data privacy requirements and technical capabilities of existing Prosumer or DSO installation.

FMAN can be used by the majority of the market roles, potentially, by any role that requires (dis-)aggregation of flexible DERS and can hold a legal (flexibility) contract with Prosumers. In practice, FMAN is a software package that can either be deployed locally at the premises of a specific role player or run in the cloud as a stand-alone Flex-Offer Manager service.

FMAR can also be used by the majority of the market roles, potentially, by any role that aims to setup a flexibility trading market place for one-to-many trading. For many-to-many trading, FMAR has to be run by an independent party so that trading processes will not be circumvented and compromised. In practice, FMAR is a software package, which will run locally at the premises of a specific role or in the cloud as a stand-alone Flex-offer Market service.

ATP sub-systems	Prosumer	Aggregator	MGR	BRP	DSO
Flex-Offer Market System		+	+	+	+
Flex-Offer Management System	+	+	+	+	
Flex-Offer Agent System	+				+

Figure 18: Mapping between ATP sub-systems and European Electricity Market Roles

3.3 Functionality

ATP supports trading in so-called *delegated* and *direct* trading modes (see Section 1.4-0). These two general trading modes will be provided for use in the GOFLEX demonstration cases (WP7-9). We now give typical use-case examples demonstrating how delegated and direct trading modes enable business interaction between electricity market roles. For each of these trading modes, we first describe a single-time installation phase followed by subsequent continuous Flex-Offer generation, negotiation, planning, control, and settlement phases.

3.3.1 Delegated trading use-case

Installation phase

A Prosumer signs an *open flexibility contract* with an Aggregator. Among others, the contract specifies the conditions of equipment installation, use of private data, flexibility activation, pricing, and settlement. Later, either Prosumer itself or Aggregator's technical staff installs required FOA equipment and/or setups and tests required data communication between FOA, FMAN, and, potentially, EMS – either built into FOA or external xEMS. Then, the Prosumer has to configure flexible loads using the provided GUIs of FOA and/or EMS, e.g., by providing flexibilities or loss in comfort that he or she tolerates.

Flex-Offer generation and negotiation phases

When configured correctly, FOA can be triggered by an EMS (built-in or external xEMS) or FMAN. When triggered, FOA updates its flexibility offering and generates (or updates) a Flex-Offer, which typically captures a default schedule of the device's future operation together with associated time and energy flexibilities of a potential deviation from this default. Note,

no price information is included into this Flex-Offer. This Flex-Offer is then sent to FMAN. If FMAN accepts this Flex-Offer, FOA prepares for adaptation and waits for one or more of schedules to be received from FMAN for this Flex-Offer.

Planning phases

After receiving the Prosumer's Flex-Offer, FMAN values this Flex-Offer in accordance to the open flexibility contract and then aggregates it with other Prosumer Flex-Offers. Later actions of FMAN depend on the *planning mode* set by the user (Aggregator). For example, if the objective is set to *demand supply balancing*, FMAN will automatically schedule aggregated Flex-Offers in the economically most effective way to balance predicted demand and supply, e.g., by invoking one of the one-to-many trading algorithms built into the FMAR. Alternatively, if the planning objective is set to *profit maximization*, FMAN will automatically and continuously look for opportunities of trading flexibilities specified by the aggregated Flex-Offers. If FMAR is available as a stand-alone service connected to FMAN, FMAN will shape and automatically generate and submit a market bid once FMAR bidding is open. As part of this, FMAN might need to (re-)shape and price the aggregated Flex-Offers to take the required form of the FMAR's market bid. For this, the Prosumer open flexibility contracts, existing Flex-Offer schedules, and won bids will be used as a reference. After FMAN completes the Flex-Offer optimization (based on a local objective or trade deals), FMAN generates schedules for each aggregated Flex-Offer. There, are then disaggregated and delivered back to individual FOAs, including to the FOA of the Prosumer in this use-case.

Control phase

The service platform and the Prosumer's FOA will continuously stream (sub-)meter data to FMAN. This data will be used to validate execution of Flex-Offers and, if needed, making fine-grained load adaptations by sending updated Flex-Offer schedules to the Prosumer FOAs. If needed, this factual data will be used for calculating the reward to the prosumer in accordance to the open flexibility contract, i.e., billing the Prosumer. In this process, uncertified sub-meter data undergoes validation and special treatment to minimize the risks of fraud.

Settlement phase

The settlement takes place periodically, e.g., every month. The Aggregator (the user of FMAN) receives revenues from the parties (e.g., DSO or BRP) that bought adaptation capacity on the market, and only if the bids were fulfilled within the specified margins. The Aggregator also encounters expenses associated to activating Prosumer loads under open flexibility contracts. The Aggregator calculates such revenues and expenses and makes the final bills for the contracted parties. Depending on the flexibility contracts, the Aggregator might

share its revenues with Prosumers (no losses are possible) or pay strict rewards based on flexibility constraints (losses are possible).

3.3.2 Direct trading use-case

Installation phase

The main difference to the delegated trading is in the point when the flexibility contract with the conditions for flexibility activation, pricing, and settlement is defined. The contract usually does not specify the flexibility availability, activation obligations and financial refunds since this is left to the actual market conditions. At the installation, the Prosumer has to, beside other parameters, configure also the adaptation price on the flexible loads using the provided GUIs of FOA and/or EMS.

Flex-Offer generation and negotiation phases

Similarly as at delegated trading, also at direct trading the FOA is triggered by an EMS (built-in or external xEMS). The trigger also contains the information about the importance of the comfort loss (the parameter is called priority) at the adaptation which is converted into the price. When triggered, FOA first updates its flexibility offering, which contains also price for the adaptation. Then, it generates (or updates) a Flex-Offer, which typically captures a default schedule of the device's future operation together with associated time and energy flexibilities of a potential deviation from this default.

Also the FOA connected to the DOMS converts its network information (imbalance, congestion, etc.) into the Flex-Offer containing the time series of energies and prices.

Planning phases

The aggregation of Flex-Offers at FMAN takes into account also price parameter. Next action - scheduling is based on economical optimisation making the maximal profit for the aggregator. The required input is Flex-Offers from the generation and negotiation phases.

Flex-Offers are scheduled in the order of the lowest price for production first and larger price for consumption first. Only the Flex-Offers which increase the profit (or reduce the cost) are candidates for scheduling. The remaining Flex-Offers are either postponed or rejected. Similarly as at delegated trading, the FMAN will continuously compare the flex offers to the opportunities on the open/organized market. FMAN will shape Flex-Offers and automatically generate and submit a market bid if its offer competes the market situation. The price defined in the bid is sourced from the flex offer.

After FMAN completes the Flex-Offer optimization, FMAN generates schedules for each Flex-Offer. These are then disaggregated and delivered back to individual FOAs, including to the FOA of the Prosumer in this use-case.

Control phase

Similarly as in delegated trading, the service platform and the Prosumer's FOA will continuously stream (sub-) meter data to FMAN. This data will be used to validate execution of Flex-Offer demands and later calculating the reward to the prosumer.

Settlement phase

The settlement process is similar as at delegated trading. The Aggregator (the user of FMAN) receives revenues from the parties (e.g., DSO or BRP) that bought adaptation capacity on the market. The Aggregator also encounters expenses associated to activating Prosumer loads under open flexibility contracts. The Aggregator calculates such revenues and expenses and makes the final bills for the contracted parties. Depending on the (marketing) policy, the Aggregator might share its revenues with Prosumers or pay exact amount based on flexibility price parameter.

3.4 Data

ATP exchanges different kinds of data *internally* between FOA, FMAN, and FMAR and *externally* with other GOFLEX systems and actors such as Data Service platform (SP), xEMS, and a (human) user. The types of the most essential exchanged data related to ATP are summarized in the table below. Note, the full overview of the exchanged data in GOFLEX can be found in D6.1.

	to User (GUI)	to FOA	to FMAN	to FMAR	to SP	to xEMS	DOMS
from User (GUI)	-	FOA configuration and load-specific parameters (e.g., comfort constraints)	Commands, queries	-			
from FOA	Settings, status, basic statistics, history data	-	Flex-Offer insert, update, and delete messages, related data, e.g., (sub-) meter measurements		-	Control parameters, schedules	Flex-Offer acceptance feedback
from FMAN	Status, query results, flexibility overviews, KPIs, etc.	Flex-Offer acceptance, assignment messages	-	Aggregated Flex-Offers as bids	-	-	
from FMAR	Status, KPIs, etc.	Flex-Offer acceptance, rejection, assignment messages as winning/loosing bids		-	-	-	
from SP		Weather predictions, smart-grid data for flex. provisioning	smart-meter data for settlement, congestion locations for a meter-id, forecasted (aggregated) open-contract (non-flexible) consumption	Grid data (e.g., registered congestion locations), cost of energy transfer			
From DOMS	-	-			-	-	-
from xEMS		Raw Flex-Offer parameters, full MPC (e.g., state-space) models, or MPC model parameters	-	-			

Figure 19: Data exchanged between ATP sub-systems and external GOFLEX systems and users

Based on Figure 19, we now overview data *actively provided* by ATP sub-systems to other GOFLEX systems.

FOA reports status, show various statistics about its state to the user (e.g., prosumer). Depending on whether FOA is configure to be connected to FMAR directly to via FMAN, FOA generates and then provides Flex-Offers and related data by sending (insert, update, delete) messages either to FMAN or FMAR. If FOA relies on xEMS, FOA will send control parameters, e.g., Flex-Offer schedules with concrete energy amounts to be consumed. Based on these, the underlying physical device and/or process will be locally optimized or controlled by xEMS. In the case of DOMS, FOA will indicate whether the requested energy amount reduction offers was accepted, rejected, or assigned specific energy amounts on the market (analog to the xEMS case).

FMAN provides comprehensive summaries and overviews through the provided GUI to the user (e.g., an aggregator analyst). FMAN allows the user posing queries on flexibility asset data, and so FMAN provides results of such queries to the user. FMAN issues Flex-Offer acceptance, rejection, and assignment messages to FOA, which are then translated to xEMS control parameters by FOA. Finally, FMAN transforms the flexibility portfolio (prosumer Flex-Offers) into a form suitable for trading on the market (e.g., aggregated Flex-Offers). These are sent by FMAN to FMAR as market bids.

FMAR reports status and KPIs to the user (market operator). As FMAR and FMAN shares the common Flex-Offer interface, FMAR sends Flex-Offer acceptance, rejection, assignment messages to all active market participants, similar to FMAN. These indicate winning/loosing bids and assigned energy amounts.

Note, FOA, FMAN, and FMAR can *passively provide* different kinds of data on-demand to other external systems and/or users through specialized APIs. For this, access control rights apply, and they are granted by the ATP sub-system user.

All this data will be further detailed in GOFLEX.

4 Depends on other work packages / components

Due to fault tolerance, robustness, and flexible deployment, our intension is to design the ATP sub-systems to be as autonomous as possible so they can run independently to other systems (of GOFLEX). Additional connections to other GOFLEX and external systems are introduced only to enhance functionality and interoperability towards new prosumer/load types, xEMS systems, better load predictions, more accurate settlement schemes, automated market operation, etc. In this chapter, we specify the most important internal and external dependencies for different ATP sub-systems.

4.1 Functionality

In general, ATP can benefit from the predictive functionality of the data service platform (SP) and some grid data management functionality of DOMS. All this functionality may improve or make possible Flex-Offer generation, monitoring, trading, and validation processes within ATP. The specification of such required data is presented next based on Figure 19.

4.2 Data

In the Flex-Offer generation and validation processes within FOA, the following two important kinds of data are required by FOA:

- **Weather data** Localized air temperatures are needed to better predict energy loads in the case of heat-pumps. Wind speed and solar intensity data may be used to predict energy produced by prosumer's wind-mills and photovoltaic system, affecting the prosumer's available flexibility.
- **Smart-meter data** When a certified local smart-meter data is available, it may be used by FOA as an additional (certified) source of information for flexibility prediction and Flex-Offer validation, in addition to sub-meter data.

Energy management, planning, and settlement processes supported by FMAN may also benefit from the following additional external data:

- **Smart-meter data** This type of data is also important at FMAN as it enables more accurate monitoring of prosumer loads and subsequent validation and settlement of Flex-Offers. When both smart- and sub-meter data is available for the prosumer, all three, non-flexible, flexible, and total energy amounts, can distinguished and
- **Price forecasts** This type of information enables additional energy optimization options at FMAN, e.g., sport-price optimization.
- **Grid data** Congestion locations and the mapping of individual prosumers to these locations may allow building flexibility (Flex-Offer) aggregates tailored to these locations. Such data is required for trading in several localized DSO markets.

Finally, in one of several supported trading modes, FMAR may be configured to run a DSO-oriented local flexibility market. For this mode, FMAR requires the following types of data:

- **Grid data** FMAR requires details about local congestion locations and congestion characteristics in the form of desired energy variation and associated price. The congestion locations and its characteristics will be registered by the DOMS to the ATP (via SP). The DOMS (via SP) also provides a list of prosumers that are relevant to the particular congestion location.

- **Cost of energy transfer** FMAR requires the costs of energy transfer between two points/segments in the DSO territory. This is also provided by DOMS (via SP).

Additionally, data service platform (SP) will be the source for the weather data, smart-meter data, and price forecast. All this data will be further detailed in GOFLEX.

5 Functional Requirements

In this chapter, we draw the essential functional requirements, applicable to both the complete ATP and individual ATP sub-systems.

As seen in Table 1, ATP is generally required to provide functionality needed to support two aforementioned trading models: *direct* and *delegated* (see Section 3.3). For these trading modes, ATP is required to use a common representation of flexibility in the form of Flex-Offers – either the original or extended variant tailored to the needs of GOFLEX. For Flex-Offers, ATP sub-systems are required to use a common harmonized Flex-Offer interface to exchange Flex-Offers and all related data. Lastly, ATP should report relevant KPIs, defined by GOFLEX, to quantify ATP performance.

Table 1: General AT functional requirements

Requirement Number	Requirement Description
F2.1	**Direct trading support** ATP should provide all necessary functionality to support the aforementioned direct trading mode.
F2.2	**Delegated trading support** ATP should provide all necessary functionality to support the aforementioned direct trading mode.
F2.3	**Common Flex-Offer representation** ATP is required to use a consolidated common representation of Flex-Offer to represent and exchange flexibility data between ATP sub-systems
F2.4	**Common Flex-Offer interface** ATP is required to provide a common unified Flex-Offer interface to exchange data between ATP sub-systems
F2.5	**Key Performance Indicator (KPI) reporting** ATP sub-systems should provide a number of quantifiable measures of the aforementioned KPIs

Table 2 gives a number of additional functional requirements associated to FOA. As seen in the table, FOA is generally required to handle diverse load types, from the simple household appliances to complex industrial loads. For controlling these loads, FOA should offer either

direct load control via an integrated management system or *indirect* load control via an existing xEMS system. Further, for all these load types and the control modes, FOA should allow the user to monitor, setup, configure loads and associated flexibility (directly or via xEMS). Finally, the FOA should allow the DOMS to submit bids for increase/decrease of energy at grid asset with associated delta energy and price at a resolution ranging from 15mins to 1 hour.

Table 2: FOA functional requirements

Requirement Number	Requirement Description
F2.FOA1	**Diverse load support** FOA has to support different kinds of flexible load - from simple dishwashers to complex factory loads
F2.FOA2	**Direct and indirect load control** FOA has to offer direct load monitoring and control in the cases with no xEMS and indirect load monitoring and control in the cases with an existing xEMS
F2.FOA3	**User configuration** FOA has to allows the user to monitor, setup, and configure loads and associated flexibility
F2.FOA4	**External Data** FOA should have an interface for requesting weather data and smart-grid data from SP.

Table 3 summarized the key additional functional requirements for FMAN. As seen in the table, first FMAN is required to support Flex-Offer aggregation and disaggregation techniques, which allow combining individual prosumer Flex-Offers into their meaningful aggregates (aggregated Flex-Offers), and then effectively de-combine schedules of such aggregated Flex-Offers. These techniques have to be tailored to the adopted representation of Flex-Offer (see F2.3). Further, FMAN is required to support the functionality of the automated planning, where the portfolio of (aggregated) Flex-Offers is continuously optimized against a fixed pre-defined objective function, while automatically taking the advantage of emerging opportunities (e.g., market opening). One such opportunity is trading in one of the pre-defined markets (e.g., FMAR), which requires FMAR to generate meaningful market bids taking risks into account. To calculate Prosumer rewards, FMAN is required to manage electronic flexibility contracts and used them as a basis for pricing flexibility and settling end-prosumers. Finally, FMAN should offer effective portfolio monitoring and analytics functionality (e.g., through effective GUIs) so the operation of FMAN (including KPIs) can be monitored and analysed.

Table 3: FMAN functional requirements

Requirement Number	Requirement Description
F2.FMAN1	**Aggregation and Disaggregation** FMAN has to support aggregation and disaggregation techniques suitable for the adopted variant of Flex-Offer
F2.FMAN2	**Automated Planning** FMAN should continuously (re-)optimize available Flex-Offer portfolios to fulfill one of the defined planning objectives
F2.FMAN3	**Market Biding** FMAN should be able to transform a portfolio of Flex-Offers into a form suitable for trading on one of several pre-defined markets (including the DSO market offered by FMAR).
F2.FMAN4	**Contract Management and Settlement** FMAN should manage electronic Prosumer (or DSO)-Aggregator contracts and use them in settlement processes
F2.FMAN5	**Monitoring and Analytics** FMAN should offer an intuitive visualization (GUI) and associated functionality to monitor and query the state of the flexibility portfolio, KPIs, etc.
F2.FMAN6	**External Data** FMAN should have an interface for requesting smart-meter data, congestion locations for a meter-id, and aggregated non-flexible demand from SP.

Table 4 summarizes essential FMAR requirements. As seen in the table, FMAR is generally required to provide a number of trading options by integrating different trading techniques for specific cases. These trading techniques are to be adopted for demand/supply balancing and congestion management applications in GOFLEX. KIBERnet and TOTALFLEX solutions should be used as a starting point. The trading has to be fully automated, but with the option for manual configuration and monitoring. The user (market operator) must be offered ways to configure trading process parameters.

Table 4: FMAR functional requirements

Requirement Number	Requirement Description
F2.FMAR1	**One-Sided Pool Auction** This encompasses one-sided pool auction where a number of supply offers are traded against a fixed demand from a single DR user (e.g. DSO or BRP)
F2.FMAR2	**Two-sided Pool Auction** This encompasses two-sided pool auction where an aggregate of DR offers (e.g., from prosumer, aggregator) are traded against several competing DR users (e.g., BRP, DSOs).
F2.FMAT3	**Many-To-Many Auction** This encompasses two-sided pool auction where several competing DR providers (e.g., prosumer, aggregator) are traded against several compering DR consumers (e.g., BRP, Aggregators).
F2.FMAR4	**Implicit Capacity Trading** The provided flexibility trading algorithms should accommodate the cost of energy transport, implicitly added to the price of energy flexibility.
F2.FMAR5	**Automatic Operation** FMAR should offer trading functionality in the fully automated fashion.
F2.FMAR6	**User configuration** FMAR should allow the user to monitor and configure trading processes
F2.FMAR7	**External Data** FMAR should provide an interface to allow the DOMS to register grid congestion locations (via SP) and to receive a list of prosumers relevant to grid congestion and bid information data. Further, it should also send a summary of the completed bids on request from DOMS.

6 Non-functional requirements

A number of non-functional requirements are applicable to the sub-systems of ATP, and these are summarized in Table 5.

Table 5: Non-functional requirements

Requirement Number	Requirement Description
NF2.1	**Availability and Robustness** All ATP sub-systems and their internal components should be designed and carefully tested for the 24/7 operation, offering high degrees (99.99%) of availability. ATP sub-systems should be able to cope with sporadic communication errors, erroneous inputs and data, and offer automatic recovery. In any situation, user processes (power consumption), if directly controlled by ATP, should not be affected beyond user tolerable limits.
NF2.2	**Compatibility** ATP systems should be compatible with the existing communication standards most suitable for energy-related data exchange.
NF2.3	**Deployment** ATP sub-systems should run on a common hardware and support a number of deployment options. Specifically, it should be possible to able to install all ATP sub-systems in a single machine, e.g., for testing purposes. Alternatively, it should be possible to deploy ATP sub-systems in the distribution fashion. Generally, it should be possible to tailor ATP sub-systems to the most typical market role installations. Particularly, FOA has to be suitable for the deployment in the cloud or at Prosumer/DSO premises (to prevent leak of sensitive data).
NF2.4	**Extensibility and Maintainability** ATP sub-systems, specifically FOA, should be extensible so future functionality and compatibility with other xEMS not considered in this project will be provided. Further, ATP should be open for other types of flexibility/DR monetization and trading, and not be limited to the specific trading modes demonstrated in this project.
NF2.5	**Scalability** ATP should be able to scale to a large number of electricity market roles, hundreds of users within GOFLEX and millions of users in general.
NF2.6	**Security** ATP sub-systems should be protected from the unauthorized access. Only users owning or having individual sub-systems installed in their premises should have access to the configuration and settings of their individual system instances.
NF2.7	**Sensitive data protection** Sensitive data exchange within ATP sub-systems and external systems should be protected using common security standard (e.g., TLS) and should not leak beyond scopes specified in user (flexibility) contracts.

7 Architectural considerations / assumptions

In this chapter, we provide a number of architectural/design considerations in order to meet all functional and non-functional requirements as well as to facilitate the practical implementation of ATP. All these proposals will be carefully considered in WP2, further refined if needed, and finally used while developing individual ATP sub-systems.

7.1 Flex-Offer Extensions

Section 1.2 overviewed the initial concept of a Flex-Offer that was originally designed and developed in the MIRABEL and TOTALFLEX projects. To be able to practically use Flex-Offers in the GOFLEX context (e.g., capture more complex types of loads), additional Flex-Offer extensions will be considered in GOFLEX, addressing the requirement F2.3. These extensions are summarized below.

Implicit Time flexibility Original MIRABEL/TOTALFLEX Flex-Offer requires an explicit definition of time flexibility using two parameters *StartAfterTime* and *StartBeforeTime*. However these parameters are not suitable for loads which are "always-on" and may be turned off for a certain time interval (example loads are various HVAC systems). In such cases, the traditional Flex-Offer need periodic updates to refresh their availability, even if the state of the load has not changed. The extension implies the following:

- Changing *startAfterTime* and *startBeforeTime* to optional parameters. If the values of these attributes are omitted, they are implicitly assumed to be equal to the current time ("*Now()*").
- Adding a new parameter *startBeforeDuration*, which specified the duration from the current time ("*Now()*") until the latest time when the activation is allowed to take place. This is equivalent to having *startBeforeTime* = *Now()* + *startBeforeDuration* being implicitly computed.

Such implicit time flexibility semantically means that the activation may start anytime from *Now()* to *Now()* + *startBeforeDuration*.

Polytopic Constraints Much flexibility might be lost [Laurynas Šikšnys et al. - 2016] when using the original Flex-Offer for complex types of loads (e.g., battery systems, heat-pumps), particularly, those that exhibit dependencies between energy amount consumed/produced at consecutive time intervals. Therefore, in addition to existing *time*, *energy amount*, and *total energy amount flexibilities*, we introduce an additional (optional) polytopic constraint in the form $A \cdot \boldsymbol{x} \leq b$, where A is an m×n matrix, $\boldsymbol{x}$ is an *n*×1 column vector of energy amounts, and *b* is an *m*×1 column vector of constants. The effect of such constraint has previously been explained [S. F. Barot. et al. - 2015]. This polytopic constraint complements

other existing Flex-Offer constraints and allows capturing any convex set of energy adaptation options in the n-dimensional space $\mathbf{R}^n$, forming the *polyhedron* in $\mathbf{R}^n$.

Dependency-based Flex-Offer This is a specialized form of the polytopic constraints, where the A and b matrices take a special so-called dependency-based Flex-Offer form. Based on recent research [Laurynas Šikšnys et al. - 2016], dependency-based Flex-Offer offers a good trade-off between constraint processing time and the loss of flexibility of the exact polytypic constraints.

Exclusive OR (XOR) profiles A standard Flex-Offer is designed for Prosumers with a load that can be adjusted/controlled in the continuous manner during the *slice period* (e.g. of 15 mins), i.e., programmable to consume/produce any energy amount in the range given by two parameters *energyConstraint.lower* to *energyConstraint.upper*. However, some (typically small) household appliances (e.g., electrical kettle) or specific industrial loads (e.g., an electrical motor that has to either run or remain in the off state for some time period) might be operated in 2 or more discrete operating modes (e.g., ON or OFF mode). In such cases, disaggregation/control errors might occur, e.g., if Flex-Offers schedule requires consuming some value between *energyConstraint.lower* and *energyConstraint.upper*, but the device can physically consume either *energyConstraint.lower* or *energyConstraint.upper*. Therefore, we propose extending the Flex-Offer concept to support multiple exclusive OR profiles, as opposed to supporting only a single one (see Figure 2). A Flex-Offer with multiple profiles has independently specified set of *slices* with associated *energy amount, total energy amount,* and *polytopic constraints*. When planning energy (i.e., optimizing Flex-Offer schedules), the generated Flex-Offer schedule has to meet the constraints of at least one such profile.

Alternatively, the *energyConstraint.lower* and *energyConstraint.upper* constraints of an individual Flex-Offer profile slice may be complemented with the discrete enumeration of all feasible energy amount combinations for a particular time interval (slice). Note, the latter XOR profile representation can easily be converted into the former representation by generating profile alternatives with equal “lower” and “upper” values of the energy constraint for each discrete combination.

Two-mode conditional operation This is a special case of the exclusive or (XOR) profiles. Such a conditional operation is used to conveniently define the operation of the load prior and after intervention, if its Flex-Offer is activated. In this case, a number of mutually exclusive profiles will be produced from 4 element vectors (priorTime, priorPortion, afterTime, afterPortion), where priorPortion and afterPortion are the portions of the total energy assigned in the flexOfferSchedule, which will be returned before and after the intervention, and priorTime and afterTime are times in seconds, which are needed for this modification.

Note, these extensions will require extending existing Flex-Offer aggregation, scheduling, disaggregation algorithms, as well as other Flex-Offer management techniques. Therefore,

their actual implementation and use in GOFLEX will be carefully evaluated. Concrete loads available in GOFLEX and the following general modelling approach will be used to determine the importance of these extensions:

Given any (prosumer) load instance, load flexibility modelling should consider (and use) simple Flex-Offer constraints first before considering more advanced ones, following this algorithm:

- ***Step 1**, Use Flex-Offer **energy flexibility** first (energyConstraint.lower - energyConstraint.upper). If sufficient flexibility is captured, stop. Otherwise - go to Step 2.*
- ***Step 2**, If the load/process exhibits time flexibility, use **time flexibility** (startAfterTime - startEndTime). Use **implicit time flexibility** if needed (see above). If sufficient flexibility is captured, stop. Otherwise - go to Step 3.*
- ***Step 3**, If the load/process requires a fixed or flexible amount of energy delivered by the end of operation, use so-called **total energy constraints** (totalEnergyConstraint.lower - totalEnergyConstraint.upper) constraining the total energy of the operation. If sufficient flexibility is captured, stop. Otherwise - go to Step 4.*
- ***Step 4** If the load exhibits dependencies of energy amounts across time intervals, use the **dependency-Flex-Offer** (first choice) or **polytopic constraints** (second choice). If sufficient, stop; otherwise – goto Step 5.*
- ***Step 5** if the load exhibit discrete alternatives, use the **two-mode conditional** operation (first choice) or **XOR profiles** (second choice).*

Extensions that are required by the majority of loads in GOFLEX will only be implemented.

7.2 Flex-Offer Exchange Interface

In this section, we propose a RESTful interface to be considered in GOFLEX for exchanging Flex-Offer and related data between ATP sub-systems. Our proposed Flex-Offer interface can, potentially, be implemented over a number of standard transport protocols, e.g., HTTP, MQTT, XMPP, depending on the requirements of a specific installation case. The actual Flex-Offer interface to be used by (the versions of) ATP will be, potentially, the variation of the RESTful interface specified in Table 6.

Table 6: Flex-Offer Exchange Interface Resources, Methods, and their Descriptions

URL	Method	Description
/contract	GET	Retrieve an electronic version of a flexibility contract, e.g., a Prosumer-Aggregator contract for delegated trad-

		ing
/bill	GET	Retrieve an electronic bill for offered and acquired DR services, with summaries of factual and predicted quantities
/flexoffer	POST	Create a Flex-Offer on a receiving party. The Flex-Offer might be immediately rejected or accepted. On acceptance, the Flex-Offer with an automatically generated Id will be returned.
/flexoffer	GET	Retrieve all active Flex-Offers, i.e., those that did not yet undergo settlement
/flexoffer/ :from/:to	GET	Retrieve all (historical) Flex-Offers, that we submitted (or updated) in the time period from “from” to “to”
/flexoffer/:id	GET	Read a Flex-Offer with a specific ID
/flexoffer/:id	PUT	Update/replace a Flex-Offer, if this is still allowed, e.g., appliance operation is not yet complete.
/flexoffer/:id	DELETE	Delete a Flex-Offer with a specific ID, if this is still allowed, e.g., appliance operation has not yet started
/flexoffer/stream	SSE/ Websockets	Receive notifications on Flex-Offer attribute changes, e.g., Flex-Offer Schedule change.
/flexoffer/ measurement	POST	Submit the factual energy amount measurements of the Flex-offer load, e.g., from the sub-meter
/timeseries/ measurement/ :from/:to	GET	Retrieve an aggregated time series of measurements for the time period from “from” to “to”.
/timeseries/ baseline/:from/:to	GET	Retrieve an aggregated time series of the baseline loads (before activation) for the time period from “from” to “to”.
/timeseries/ schedule/ :from/:to	GET	Retrieve an aggregated time series of scheduled energy for the time period from “from” to “to”.
/kpi	GET	Retrieve summarized KPI measures

As it can be seen in the table, the interface allows two parties (e.g., FOA and FMAN) exchanging Flex-Offers and related data. Using this interface, the Flex-Offer issuing party (FOA)

can query existing contract data and billing information managed by the Flex-Offer receiving party (FMAN). This information allows, e.g., evaluating the price for offered flexibility. Using this interface, the issuing party is able to submit new or update and delete an existing Flex-Offer, and get notified about changes of the internal Flex-Offer attribute values, e.g., pre-scribed Flex-Offer schedules. Further, the issuing party is able to submit sub-meter data (if available) for validation of the Flex-Offer schedule execution. Lastly, the interface allows querying measurement, baseline, and scheduled energy time series and KPIs, aggregated across a number of issuer's Flex-Offers.

Lastly, in relation to project cross-breeding, the use of USEF interface [USEF - 2017] will be considered in GOFLEX as an alternative for communicating flexibility. This will require a complementary handling of USEF-specific messages like *FlexRequest*, *FlexOffer*, *FlexOrder*, *FlexOfferRevocation*, *D-prognosis*. This will ensure the compatibility and allow integration of ATP with other USEF-based systems.

7.3 Flex-Offer Agent Design Considerations

In this section, we present an initial considered software / hardware architecture of the Flex-Offer Agent (FOA) – one of the ATP sub-systems.

To meet the aforementioned requirements, WP2 aims to develop FOA as an extensible and highly customizable system, consisting of the following three major components:

- **FOA Core** – it is a passive components that just collects data from underlying compo-nents (FOG) over a strict API, forms Flex-Offers, and handles their exchange via the Flex-Offer Interface (Table 6). Also, it ensures that Flex-Offer exchange protocol is strictly followed (see Figure 7).
- **Flex-Offer Generator (FOG)** – it is an active component that can be extended to sup-ports various load-type-specific predictive models. FOG manages instances of such models based on sensor/user data, and handle their conversion to FOs. To create and maintain flexible load model instances, FOG relied on (near) real-time data collected from a number of interfaces – the (graphical) user interface, external data interface, smart-meter interface, and xEMS interface.
- **Built-in EMS** – it is an active component for direct monitoring and control of a num-ber of flexible loads (in cases when no xEMS is available).

The overall envisioned FOA internal architecture is given in Figure 20.

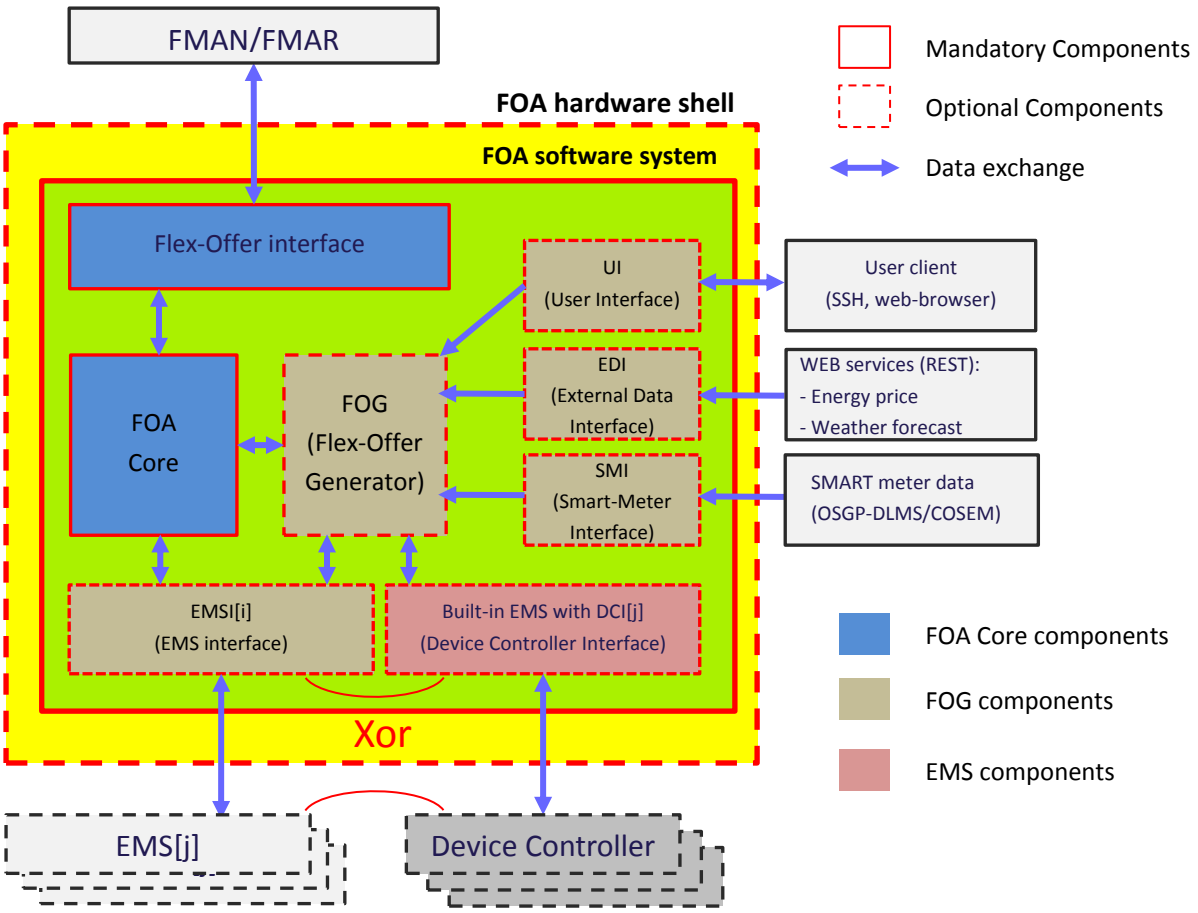

Figure 20: Internal FOA architecture

As seen in Figure 20, the FOA architecture includes a number of additional components. The *Flex-Offer interface* component handles communication and secure exchange of Flex-Offers between FOA and FMAN or FMAR sub-systems. When FOA is used with xEMS, the EMSI components offers a number of standard interfaces to connect with existing xEMSes, primarily for retrieving (near) real-time EMS data - either raw (load, temperature, etc.) sensor measurements or consolidated operational and flexibility data (see *FOA-xEMS interface* in WP3 for details). EMSI is also used for pushing operation (Flex-Offer) schedules to be executed by an underlying xEMS. Note, when no xEMS is available, the collection of raw measurement data and execution of Flex-Offer schedule is performed using the built-in EMS component, which communicates with devices through DCI (Device Control Interface). For FOG to be able to maintain predictive models (or device specific Flex-Offer model) the EDI (external data interface) and SMI (smart-meter interface) components are to be used as supplementary data sources. Respectively, they provide external data (e.g., weather predictions) and measurements from certified smart-meters for flexibility provisioning as well as Flex-Offer validation (settlement). Finally, the UI components handles exchange of data between FOA and an administrating (admin) user, e.g., via SSH or a standard web browser. Specifically, the

UI component allows the user to setup and configure FOA, e.g., by following the proposed process of FOA configuration shown in the figure below.

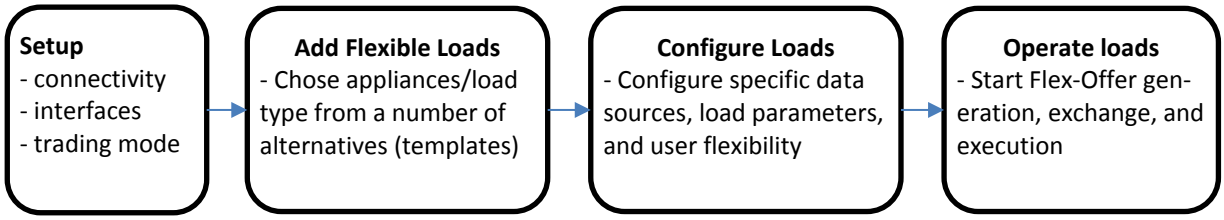

Figure 21: FOA configuration process

Note, each time a new (yet unsupported) type of loads needs to be added, a new FOG extension needs to be developed and registered in FOG to handle the generation of a Flex-Offer and schedule processing in that specific load case.

In summary, as shown in Figure 20, FOA is highly configurable system with a number of optional software components and an optional hardware shell. A specific FOA configuration should be established for a concrete installation site. Generally, we envision three typical FOA configurations/deployment cases in GOFLEX:

- **FOA as software** In this configuration, FOA will be deployed as a purely software solution (e.g., a JAR package) with most relevant components included. Note, it might not be possible to include some of the data source/sink components (e.g., SMI, built-in EMS, EMSI) due to their dependence on a specific hardware platform (HW shell). This FOA configuration is suitable for both *direct* and *delegated* trading modes.
- **FOA as HW box** In this configuration, FOA will be provided as an industry-grade hardware-based solution with a number of standard ports and physical interfaces. In this configuration, FOA software may include all except the built-in EMS component. The latter component is not needed as the load control is handled via xEMS, and no direct load control is pursued. This FOA configuration is suitable for both *direct* and *delegated* trading modes. An example of FOA in this configuration is depicted below.

Figure 22: An example of the FOA in the HW box configuration

FOA as HW box + Smart plugs In this configuration, FOA will be provided as a number of smart-plugs and a controller device running the FOA software. This configuration will be suitable for a narrow class of appliances (e.g., boilers, electrical heaters, freezers) and is to be used for the *delegated* trading mode only. An example of FOA in this configuration is depicted below.

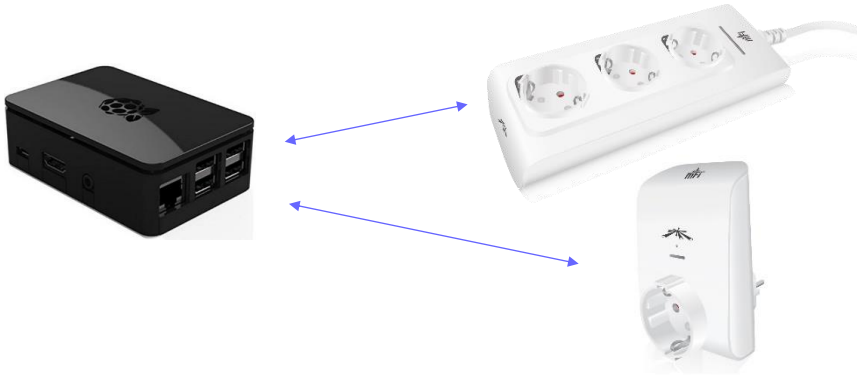

Figure 23: An example of the FOA in the HW box + smart-plug configuration

7.4 Flex-Offer Manager Design Considerations

In WP2, we plan to develop FMAN by combining, merging, and further developing two existing TOTALFLEX and KIBERnet software systems: *Aggregator Manager System* and *inEIS*.

Aggregator Manager System (AAU) is an open-source prototype system from the TOTALFLEX's technology stack. It is capable of automatically collecting, (dis-)aggregating, scheduling Flex-Offers based on one of the pre-defined objectives in the context of the *delegated trading*. Aggregator Manager automatically prices Flex-Offers and their candidate schedule based on an open flexibility contract and different flexibility price components set in the contract. The system is integrated with TOTALFLEX's flexibility market, and so this it is capable of generating and submitting meaningful flexibility market bids. Energy (Flex-Offers) is optimized to fulfil both the winning bids and the objective set. For exchanging Flex-Offers and related data, it can use either XMPP or simple RESTful HTTP interface. In GOFLEX, we plan to increase the overall technology readiness level (TRL) of the system, and introduce new features, e.g., Flex-Offer extensions and new aggregation algorithms, as required by GOFLEX.

inEIS System (INEA) is one of the KIBERnet family systems, which serve as a platform for energy management of (mostly industrial) consumers. The platform can be extended to allow for benchmarking and comparison of similar loads and processes within same or different consumers and producers.

Currently inEIS offers several drivers for measures acquisition. Amongst them are Modbus RTU and Modbus TCP, HTTP, OPC-UA, SNMP and others. Since GOFLEX will need to gather the measurements, we plan to utilize those existing functionalities. Furthermore, the inEIS platform offers simple and efficient visualization using charts and dashboards, which will be used to build the graphical interface and management console for FMAN.

inEIS platform offers several ways of interaction. For GOFLEX, we plan to use WEB GUI, RESTful APIs and notification/alarming engine for E-mail and/or SMS notifications.

Database will be extended in order to account for Flex-Offer management. We will also prepare stored procedures, which will serve the API, defined at the beginning of this chapter. We will explore the use of specific time-series optimized databases to be able to better cope with large amounts of data.

inEIS is based on the following open-source technologies:

- CentOS Linux operating system
- MySQL database
- C++ and Java as service layer
- PHP, HTML5 and Javascript as frontend layer

Ultimately, the two systems, Aggregator Manager and inEIS, and will gradually be merged into a common FMAN.

7.5 Flex-Offer Market Design Considerations

In WP2, we plan to develop FMAR by combining the functionalities of two existing TOTALFLEX and KIBERnet software systems:

Market Manager (AAU) this is one of the system from the TOTALFLEX technology stack. It is a prototype/reference implementation of the local DSO-oriented flexibility market developed in TOTALFLEX. It receives flexibility market bids in the form of Flex-Offers with associated price information, and clears the market in the socio-economical way by solving a mixed integer programming (MIP) problem. Winning bids are distributed back to the bid issuers – the users of the Aggregator Manager. The market clearing functionality used by Market Manager will be used as a starting point for providing the *many-to-many trading* option in GOFLEX.

KIBERnet FLEX (INEA) system currently integrates the functionality of the one-sided pool trading for demand and supply balancing applications. This functionality will be generalized to fit the needs of congestion management in GOFLEX. Further, new action-based tech-

niques will be developed to fit the needs of the two-sided pool and many-to-many trading applications in GOFLEX.

Three alternative variants of FMAR are envisioned in WP2:

- **FMAR as library** In this variant, all FMAR functionality with accompanying auctioning algorithms is packaged into a library. This variant is to be used in cases when the bidder (e.g., aggregator) and the market operator is the same entity. This variant is also to be used in the prototype version of ATP.
- **FMAR as a service** In this variant, FMAR runs as a stand-alone (web) service. Only individual ATP sub-systems (FOA and FMAN) have an access to it. It is designed for cases, when FMAR is owned and the market run by the third party market operator.
- **FMAR as a stand-alone trading platform** In his variant, FMAR be hosted by an independent third party entity. Different groups of flexibility providers and consumers will be able to submit their localized flexibility requests and offerings in the standardized Flex-Offer form. The system will automatically initiate trading processes and handle validation and settlement of offerings. This variant is the ultimate form of FMAR.

7.6 Consideration for KPIs

To address the requirement F2.5, a subset of KPIs defined in the GOFLEX proposal will be considered in the context of ATP. These KPIs, together with our proposed quantification measures, are listed below.

- **KPI 2.1.1.1.a "The EU power network will be capable of integrating large share of renewables exceeding 50% by 2030, in particular variable energy sources, in a stable and secure way"**

  ATP (FMAN or FMAR) should be able to report *electricity load adaptability level* as a ratio between electricity demand variation (∆MWh/h) and peak demand (MWh/h). This measure should be reported (and visualized) for every operational hour and computes based on *time* and *amount* flexibility information in the aggregated Flex-Offers.

- **KPI 2.1.1.1.c "2.1.1.1.c "Demonstrated solutions have the potential to be scaled (if needed) and replicated"**

  ATP (FMAN or FMAR) should be able make projections of the available flexibility (∆MWh/h) and trading deals (EUR/h) in the (hypothetical) cases of having a number of active Prosumers, Aggregators, DSOs is 10, 100, 1000 times larger.

- **KPI 2.1.1.2.a "Competitive demand response schemes for the benefit of the grid and the consumers"**

  ATP (FOA, FMAN, and FMAR) should report factual (EUR) and projected (EUR/MW/year) revenues and/or savings associated to using ATP for DR trading.

- **KPI 2.1.1.2.b "Validated contributions for improved stability and flexibility in the distribution grid, avoid congestion; enabling near real-time pan European energy balancing market"**

  ATP (FMAN) should report, on the hourly basis, the level of adaptation (MWh) introduced by ATP by contrasting baseline and optimized consumption/production Flex-Offer schedules.

No, all data needed to compute these KPI measures will be available in ATP though prosumer/DSO provided Flex-Offers. These KPI measures will be available through the provided APIs and GUIs of the APT sub-systems (FOA, FMAN, and FMAR).

8 Implementation Plan

In this chapter, we explain how exactly ATP will be implemented within and after the duration of GOFLEX. First, we present the long term development plan spanning after the duration of GOFLEX, and later explain immediate actions to be pursued in GOFLEX.

8.1 Implementation Roadmap (long-term)

As explained before, WP2 will combine individual TOTALFLEX and KIBERnet solutions into different ATP sub-systems – FOA, FMAN, and FMAR. However, as shown in Figure 24, we foresee that the level of system integration will vary for different ATP sub-systems at the different stages of ATP development.

Within GOFLEX until the month 30 (M30), the most of the integration efforts will target the prosumer-level (and DSO-level) system FOA. At the FOA level, both TOTALFLEX and KIBETnet solutions will be fully integrated, resulting in the common FOA system capable of running in the stand-alone mode or in the combination with one of the xEMSes used in GOFLEX. Within the duration of GOFLEX, the KIBERnet solution will be used as a base for FMAN and FMAR. It will also be possible to use the extended TOTALFLEX solution as a specialized instance of FMAN for managing flexibility portfolios in the cases of delegated trading. In this configuration, both the KIBERnet and TOTALFLEX solutions will be extended to provide the required

ATP functionality. The two systems will share the common Flex-Offer handling functionality – used for Flex-Offer data exchange, (dis-)aggregation, optimization, etc. This common functionality will be packaged and shared by both systems.

Within GOFLEX until the month 36 (M36), as seen in Figure 24, we will also draw essential guidelines and specify how exactly FMAN and FMAR have to be combined into the two fully integrated ATP sub-systems. For this, the experiences gained in the demonstration cases will be used. Additionally, the support for additional popular xEMSes and their integration with FOA will be considered.

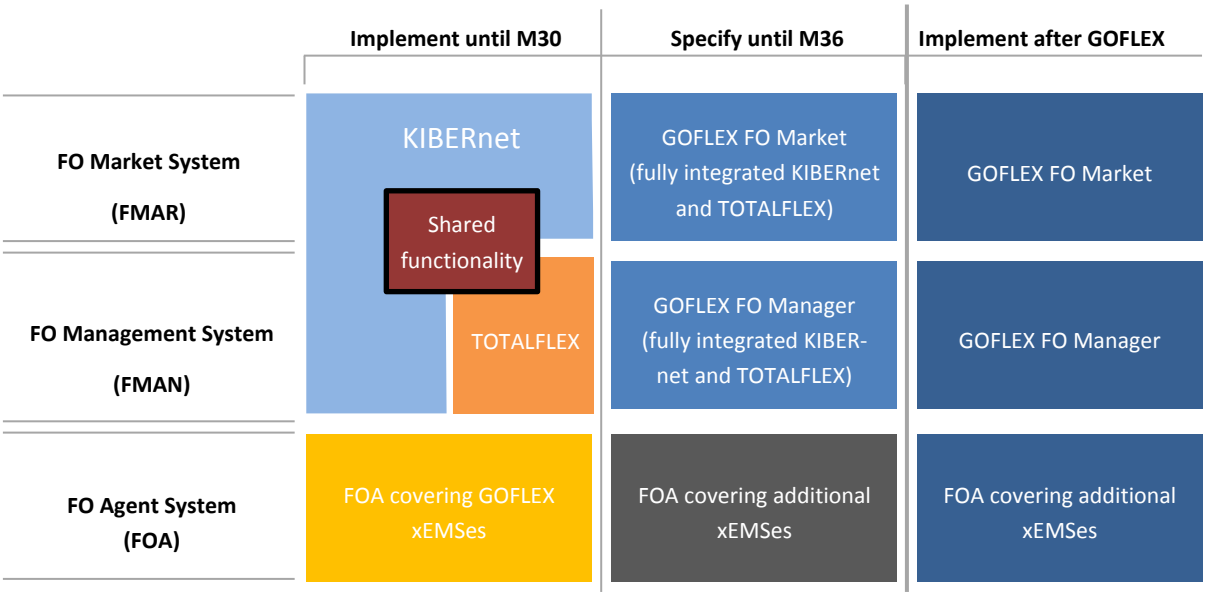

Figure 24: The roadmap for the full TOTALFLEX and KIBERnet integration

After the duration of GOFLEX, we will start the remaining TOTALFLEX and KIBERnet integration efforts aiming for fully integrated FMAN and FMAR solutions. As the same time, extensions to FOA will be developed for handling other popular commensal xEMSes (not considered in GOFLEX). In parallel, the commercialization of ATP will start.

8.2 WP2 implementation plan (short-term)

The implementation efforts within GOFLEX until M30 (the left of Figure 24) will follow the three major iterations, leading to three ATP instances: ATP prototype, ATP full version, ATP final version. First, a working prototype of the ATP will be developed and tested in a local (laboratory) setup. Thereafter, the full version of the ATP will be developed, assembled, and iteratively deployed at the demonstration sites (onsite and cloud). Based on the trial experiences, the final version of the ATP will ultimately be provided.

8.2.1 Prototype

The initial prototype of the ATP will be developed adapting the KIBERnet and TOTALFLEX solutions to the requirement of the GOFLEX. The prototype will be implemented in an internal environment that supports generation and trading of flexibility for diverse load types in different trading modes. The prototype implementation will replicate the demo cases in a simulated environment and it will be used to testing the solutions for each prosumer type. Further, the ATP prototype will be carefully validated to ensures proper communication and data exchange between various intra-ATP sub-systems (FOA, FMAN, FMAR, discussed in Section 3). The integration of FOA with the existing EMSes will be tested using the KIBERnet solution. However, in the case of no xEMS, FOA with the built-in EMS is implemented and tested for interaction with a device controller.

8.2.2 Full Version

The ATP sub-systems and functionalities developed in the prototype are replicated to each demo site with actual prosumers having FOA solutions installed. The FOA system will interact with actual xEMS, device controllers, users, smart meters, and external web services through respective interfaces (APIs). The FOA will produce Flex-Offers for the connected loads and exchange it with the FMAN. It will also ensure the execution of retrieved Flex-Offers schedules. The FMAN will be enhanced to support One-Sided Pool, Two-Sided pool, and Many-to-Many trading options for both direct and delegated trading mode. The communication with the external GOFLEX components will be extensively tested and updated (if required) to ensure the robustness of the ATP system. The modules for the KPI reporting will be implemented based on data from the actual grid system. The system will then be integrated with the DOM and Service platform, in order to send and receive bid information, meter data, and grid status data. The FOA will be enhanced to support smart meter data collection and transportation to/from the service platform.

8.2.3 Final Version

The final version of ATP will be fully integrated with other GOFLEX solutions, and the core ATP sub-systems will be improved and finalized based on previous simulation/laboratory tests and their performance in demonstration sites. The final version of ATP will include the functionality of the *implicit capacity trading* and provide the documentation required for configuring and using ATP, and guidelines for using its interfaces. Final performance analysis against specified metrics and the KPI measures are also reported. FOA will become mature, still highly extensible and flexible to be integrated with the variety of different xEMSes and/or physical devices, and FMAN/FMAR will supports various trading modes, sufficient to cover the cases in GOFLEX.

8.3 The tentative use of ATP in the demonstration cases

Figure 25 generalizes and illustrates the overall ATP operating environment in GOFLEX. Each individual demo cases will include a subset of actors from this environment, as discussed in the Sections 7 and 8 of D6.1. Further, each demo case will have different types of prosumers. Depending on the prosumer type, ATP will support delegated, direct, or both trading modes. Below, the detailed specifications of ATP configuration are given for each GOFLEX demonstration case.

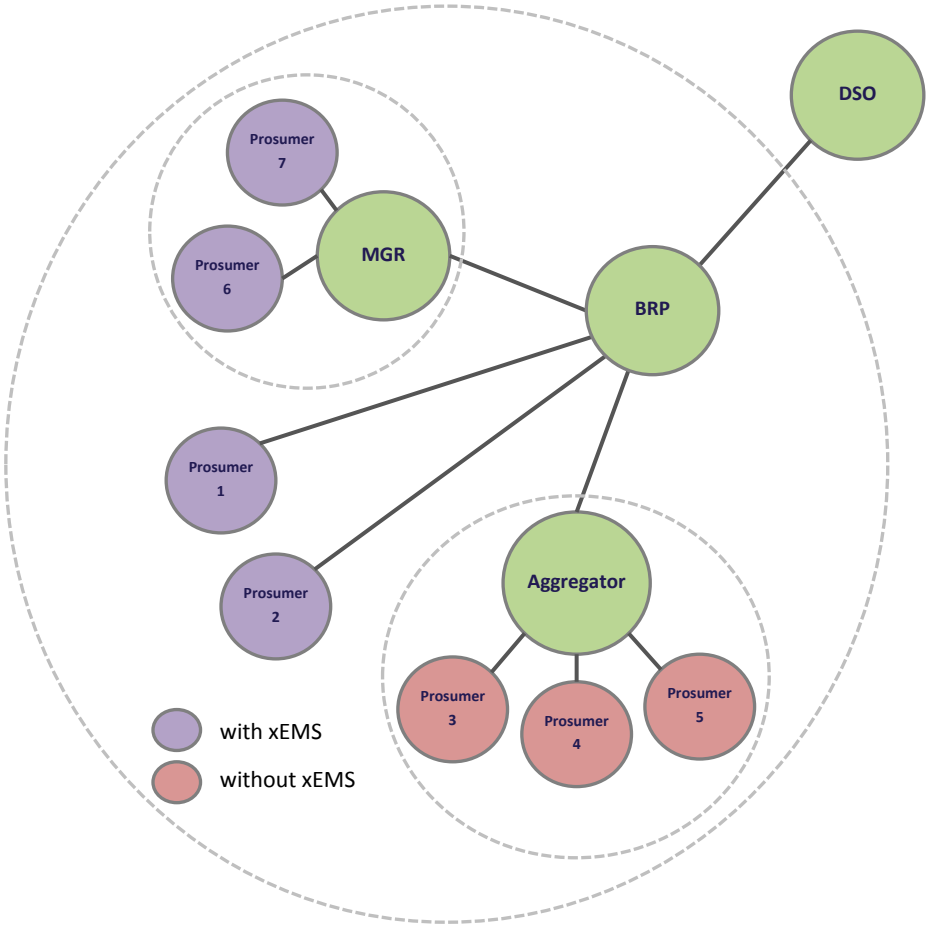

Figure 25: ATP operating environment

8.3.1 Full version DC 1 (FOSS)

120 prosumers are to be installed with FOA as HW box + smart plugs, and the ATP trades Flex-Offers for these prosumers in delegated mode. The rest 63 prosumers are to be installed with FOA as hardware and ATP trades Flex-Offers for these prosumers in direct mode (detailed in D6.1 Section 8.3).

FOA has to generate current-time to 24 hours ahead Flex-Offers at 15 minutes to hourly granularity.

8.3.2 Full version DC 2 (ESR)

230 prosumers are to be installed with FOA as HW box + smart plugs, and the ATP trades Flex-Offers for these prosumers in delegated mode. 40 of the prosumers are to be installed with FOA as hardware and ATP trades Flex-Offers for these prosumers in direct mode (detailed in D6.1 Section 8.2).

FOA has to generate current-time to 24 hours ahead Flex-Offers at 15 minutes to hourly granularity.

8.3.3 Full version DSC 3 (SWW)

10 prosumers are to be installed with FOA as HW box + smart plugs, and the ATP trades Flex-Offers for these prosumers in the delegated mode. Rest 37 prosumers are to be installed with FOA as hardware, and ATP trades Flex-Offers for these prosumers in the direct mode (detailed in D6.1 Section 8.1).

FOA has to generate current-time to 24 hours ahead Flex-Offers at 15 minutes to hourly granularity.

9 Conclusion

WP2 is one of the major solution work packages in GOFLEX. It is responsible for providing a comprehensive automatic trading platform (ATP) for a variety of flexibility providers (e.g., prosumers), flexibility consumers (e.g., BRP, DSO), and trading modes. Thus, the main focus of the WP2 is to design an ATP capable of supporting diverse prosumers types, including the prosumers from the three GOFLEX test/demonstration sites (described in WP7-WP9), and provide a platform for flexibility trading in different trading cases which are instances of so-called *direct* and *delegated* trading. In this regards, this report provided the specifications of the core functional and non-functional ATP requirements, exchanged data, and its tentative use in different GOFLEX demonstrations, as well as initial design considerations and an implementation plan.

10 References

GOFLEX [2016]. http://goflex-community.eu/,

KIBERnet [2017], http://www.kibernet.hu/home/,

TOTALFLEX [2012]. http://www.totalflex.dk/,

Laurynas Šikšnys et al. [2016]. Dependency-based FlexOffers: scalable management of flexible loads with dependencies. *In Proceedings of the Seventh International Conference on Future Energy Systems (e-Energy '16). ACM, 13 pages.*

S. F. Barot et al. [2015]. Load aggregation for demand response using polytrophic models and the Minkowski sum. *In Proc. of CIGRÉ Canada.*

Universal Smart Energy Framework (USEF) [2017]. https://www.usef.energy

MIRABEL [2010] http://cordis.europa.eu/project/rcn/93821_en.html