Intelligent Assistants for Flexibility Management
(Grant Agreement No. 957670)

D6.6 Revised Application-specific iFLEX Assistant prototypes

Date: 2022-12-16

Version 1.0

Published by the iFLEX Consortium

Dissemination Level: PU - Public

Co-funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 957670

Document control page

Document file: D6.6 Revised Application-specific iFLEX Assistant prototypes.docx
Document version: 1.0
Document owner: **ECE**

Work package: WP6 System integration and service packaging
Deliverable type: DEM - Demonstrator, pilot, prototype

Document status: ☒ Approved by the document owner for internal review
☒ Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Andraž Javernik (ECE)	07.06.2022	ToC
0.1	Kristjan Koželj (ELE)	03.10.2022	AMS Slovenian pilot
0.2	Janne Takalo-Mattila, Jussi Kiljander (VTT)	08.11.2022	Finnish pilot deployment updated
0.3	Kokos Isidoros (ICOM), Charitos Nikolaos (ICOM)	08.11.2022	3.2 Greek iFLEX Assistant prototype Aggregation & market interface (Slovenian pilot), User interface app (Slovenian pilot)
0.3	Andraž Javernik (ECE)	22.11.2022	HEMS in Slovenian Pilot
0.3	Arso Savanović (SCOM)	29.11.2022	Updated DMRS on section 3.3.1, 3.3.1.2.9
0.3	Dušan Gabrijelčič (JSI)	01.12.2022	Updated section 3.3 Slovenian iFA
0.4	Andraž Javernik (ECE)	05.12.2022	Preparation of a document (D6.6) for internal review
1.0	Andraž Javernik (ECE)	16.12.2022	Drafting report regarding the reviewers' comments and suggestions

Internal review history:

Reviewed by	Date	Summary of comments
Nikolas Ntavarinos (AUEB)	2022-12-13	Accepted with minor comments and modifications
Louise Birch Riley (IN-JET)	2022-12-13	Approved with suggestion for general improvements

Index:

1 Executive summary

The results from task 6.6 Revised Application-specific iFLEX Assistant prototypes are used to create a baseline for the second piloting phase of software framework for modelling and optimization of consumer flexibility management. This will lead in to a more efficient system which can manage demand and production flexibility for consumers and prosumers and is based on the data that is provided by the Home Energy Management System (HEMS), Building Energy Management System (BEMS), Distributor System Operator (DSO) and energy supplier. In the sector of residential and small commercial customers, the RES and consuming appliances can also be controlled behind the meter with HEMS or BEMS. Most of the current solutions installed on the market are using manual or local control of appliances. Task 6.6. is responsible for testing those components, analyse their functionalities and limitations and describe interfaces between hardware and software functional components. The task provides analysis which is used as baseline for an upgraded design to integrate the connection between the HEMS, BEMS into the common iFLEX software framework.

In the second piloting phase, the focus of pilots is on testing and deploying the existing equipment that will be connected to the iFLEX framework. The purpose is to determine limits and obstacles and to experiment and evaluate the technical functionality of the iFLEX framework, mainly related to checking the possibilities of installation devices in the end-user environment, data collection, ability of services to access data collected by the smart meter, HEMS, BEMS, devices, forecasts, flexibility control and user interfaces.

The second piloting phase version of the iFLEX framework consists of different functional components; Resident interface, Facility Manager Interface, Resource Abstraction Interface (RAI), Automated Flexibility Management (AFM), Digital Twin Repository (DTR), Weather Service Interface (WSI), Aggregator & Market Interface Module, Demand Response Management Systems (DRMS), Real-Time Energy Metering & Actuation Platform (REMAP), Enrolment Module, Home Energy Management System (HEMS), User Interface (UI in house and building level), Smart Metering and Weather Data ingest Module.

The development of the common iFLEX framework is a continuous and iterative process executed with collaboration of pilot users in all phases. In this second piloting phase initial user engagement activities were carried out in workshop activities and their contribution is considered and implemented in mock-ups and user interface design. In Task 6.6 the updated End User Interface (EUI) screenshots / mock-ups are created, which were designed in collaboration with some users who participated in 1.st and 2.nd piloting phases (pre-pilot phase and second piloting phase).

2 Introduction

2.1 Purpose, context and scope

This delivery paper focuses on the Finnish, Greek and Slovenian second piloting phase which covers experimental testing, deploying and implementation of higher-level hardware and software functional components and assessing the technical functionality of the iFLEX framework primarily in relation to data collection, flexibility control, predictions and user web browser (testing mode) or mobile app base interface (for user data visualization and HEMS, BEMS system manipulation). The delivered paper also describes experimental implementation methods, that are capable of monitoring and controlling HEMS and BEMS deployed systems related to weather forecasts, emission factor of electricity production, power consumption, electricity price, user comfort, sensor and smart power meter measurement, data visualization, etc.

2.2 Content and structure

This document is structured in six sections and the main focus is on section 3 as follows:

- Section 3, iFLEX framework prototypes for second piloting phase describes how basic revised functional blocks relate to each other, how they communicate (internally and externally) and what was the main purpose of individual function set. Basic HEMS functionalities are given with sets of possible external devices that are capable of communicating with HEMS and BEMS through different physical communication protocol. High-level communication protocols like REST API and MQTT are tested and collected data was then visualized via the second phase python developed Graphical User Interface (GUI). Furthermore, the revised EUI was presented, and first functional screenshots / mock-ups are shown and presented on workshop with presence of potential pilot users.

2.3 Key changes in this deliverable according to the previous one

The overall scope of changes in the second-piloting phase according to first-piloting phase can be summarized as follows:

Finnish pilot:
In the second-pilot phase, the Finnish pilot oriented on the following iFLEX Assistant components:

- Aggregation and Market Interface was change from demo and testing interface to Enerim's Aggregation Platform,
- End user feedback,
- End user logs,
- End user DevOps and MLOps,
- Building Management System update with new HVAC sensors data,
- Containerization technology is deployed for better development during iFLEX project.

Greek pilot:
The description of the application-specific iFLEX Assistant prototype for the Greek pilot has been updated according to the specifications of second-pilot phase. More specifically, the deployment diagram has been modified, as present the iFLEX components and external systems, which will be integrated into the pilot by the end of this phase, as well as the communications protocols between them. Furthermore, the functionalities of the following components or systems are elaborated:

- User Interface component,
- Aggregator and Market Interface component,
- ICOM's Demand Response Management System,
- Heron's Real-time Energy Metering & Actuation Platform.

Slovenian pilot:
The second piloting phase deliverable has been updated with information on designed, developed, integrated and deployed components forming together an application specific iFLEX assistant prototype used in the

Slovenian pilot. In particular, the information on following components have been updated with the advances in the last piloting period:

- HEMS deployment and installation, together with needed details for application integration,
- Illustration of RAI deployment and application integration,
- Digital Twin deployment and functionality,
- User Interface's instantiation for the Slovenian pilot and supported features,
- Aggregator and Market Interface's functionalities.

3 IFLEX Assistant prototype

3.1 Finnish iFLEX Assistant prototype

3.1.1 Overview

The Finnish pilot consists of only a single iFLEX Assistant (iFA) that is responsible for a whole apartment building. Please refer to the *HLUC-3: Manage flexibility at building community level*, documented in D2.1 [1] for further details on the use case. This use case covers an important consumer sector in Finland, because roughly 40% of people[1] live in an apartment building and it is typical that the majority of the energy costs in an apartment building (e.g., heating, warm water, sauna) are shared by the housing cooperative.

In this so called second-pilot phase, the focus in the Finnish pilot is on experimenting and evaluating the technical functionality of the iFLEX Assistant, mainly related to data collection, forecasts, flexibility control and user interfaces.

The iFA is deployed into an apartment building with 90 apartments. The apartment building has a Building Management System (BMS) that monitors and controls the building's heating ventilation and air conditioning (HVAC), as well as, lighting, electric sauna and elevators. Figure 1 presents the stakeholders and external systems of the Finnish pilot iFA.

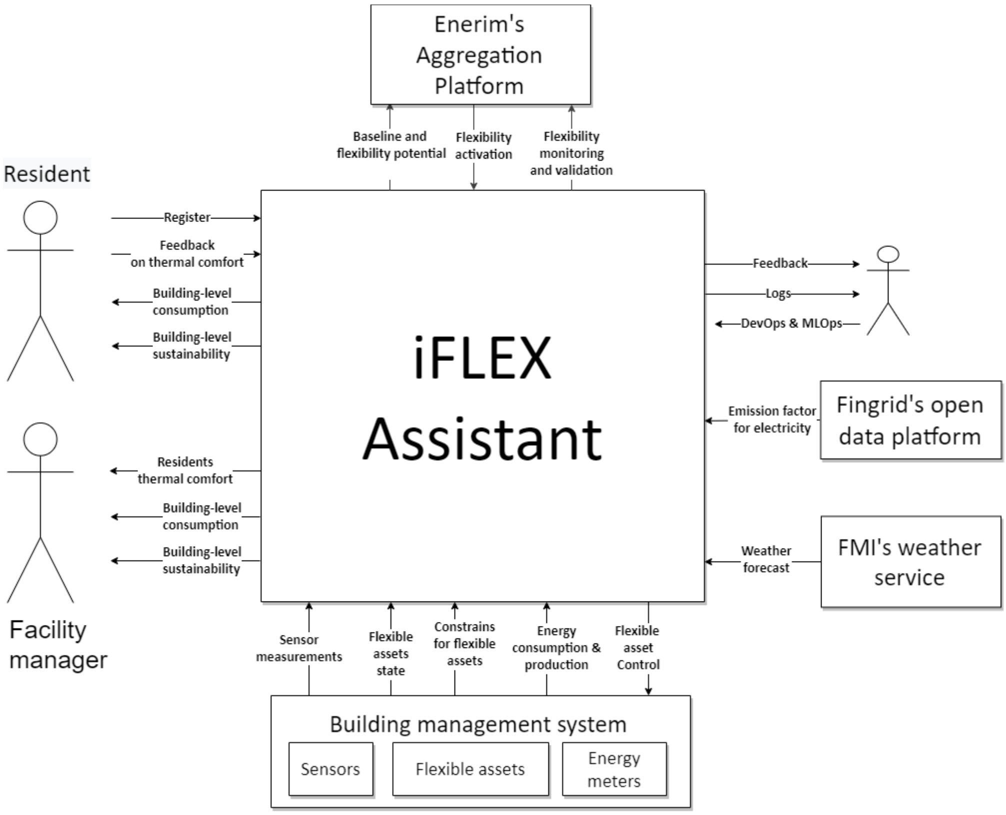

Figure 1: iFLEX Assistant of the Finnish pilot with relevant stakeholders and external systems

There are two types of end-users: Residents and Facility manager. Please refer to iFLEX deliverables D2.1 [1] and D2.3 [2] for further details on the services provided to these end-users.

The iFA interacts with several external systems. The BMS provides iFA with means to monitor and control building assets. The pilot building had an old BMS system without any external interfaces. For this reason, a JACE-8000 gateway was deployed into the building. The iFA accesses the following measurements via the JACE-8000:

- Building level electricity consumption, power and frequency at one-minute resolution.
- District heating energy consumption at one-minute resolution.

[1] https://www.stat.fi/til/asas/2017/asas_2017_2018-05-17_tie_001_en.html

- Ventilation units' return air temperature and optionally return air relative humidity and CO_2.
- Average indoor air temperature, relative humidity, and CO_2 of the building.
- Various control-related parameters, including set-point and measured values for water temperatures across the system (e.g., space heating, domestic hot water, exhaust air heat pump supply), as well as, status information (on water pumps', fans', control valves and heat pump compressor status).

Fingrid Open Data platform provides the iFA with estimates of CO2 emissions for electricity consumptions. Finnish Meteorological Institute (FMI) provides the weather forecasts utilized by iFA in forecasting the energy consumption and flexibility. In the second phase, there is also no aggregation of buildings, and an external interface is set-up demonstrating and testing flexibility management functionalities of the iFA.

3.1.2 Deployment view

The second-phase version of the iFLEX Assistant consists of following functional components: Resident Interface, Facility Manager Interface, Resource Abstraction Interface (RAI), Automated Flexibility Management (AFM), Digital Twin Repository (DTR), and Weather Service Interface (WSI). These logical components are implemented with concrete software artifacts as follows:

- **Resident Interface:** The resident interface consists of Resident frontend, Resident backend and User Database (DB) artifacts. The Resident frontend is implemented as web browser application, usable with mobile phones and computers. The Resident backend is responsible for interfacing with the RAI component to fetch relevant data for the Resident interface. It is implemented with Node.js. The User DB is implemented on top of MongoDB. These back-end software artifacts are deployed on a Resident server.

- **Facility Manager Interface:** The facility manager interface consists of a variety of dashboards to visualize buildings energy parameters. It is implemented on top of Grafana framework[2]. The Grafana back-end is hosted on the same server as the software artifacts of the RAI module.

- **Resource Abstraction Interface:** The second pilot phase implementation of the RAI components consists of four software artifacts: oBIX database, Building Energy Management System (BEMS) interface, Fingrid integrate, and Client interface. The oBIX database stores all the data collected from BEMS, Fingrid, Finnish Meteorological Institute (FMI). The BEMS interface is implemented with Java and provides mechanism for collecting and controlling building resources. It interfaces with a JACE gateway of the BEMS system with standard oBIX protocol. The Fingrid interface component is implemented with Java and provides means to collect CO2 emission data from Finnish Transmission System Operator (TSO), Fingrid. The Client interface provides other functional components of the iFLEX Assistant access to the data stored in oBIX database. The RAI components are deployed in a single server, called RAI server.

- **Digital Twin Repository:** In the iFLEX Framework the digital twin repository consists of a variety of Digital Twins (DTs) for the consumer, including people and buildings they live in. For the Finnish iFLEX Assistant, a digital twin of an apparent building is used from the repository developed for phase one. The Building twin is implemented with Python programming language on top of scipy[3] and tensorflow[4]. It consists of two models for estimating the building's baseline load profiles both for electricity and district heating, as well as, flexibility model, including indoor temperature model, heat pump model and space heating model. The Building twin is implemented as a single software artefact (Python class) that is interfaced by the AFM component.

- **Automated Flexibility Management:** The AFM component is realized as a single process that imports also the Building's twin provided by the DTR component. The AFM is implemented with Python programming language and it provides an MQTT interface for flexibility management. In the second-phase the AFM interface will be demonstrated and tested via specific interface. In the second phase, the AFM interface will be integrated to the Aggregator and market interface component.

- **Weather Service Interface:** The WSI component deployed for the Finnish pilot interfaces with the Finnish Meteorological Institute (FMI) weather forecast service and stores the forecast into the oBIX database. The FMI interface is implemented as a single process that is deployed into the same server as the RAI components.

[2] https://grafana.com/
[3] https://www.scipy.org/
[4] https://www.tensorflow.org/

- **Aggregation and Market Interface:** The Aggregation and Market Interface (A&M Interface) provides an MQTT interface for the Enerim's Aggregation Platform to interface with the iFLEX Assistant (i.e., it provides messages on the baseline and flexibility forecasts and enables activation of the flexibility provided by the consumer). Additionally, the A&M Interface provides the AFM module with means to fetch energy price information from the Enerim's Aggregation Platform.

Figure 2 illustrates the deployment view of the iFLEX Assistant for the Finnish pilot.

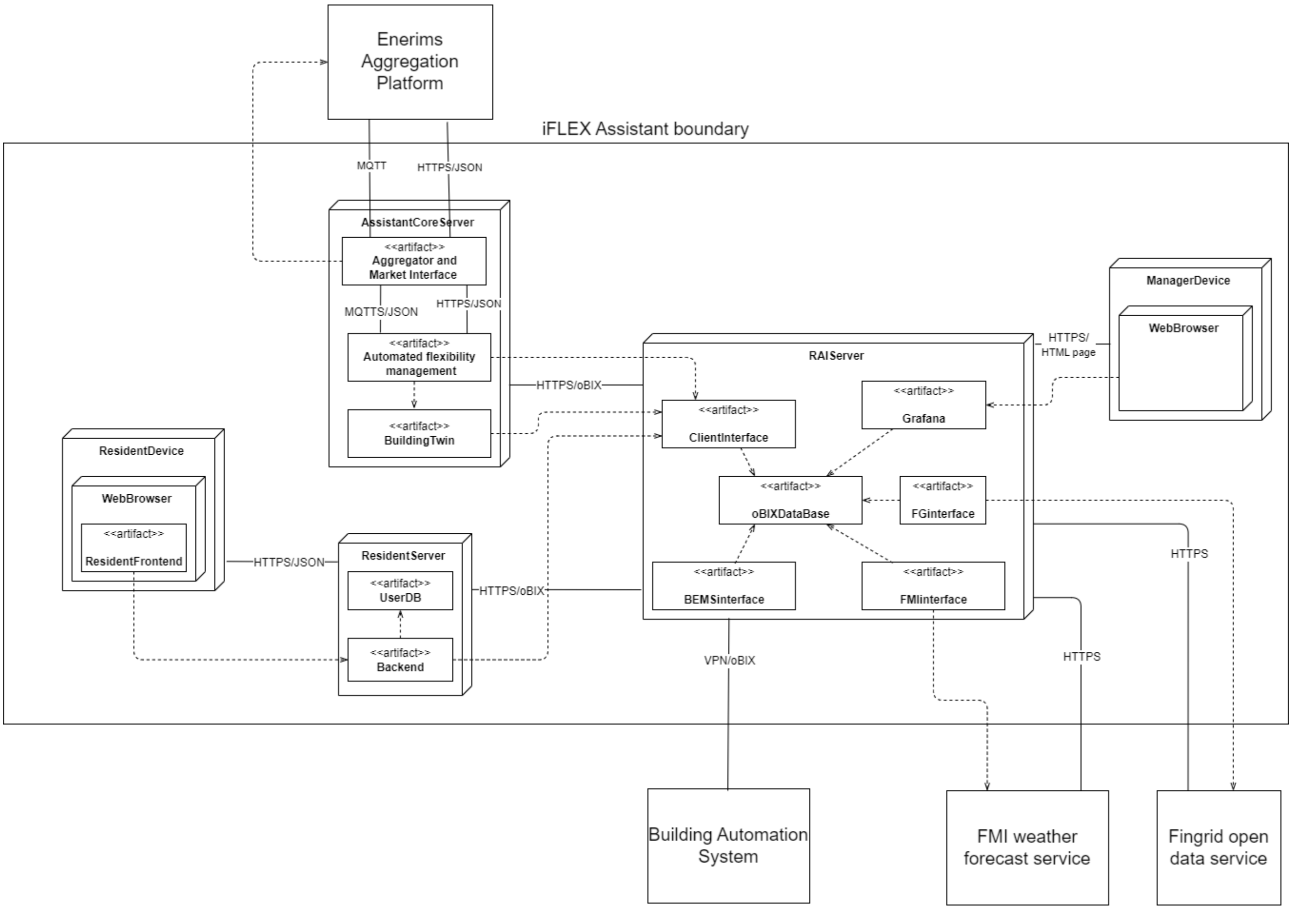

Figure 2: Deployment view of the iFLEX Assistant prototype for the Finnish pre-pilot

3.1.3 iFLEX assistant in the containerized environment

The core parts of the iFA are deployed in cloud environment as a containerized application. These parts of iFA include AFM, required models from digital twin repository and interfaces. All of these components have been developed using Python version 3.9.5. Conda package and environment management system have been used

in the development phase in order to share identical environment among the developers. Following packages are required by the iFA and therefore installed to Conda environment:

```
- pip=21.1.2
- python=3.9.5
- pandas=1.3.4
- paho-mqtt=1.5.1
- requests=2.25.1
- joblib=1.0.1
- matplotlib=3.3.4
- scikit-learn=0.24.2
- holidays
- PyYAML=6.0
- tensorflow-cpu==2.6.0
```

Figure 3: Required packages in Conda environment

Using containerization technology, iFA can easily be deployed into the cloud platform and this technology enables creating new iFA instances with minimal human effort. In iFLEX, Docker and Docker-compose are used as a containerization technology.

The following workflow demonstrates, how iFA instance can be created for the pilot. Docker builds container images using dockerfile as a source of instructions and Figure 4shows an instructions to create an iFA instance.

```
# syntax=docker/dockerfile:1
FROM continuumio/miniconda3
WORKDIR /instances/hoas/
COPY environment_docker.yml .
COPY /instances /instances

COPY /bema /bema
ARG PIP_NO_CACHE_DIR=1
RUN conda env create -f environment_docker.yml nomkl && conda clean -afy
CMD ["/opt/conda/envs/building_ema/bin/python", "afm_main.py", "--conf-file",
"afm_config_deployment.yml"]

# Build in terminal
# docker build --tag building_ema -f Dockerfile --no-cache .
```

Figure 4: Dockerfile instructions for iFA

for iFlex pilot. First, it copies the environment file presented in Figure 3 to image and copies also required source codes. With dockerfile instruction “*conda env create -f environment_docker.yml nomkl && conda clean -afy*” conda environment is created into Docker image. Docker image can be created in Linux/Mac terminal with command “*docker build --tag building_ema -f Dockerfile --no-cache .*”

When Docker images are created, containers need to be created based on the images. Docker Compose is used to create and manage multi-container applications. It uses docker-compose.yml configuration file to define how applications are created based on Docker images and how applications interact with each other. Figure 5 describes how an AFM component is created based on Docker image created with dockerfile shown in Figure 4. Docker-compose file also includes a container for visualizing and activating flexibilities and a container that is responsible for managing the forecast history. Using docker-compose, new instances can be easily created just by adding new service to docker-compose file and modifying the configuration files.

```yaml
version: "3"
services:
  building_ema_hoas:
    container_name: building_ema_instance_hoas
    image: building_ema
    working_dir: /instances/hoas/app1
    command: /opt/conda/envs/building_ema/bin/python afm_main.py --conf-
file config/afm_config_deployment.yml
    volumes:
    - ./building_ema/instances/hoas/config:/instances/hoas/app1/config
    - ./building_ema/instances/hoas/tmp:/instances/hoas/app1/tmp
    - ./bema/rai/config:/bema/rai/config
    depends_on:
      - "mqtt_visualizer_hoas"
    restart: always
  mqtt_visualizer_hoas:
    container_name: mqtt_visualizer_instance_hoas
    image: mqtt_visualizer
    command: /opt/conda/envs/building_ema/bin/bokeh serve --allow-
websocket-origin=* mqtt_visualizer.py --args --conf-file
config/visualizer.yml
    volumes:
    - ./visualizer/config:/scripts/config
    - ./visualizer/history:/scripts/history
    - ./visualizer/simulation_hist:/scripts/simulation_hist
    restart: always
    ports:
    - 5006:5006
  history_saver:
    container_name: history_saver_instance
    image: mqtt_visualizer
    command: /opt/conda/envs/building_ema/bin/python history_saver.py --
conf-file config/visualizer.yml
    volumes:
      - ./visualizer/config:/scripts/config
      - ./visualizer/history:/scripts/history
      - ./visualizer/simulation_hist:/scripts/simulation_hist
    restart: always
```

Figure 5: Example docker-compose description

3.2 Greek iFLEX Assistant prototype

3.2.1 Deployment View

The version of the iFLEX Assistant (iFA) for second pilot phase in the Greek pilot consists of the following iFA components: User Interface (UI), Aggregator and Market Interface (A&M), Resource Abstraction Interface (RAI), Digital Twin (DT), Automated Flexibility Management (AFM), Security and Privacy. Furthermore, it interacts with the following external components: ICOM's Demand Response Management System (DRMS), HERON's Real-time Energy Metering & Actuation Platform (REMAP), a Push Notification Server, a Weather Service, HERON's tariff API and a CO_2 emissions Service.

The following Figure 6 presents the deployment diagram of the iFA's instance for the Greek pilot, whilst a presentation of the different solutions can be found in the following sections.

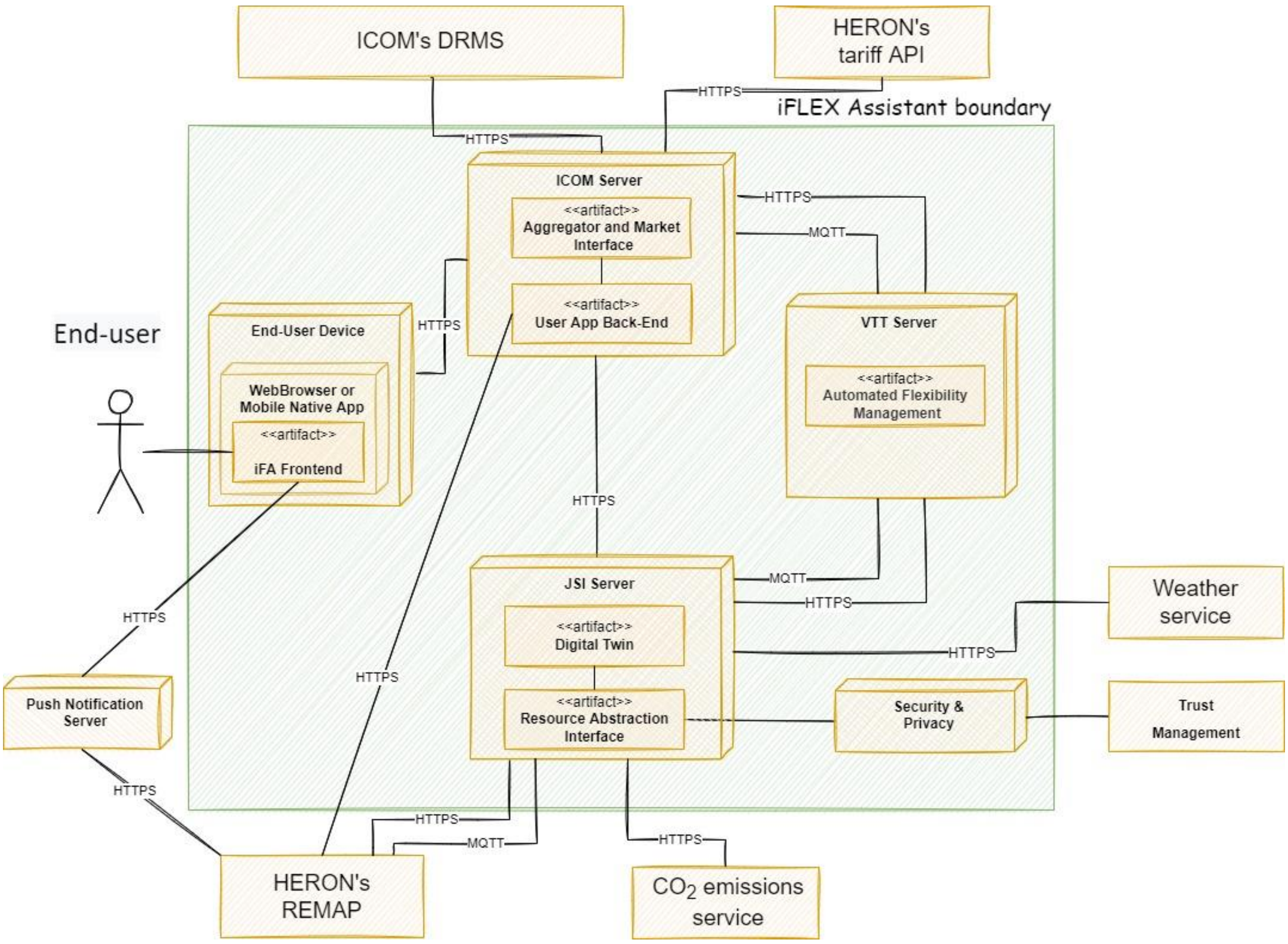

Figure 6: Deployment view of iFA for Phase 2 of the Greek pilot

3.2.2 User Interface App

The user interface module (iFA FrontEnd component in Figure 6) in the Greek pilot will be instantiated from a collaboration between HERON's baseline native application and iFA-specific interfaces developed by ICOM in the project, which will be exposed as a web view (through the User App Back-End in Figure 6). The former shall provide core operations related to User/Home/Device management and energy monitoring (specific to customer offering of HERON), as well as communication of push notifications and security mechanisms, which will be integrated with ICOM's solution. The latter should essentially provide to the pilot users its full set of functionalities by the end of second piloting phase. These functionalities concern user preferences, energy monitoring, Demand Response (DR) services, visualisation of cost-related information, energy advice, as well as sustainability and energy efficiency performance tracking. During this phase, integration between HERON's and ICOM's applications shall also be achieved, as the application should be fully operational to be deployed in the pilot.

More specifically, the following requirements related to iFA's UI component [3] for the Greek pilot should be supported by the end of second piloting phase:

- FN-UI-01 – Operation mode customization
- FN-UI-02 – User-defined time and operational constraints
- FN-UI-03 – End-user feedback
- FN-UI-04 – Optimization policy selection
- FN-UI-05 – Automation level customization
- FN-UI-07 – Supported system interface languages
- FN-UI-08 – Provision of consent for the schedules of dispatchable assets
- FN-UI-09 – DR notification policy
- FN-UI-10 – Insights into sustainability metrics
- FN-UI-11 – Real-time energy data
- FN-UI-12 – Past energy data
- FN-UI-13 – DR reports
- FN-UI-14 – Insights into energy efficiency
- FN-UI-15 – Customized alerts
- FN-UI-16 – Energy advising service
- FN-UI-17 – Inspection of energy tariffs
- FN-UI-21 – DR event notification
- FN-UI-22 – Presentation of DR event history
- FN-UI-26 – View asset schedules when offline
- FN-UI-27 – Actual schedules of assets
- FN-UI-28 – Estimation of energy costs
- FN-UI-29 – Savings thanks to iFA's operation
- FN-UI-01 – Integration of the iFA's UI for the Greek pilot into HERON's app
- FN-UI-02 – Easy on-boarding in iFA's usage

Thus, in the last phase of the project the UI-related work will be mainly focused on potential adjustments based on users' feedback.

3.2.3 Aggregator & Market Interface Module

The A&M Interface module will exchange signals with ICOM's Demand Response Management System (DRMS) based on OpenADR2.0 protocol [4]. The DR-related sub-module is based on the client side (i.e., Virtual End Node – VEN) of OpenLEADR[5], an open-source implementation in Python[6], compliant with OpenADR. When needed, extensions to the OpenADR protocol will be employed, so that the DR-related requirements of the iFLEX project can be fully met. The A&M Interface component is also responsible for retrieving the electricity tariffs from a dedicated API exposed by HERON. It will also transform external DR- and price-related signals to the iFLEX data model, so that they can be communicated to the other components of the iFA. During this phase, functionalities related to both DR and energy tariffs should be supported and tested.

More specifically, the features below should be supported by the A&M Interface Module [5] of the Greek pilot by the end of second piloting phase:

- FN-AM-04 – Information on participation in explicit DR actions
- FN-AM-05 – Communication of flexibility potential
- FN-AM-06 – Access to DR reports to end users
- FN-AM-08 – Receiving flexibility signal
- FN-AM-09 – Communication of flexibility signal
- FN-AM-10 – Response to flexibility signal (explicit DR)
- FN-AM-13 – Communication of electricity tariffs from external system
- FN-AM-14 – Access to electricity tariffs to end users

[5] https://github.com/openleadr
[6] https://www.python.org/

- FN-AM-17 – Access of flexibility validation data from the RAI
- FN-AM-18 – Communication of flexibility validation data
- FN-AM-19 – Assessment of end user participation in a DR event
- FN-AM-20 – Communication of participation assessment data

3.2.4 ICOM Demand Response Management System

The solution of ICOM's DRMS, which will be deployed during this phase of the project, should support the following functionalities [5]:

- FN-DR-01 – iFA end-users' flexibility potential
- FN-DR-02 – Communication of flexibility information history to iFA end users
- FN-DR-03 – Sending Flexibility Signal
- FN-DR-04 – Response to Flexibility Signal (explicit DR)
- FN-DR-05 – Information on participation in explicit DR actions
- FN-DR-06 – Access of flexibility validation data from the iFA
- FN-DR-08 – Response to flexibility request
- FN-DR-09 – Flexibility dispatch
- FN-DR-10 – Provide activated flexibility report
- FN-DR-11 – Calculate individual performance and participation rate metrics of DR participants
- FN-DR-12 – Calculate statistics per DR event and DR program

The DRMS employs the OpenADR2.0 protocol to communicate with iFA, while extensions to this protocol will be utilized to meet iFLEX requirements when necessary. The DR communications solution is based on the OpenLEADR implementation, as in the case of iFA's A&M Interface module, but on the server side (i.e., Virtual Top Node – VTN). As regards the interaction with the BRP system, which is operated by OPTIMUS in the context of the Greek pilot, flexibility requests are exposed through a REST API, documented via OpenAPI specification[7].

3.2.5 HERON's REMAP

HERON operates an integrated platform consisting of software and hardware systems for energy monitoring and management, the Real-time Energy Metering & Actuation Platform (REMAP). HERON's solution will provide an interface for iFLEX Assistant in order to access energy data, access user schedule provided through manual operation and provide asset control commands to the on-premises assets and will support the baseline User Interfaces application. The metering infrastructure utilized includes real-time power meters, a back-end system for measurements collection and storage and a remote API to provide access to iFA. Furthermore, following the Revised Greek pilot specifications outlined in Deliverable D7.2 [6], HERON's platform is being updated to accommodate the inclusion of additional smart assets such as various types of smart plugs. A Grafana dashboard (Figure 7) which can be accessed by pilot participants independently of the iFLEX app has been provided.

[7] https://swagger.io/specification/

Figure 7: Smart Plugs Grafana dashboard

3.3 Slovenian iFLEX Assistant prototype

The Slovenian iFLEX Assistant framework prototype in the second piloting phase consists of the following functional components: an Enrolment Module, Home Energy Management System (HEMS), Resource Abstraction Interface (RAI), Smart Metering and Weather Data Ingest Module, Digital Twin, Demand Response Module, Trust Management Module, Security and Privacy Interface and End-User Interface.

The second phase Slovenian pilot focuses, as has been defined in deliverable D7.2 [6], deployment and integration wise at deployment, testing and evaluation of the HEMS in working conditions with up to and including second phase pilot group of users, pilot infrastructure and background components, integration validation of the input data sources with the background system, and improving of the iFLEX Assistant end-user interface design and implementation. The status of this work is reported in Section 3.3.1 on deployment view. In the section, at first, the status of the components and iFLEX Assistant subsystems are briefly abstracted. The components' deployment illustration beyond the reports given so far is then reported in the section subsections.

According to the plan specified in D7.2 [6],the iFLEX Assistant prototype will be used in the second piloting phase with a focus on implementation of primary use cases and pilot deployment. The targeted use cases to be addressed were selected and described in the Revised architectural deliverable D2.4 [7].

3.3.1 Deployment view

At the beginning of the second phase piloting, the iFLEX Assistant framework consists of a number of components:

- **Enrolment module**: the Enrolment module enables user friendly and privacy compliant enrolment of the end users in the pilots. The users get invited in the project by the pilot host through the module. The module is used to enable pilot users to give consent for data collection and processing and to initialise basic data structures needed for backend system operation. More information on the enrolment procedure is available in the deliverable D4.7 [8]. The enrolment module has been deployed at address *uporabnik.iflex-project.eu*. The name 'uporabnik' stands for the user in Slovenian. The reason for such name is because the module is a frontend for all current and future piloting users activities.
- **HEMS**: the HEMS gets installed and connected with the rest of the framework during the enrolment procedure. It is essential component for sensing, data collection and activation of devices and their flexibility in pilot user's households. HEMS functionality and piloting deployment are further described in Section 3.3.3,
- **Resource Abstraction Interface**: the Resource Abstraction Interface (RAI) presents a component that collects the data from HEMS and other external sources, stores the data and provides the data to other iFLEX components, such as the Enrolment Module (EM), Digital Twin (DT) and End User Interface (EUI). The RAI provides interfaces for actuators available at HEMS as well. RAI design and implementation have been reported in the deliverable D4.2 [9]. Second piloting phase RAI has been implemented as an extension of a Prosumer Cloud Service (PCS). The following functionality has been deployed:
  - Services for data collection from the HEMS, second phase implementation are based on MQTT protocol. The data include measurements from sensors, devices configuration information and devices state. The service receives designed and implemented MQTT messages from HEMS devices periodically, on One-minute intervals, and stores it in the PCS storage,
  - Services for data ingest from external services: an interface has been provided for ingest of data from external data sources like smart meter data and weather data. The data is stored in the PCS and is aggregated in other time slices (1h, 2h, 6h, 1d, 1w, etc.),
  - Collects weather data for selected household locations from Open Weather Map (OWM) and stores the data in the PCS storage,
  - Provides backend service for selected data collected in enrolment process, needed for DT operation,

  - Provides backend service for storing DT models data as well as household building parameters. The RAI will be likely used for storing some model states in the next iteration,
  - Exports all the selected data to the rest of components in an as unified way as possible via designed and implemented RAI API. Aggregates of the data are available for many collected data, some will be added in the next iteration,
  - Provides information on tariffs and prices per household. The interface will be moved to Aggregation & Marketing interface in the next iteration,
  - Provides control functionality over assets that can be controlled by the project. The assets include on-off devices, Photo Voltaic (PV) installations, heat pump set-points, etc.
- **External data ingest module**: The ingest module pushes the data from the data provider towards the RAI module. The ingest module has two parts, the first being the pusher and the second the ingest. The deployment of both modules is as follows:
  - The pusher takes care to push the data from the source towards the ingest module. In the case of Slovenian pilot the pusher pushes smart metering data from Elektro Celje (ELE) data sources towards ingest module. The deployment is based on existing software services available at ELE; the same services are used to provide data to other ELE clients, like transmission operator, retailers, etc.
  - The ingest provides FTP server for receiving the pusher data. The ingest reads the inputs periodically, transforms the inputs according to the ingest specification requirements and pushes the data towards the RAI module. The RAI implementation stores the data in the backend storage.
- **Digital Twin**: The digital twin module implements models as discussed in the deliverable D3.2 [10]. The module is currently integrated close to the RAI module, since the data access is crucial to model operation. The deployed module provides the following functionality, some other functionality developed in research notebooks will be integrated in the final phase:
  - The heat and heat consumption demand physical models,
  - Heat pump COP model,
  - Scheduling interface for control from Automated Flexibility Module,
  - REST interfaces to control the DT, access the data calculated by the twin and to load and store models from the RAI storage.
- **Trust Management Module**: The Trust Management Module (TMM) enables cryptographic trust management between system entities like system components, end users, etc. The TMM supports two types of trust management: X.509 Public Key Infrastructure and Self Sovereign Identity (SSI) based trust management. TMM module design and implementation is further reported in D4.8 [11].The X.509 management is fully implemented, SSI one is in an experimental stage. The TMM is in a process of integration with the rest of system components beyond RAI and DT.
- **Security and Privacy Interface**: The Security and Privacy Interface (SPI) provides access control functionality based on security and privacy policies, identities or roles of the entities in the system established through secure communication with iFLEX framework services and enforcement mechanisms provided by the services. The SPI utilizes identities and their credentials as these are created and managed through Trust Management Module. The SPI module is further described in D4.8 [11]. Both REST and MQTT interfaces are supported. The SPI module is already deployed and controls access to the RAI and DT module. In third phase the module will offer access control support to the rest of the components as well.
- **Demand Response**: The updated Demand Response management solution is based on the combination of SCOM's background solution and open-source implementation of OpenADR specification, the OpenLEADR[8]. The updated DRMS prototype is capable of emitting and receiving flexibility events information as well as viewing and managing events via a web interface. While the first phase was focused on development of initial DRMS prototype and its integration with the iFA for the purposes of collecting relevant data from various heterogenous sources, the second phase

[8] See OpenLEADR home page for details: https://www.lfenergy.org/projects/openleadr/

expands this work by exploiting the updated DRMS for carrying out experiments with execution of flexibility events. For this purpose, the user's flexibility potential is provided by iFA to the DRMS, which then dispatches the flexibility events (signals) according to the potential and event schedule. Relevant information on the executed events is fed back to the DRMS, which processes the information and delivers the Flexibility report to the iFA. Some additional information on Demand Response Management System and Marketing & Aggregation module is described in Sections 3.3.5 and 3.3.6 respectively.

- **End-User Interface**: The End-User Interface enables the interaction of the end users with the iFLEX Assistant. In the second phase, it should equip the users with the ability to set their preferences and assist them in managing both energy and flexibility. The UI was designed with the intention to be as simple as possible, while at the same time providing the users with useful insights. The EUI is described in more details in Section 3.3.7.

3.3.2 Advanced Metering System

An Advanced Metering System (AMS), or Advanced Metering Infrastructure (AMI) is understood as a system of smart meters, communication infrastructure and IT systems that enables measurement, remote reading and management of data on the use of electricity for the purposes of billing, monitoring the quality of supply and operation and planning of distribution networks. The emphasis is on providing sufficiently frequent information about usage, billing according to actual consumption, using advanced tariff systems, implementing consumption management measures, and developing and implementing new services.

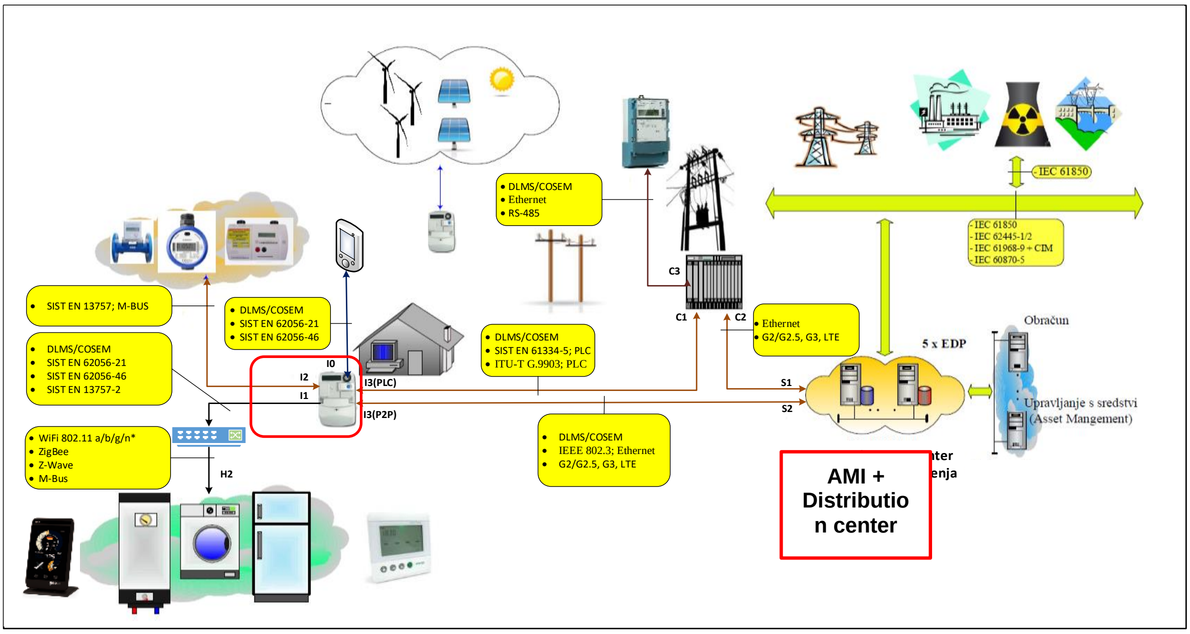

Figure 8: Building an advanced metering system

Interfaces on the smart meter:

- I0 local service interface for DSO,
- I1 interface intended for system users for local access to data and information (dedicated display, Smart Home systems, HEMS, BEMS, etc.),
- I2 interface for local connection with other meters (gas, heat, water...),
- I3 interface between the smart meter and the DSO (PLC OFDM G3 or 4G(LTE)).

Sending data to the I1 interface, as we do for active users in the iFLEX project is for local access to data for the purposes of displaying data, transmitting measurement data to other modules, home devices and systems like HEMS or BEMS.

Basic technical requirements for this interface are:

- one-way communication channel intended exclusively for reading sent data compliant with SIST EN 62056-7-5;
- HAN channel (one-way communication from the meter to house systems and devices such as: Energy House Displays (IHD), Smart House Systems (SM) and Home Energy Management System (HEMS);
- speed ≥ 2400 b/s, basic setting 2400 b/s;
- receiving data requires protection using appropriate cryptographic encryption methods as described in the DLMS/COSEM standard (Green Book, Edition 7 and Edition 8).

The main goal of introducing an advanced metering system is to provide system users with better information and services, and to enable them to better manage their consumption. The configuration of PUSH intervals and objects that are sent to I1 is configurable via the I0 and I3 interfaces. The default configuration for the test samples is specified in Table 1 and Table 2.

Table 1: Data sent from the I1 interface every 5 seconds

Parameter number	Parameter name
0-0:42.0.0	COSEM name
0-0:96.1.2	Device ID3
1-0:32.7.0	voltage L1
1-0:52.7.0	voltage L2 only 3 phase meter
1-0:72.7.0	voltage L3 only 3 phase meter
1-0:31.7.0	current L1
1-0:51.7.0	current L2 only 3 phase meter
1-0:71.7.0	current L3 only 3 phase meter
1-0:1.7.0	A+ current active power
1-0:2.7.0	A- current active power
1-0:3.7.0	R+ current reactive power
1-0:4.7.0	R- current reactive power

Table 2: Data sent from the I1 interface every 1 hour

Parameter number	Parameter name
0-0:42.0.0	COSEM name
0-0:96.1.3	Device ID4
1-0:1.8.1	Energy A+ T1
1-0:1.8.2	Energy A+ T2
1-0:2.8.1	Energy A-T1
1-0:2.8.2	Energy A-T2
1-0:3.8.1	Energy R+ T1
1-0:3.8.2	Energy R+ T2
1-0:4.8.1	Energy R-T1
1-0:4.8.2	Energy R-T2

The I3 communication interface between the meter and the DSO (WAN) is intended for two-way communication with the HES in AMI.

The technical requirements for this interface are:

- PLC communication
  - Narrow-band OFDM G3-PLC (G3-PLC Alliance),
  - G3 PLC chip: frequency range CENELEC A and FCC;

- RF communications (the communication module must simultaneously support 4G and 2G mobile networks)
  - 4G (LTE) mobile network with frequencies of 800, 900 and 1800 MHz
  - 2G mobile network (fallback) 900 and 1800 MHz

3.3.3 HEMS in Slovenian pilot

In the second pilot stage of the iFLEX project, changes took place in the Slovenian part of the pilot, pertaining to the technical part of the HEMS (Home Energy Management System) hardware installed at residential end users. Hardware by another manufacturer was updated. Examples of equipment installation at one of the pilot users will be presented hereinafter.

3.3.3.1 HEMS functionalities and limitations

In the second Slovenian piloting phase the following hardware functional components are used:

- HEMS main control unit:
  - Supports Wi-Fi and Ethernet communication with Resource Abstraction Interface (RAI) and user devices in the Local Area Network (LAN) or via external expansion models connected within the LAN.
  - MODBUS TCP/RTU communication with user devices.
  - Enables power supply via an external 12 VDC connection (power supply unit) or PoE (Power over Ethernet).
- Energy Management System (EMS) to Ethernet gateway:
  - Supports wired communication with heat pumps by the manufacturer BOSCH via an EMS bus.
  - The interface can be powered via EMS bus, Ethernet (PoE) or an external 12 VDC power supply unit.
  - The interface connects to an internal LAN by wired or wireless connection.
  - The interface communicates with the main HEMS control via MQTT or REST API data exchange protocol.
- M-bus to Ethernet gateway:
  - Supports wired communication with smart electricity distribution meters with lower nominal power ratings (< 50 kW) by the manufacturer Landis+Gyr and Iskraemeco.
  - The interface can be powered by wire via Ethernet (PoE) or from a 12 VDC power supply unit.
  - The interface connects to the LAN via Ethernet (wired connection) or wirelessly via a Wi-Fi communication interface.
  - The interface communicates with the main HEMS control via MQTT or REST API data exchange protocol.
- RS-485 to Ethernet gateway:
  - It supports wired communication via MODBUS RTU protocol with the devices that support MODBUS communication (heat pumps, car chargers, solar power plant inverters, industrial distribution meters for electricity with power capacity of over 50 kW).
  - The interface can be powered by wire via Ethernet (PoE) or from a 12 VDC power supply unit.
  - The interface connects to the LAN by Ethernet (wired connection) or wirelessly via a Wi-Fi communication interface.
  - The interface communicates with the main HEMS control via MQTT or REST API data exchange protocol.
- Smart plugs to Ethernet gateway:
  - Smart plugs allow electricity consumption measurement and ON/OFF control for user devices that do not support other digital communication protocols that would allow monitoring their consumption and controlling them.
  - The smart plug is powered from 230 VAC power mains at the intended spot of user device control.
  - A smart plug connects to the LAN wirelessly via a Wi-Fi network.
  - The smart plug communicates with the main HEMS control via MQTT or REST API data exchange protocol.

The figure below illustrates an example of connecting an integrated HEMS complete with external interfaces (Figure 9).

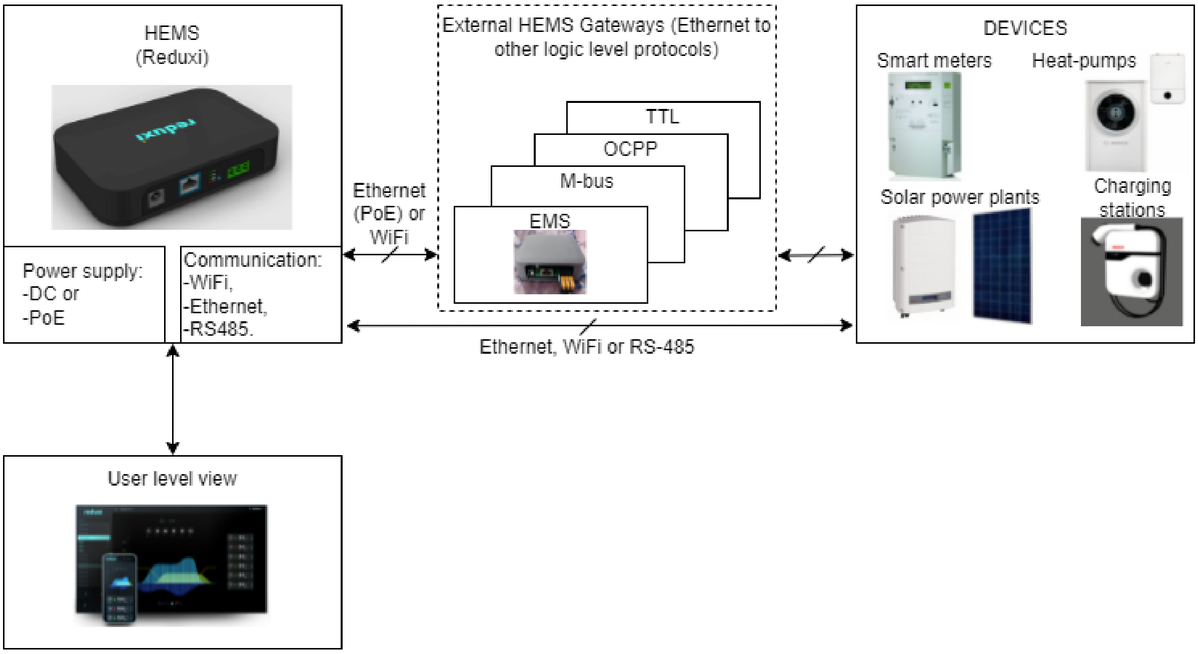

Figure 9: Connection of the HEMS with end user devices

3.3.3.2 Measurement and control signals from a HEMS device

The external measurement/control units referred to above, which communicate with the main HEMS control unit, can be used to address many devices within the iFLEX project at the user's site. The most commonly addressed devices within the pilot area in Slovenia include the following:

- heat pump,
- domestic hot water heater,
- solar power plant inverter (PV inverter),
- distribution electricity meter,
- other electricity meters.

3.3.3.3 Heat pump

The HEMS device and relevant external measurement/control units allow obtaining measurement data from the heat pump, as well as controlling the heat pump. Measurement/control signals are read or sent to the heat pump via EMS or MODBUS RTU (RS-485) communication protocol.

The following measurement data can be read from the heat pump:

- electric energy consumed,
- electric energy produced,
- coefficient of performance (COP) defined as electric energy consumed / electric energy produced,
- temperature of the supply (outlet/hot water) line of the hot water tank,
- temperature of the return (cold water) line of the hot water tank,
- temperature inside the hot water tank,
- temperature at respective thermostats installed within the premises,
- environment temperature as measured at the external unit of the heat pump.

Write or control signals can also be sent to the heat pump for the following parameters:

- changing the desired temperature at each respective thermostat,
- changing the desired temperature inside the hot water tank,
- switching the heat pump off and on.

3.3.3.4 Domestic hot water heater

Most domestic hot water heaters use a simple thermostat for temperature control. The thermostat senses the temperature of the hot water inside the heater and switches the heater on or off depending on the desired/set temperature. The main HEMS unit connects to the domestic hot water heater via an external smart plug that allows power consumption measurement and control of the heater. The smart plug intended for domestic hot water heater control allows reading the following measurement signals:

- electricity consumption,
- domestic hot water temperature measurement (optionally via a PT100 temperature sensor).

The domestic hot water heater can also be controlled with the ON/OFF mode via a relay switch. Currently, the smart plug allows control of single-phase alternating current (AC) devices with a nominal voltage of 230 VAC and a maximum current of 16 A (which is equivalent to ~3,600 WAC).

3.3.3.5 Solar Power plant Inverter (PVI)

A solar inverter, or photovoltaic inverter, is the component of a solar power plant that converts the Direct Current (DC) output of a photovoltaic solar panel into Alternating Current (AC). In the Slovenian pilot, we address the users with installed self-sufficient supply solar power plants, the power of which is defined by a maximum of 80 % of the residential unit's connected load to the distribution network. For example, if a unit is connected to the distribution network via three 25-amp fuses (3× 25 A), which corresponds to a connected load of 17 kW, then power from the solar plant can be fed back into the network with a maximum of 80 % of connected load, which is 13,6 kW in this particular case. Many users opt for solar power plants that can, at a given moment, feed into the network electric energy with a power that exceeds the limit specified by the power distribution company. At such times, the difference between the maximum allowed power sent to the grid and the available power of the solar power plant is lost. Such loss represents the potential to use the iFLEX project to exploit within the residential unit the energy that would otherwise have been lost. For this purpose, we include solar power plants in the Slovenian part of the iFLEX project, which can be monitored and controlled as necessary via the main HEMS unit and external measurement/control units. The following measurement data can be read from the solar inverter:

- voltage and power on each phase of an alternating connection,
- frequency,
- current solar power plant power,
- electric energy produced.

At a certain point, the inverter can also be controlled to decrease the power of feeding excess electric energy into the grid (in percentages relative to the inverter's nominal power, e.g., 100 % means full power of feeding power into the grid) if it is found that this would improve the network conditions.

3.3.3.6 Distribution electricity meter

Distribution electricity meter is a device installed by the power distribution company at the point where a residential unit is connected to the distribution network (power grid). In Slovenia, electricity meters by the manufacturers Landis+Gyr and Iskraemeco are predominantly used. These meters allow the power distribution company to collect data at 15-minute intervals and transfer them once per day to their information system via GSM communication or via PLC communication (communication via power lines). The meters installed at the unit's connection point allow one-way communication for reading measurement data intended for the user. Readings of measurement data is enabled via M-bus (meter bus) or via RS-485 (MODBUS RTU) communication protocol. The following measurement data can be read from the distribution electricity meter, using the main HEMS unit, at an interval of a minimum of five seconds:

- current and voltage on each phase,
- frequency,
- power of feeding the excess electricity into the grid and consumption,
- reactive and active electric energy,
- date and time.

3.3.3.7 Example of installation of a HEMS complete with external units

In the Slovenian pilot region, we mostly address the users with installed heat pumps, domestic hot water heaters, distribution meters, and solar power plants. We will seek to present with a specific example, how we equip the users who agree to take part in the iFLEX project.

Figures 10-13 illustrate a case of a user with an installed heat pump BOSCH Compress 6000, with a hot water heater/tank with a capacity of 350 litres. Also installed on the premises is a solar power plant with a capacity of 14 kW, which forms a whole together with a solar (PV) inverter by the manufacturer SolarEdge SE17K. Protective fuses of 3 × 25 A are installed at the connection point (connected load of 17 kW), and a distribution electricity meter by the manufacturer Landis+Gyr E450.

The heat pump is connected to the main HEMS unit via EMS communication interface (the interface is connected to the heat pump control unit). Connection of the HEMS unit is presented by the following figure (Figure 10).

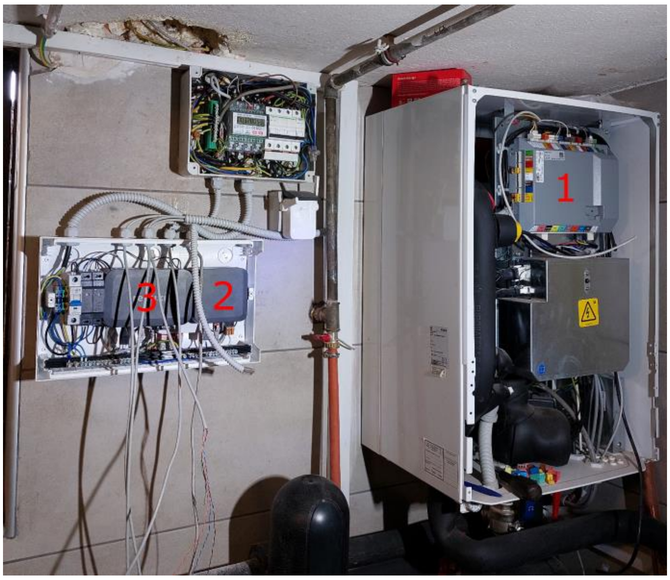

Figure 10: Connection to the heat pump; 1. Heat pump control unit; 2. EMS communicator; 3. Main HEMS control unit

The EMS communicator is connected to the heat pump control unit via connections (lines) BBT, BBTgnd, and BBT 12 VDC intended for heat pump diagnostics. The connection is depicted in the following figure (Figure 11).

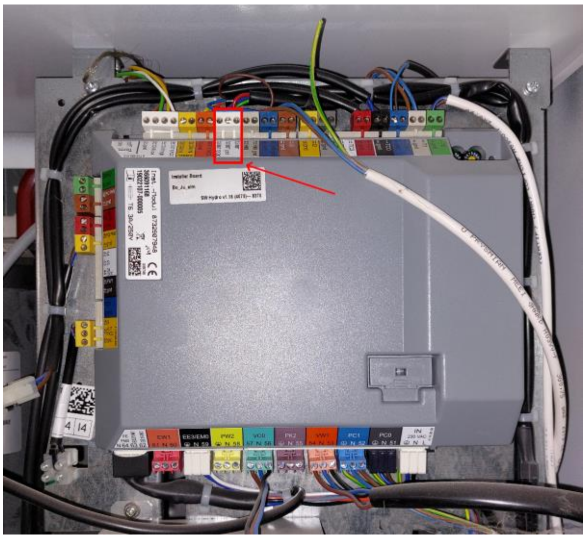

Figure 11: Connection point for connecting the EMS communicator to the heat pump control unit

Since the user has a self-sufficient supply solar power plant installed at the premises, connection to the SE17K solar inverter was made via communication protocol MODBUS RTU. The connection is made via a two-wire (two-conductor) twisted communication line. HEMS unit is connected to the solar inverter via connection points A and B on the communication port RS-485-2 (Figure 12).

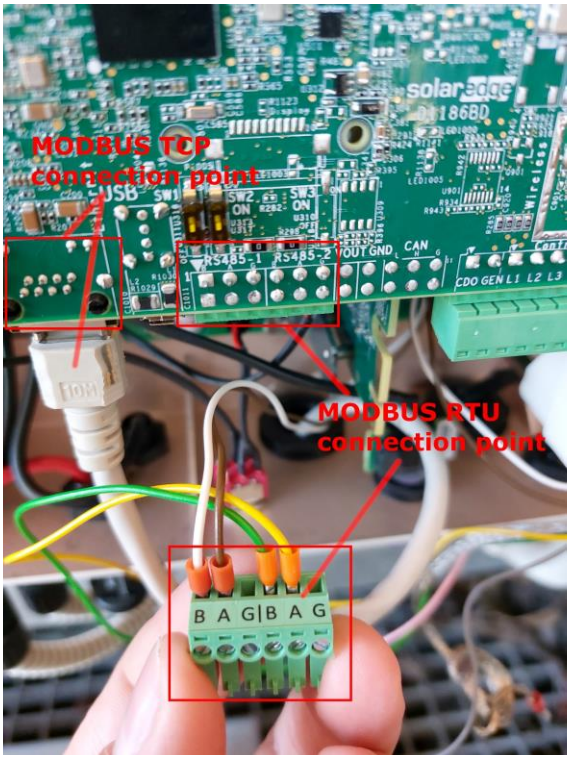

Figure 12: The connection point for the HEMS unit to the self-sufficient supply solar power plant via MODBUS RTU communication protocol

The user has a distribution electricity meter supporting the M-bus communication protocol, installed at the connection point. It follows that the HEMS device is connected to the distribution electricity meter via a two-wire (two-conductor) twisted line and connection points (terminals) labelled 28 and 29. The connection to the distribution electricity meter is depicted in the figure below (Figure 13).

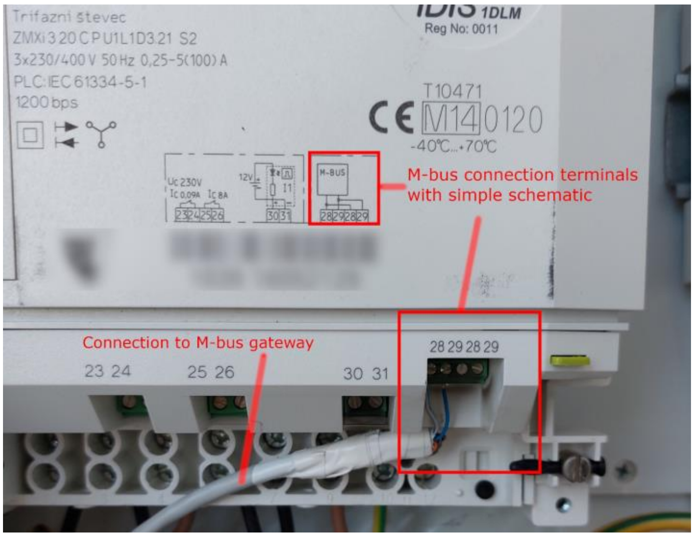

Figure 13: Connection to the distribution electricity meter Landis+Gyr via M-bus one-way communication protocol

3.3.4 HEMS deployment and integration illustration

The HEMS is integrated with the RAI via MQTT protocol. Three major types of messages are supported, defined in D4.2 [9]: device configuration messages, device state messages and measurement messages. Each message is communicated over its own, household specific topic. The HEMS MQTT part resides at deployed Reduxi HEMS modules, at each household. Example of messages exchanged are given below, using the control tool provided to partners. The tool queries the RAI database first for the household data, then for the devices configuration data. The configuration data is essentially the configuration message received through the MQTT. The configuration data reports on devices, attached to HEMS, their properties, sensors and controls through actuators. The last message is presented as a query for the devices state. Again, the state is as reported trough MQTT. Three messages are denoted with darker grey colour for easier navigation.


```
xyz:~/delo/src/iflex/ece:> python3 control.py -k certs/si.manager@iflex.p12 -e 4JDB-OR78-0J8L-WY39 -t
INFO   2022-12-01 14:44:57,169 control [85]: -> Household 325adfaf-0371-4dad-9ac4-789adab7c0ca info:
INFO   2022-12-01 14:44:57,169 control [86]: {
  "id": "325adfaf-0371-4dad-9ac4-789adab7c0ca",
  "bt": "2022-11-24T13:34:21.661000",
  "connection_type": 3,
  "current_limit": 25,
  "geolocation": {
    "latitude": 46.xyz,
    "longitude": 15.xyz
  },
  "hems_id": "4JDB-OR78-0J8L-WY39",
  "household_data": null,
  "household_type": "office",
  "p_id": "101",
  "power_limit": 17,
  "tariff_type": "one_tariff"
}
INFO   2022-12-01 14:44:57,185 control [94]: -> HEMS 4JDB-OR78-0J8L-WY39 configuration:
INFO   2022-12-01 14:44:57,185 control [95]: {
  "elements": [
    {
      "device_id": 2,
      "device_name": "ECE kuhinja",
      "device_type": "Actuator",
      "control": [
        {
          "control_type": "on/off",
          "actuator_id": 284,
          "control_name": "Defines on/off control capabilities of a device"
        }
      ],
      "sensors": [
        {
          "sensor_id": 289,
          "sensor_name": "Voltage L1",
          "sensor_unit": "V"
        },
        {
          "sensor_id": 290,
          "sensor_name": "Voltage L2",
          "sensor_unit": "V"
        },
        {
          "sensor_id": 291,
          "sensor_name": "Voltage L3",
          "sensor_unit": "V"
        },
        {
```

```
        "sensor_id": 286,
        "sensor_name": "Current L1",
        "sensor_unit": "A"
      },
      {
        "sensor_id": 287,
        "sensor_name": "Current L2",
        "sensor_unit": "A"
      },
      {
        "sensor_id": 288,
        "sensor_name": "Current L3",
        "sensor_unit": "A"
      },
      {
        "sensor_id": 283,
        "sensor_name": "Active power",
        "sensor_unit": "W"
      },
      {
        "sensor_id": 517,
        "sensor_name": "Import - active energy",
        "sensor_unit": "Wh"
      },
      {
        "sensor_id": 284,
        "sensor_name": "State code",
        "sensor_unit": ""
      },
      {
        "sensor_id": 285,
        "sensor_name": "State name",
        "sensor_unit": ""
      },
      {
        "sensor_id": 282,
        "sensor_name": "Import - active power",
        "sensor_unit": "W"
      },
      {
        "sensor_id": 281,
        "sensor_name": "Communication error",
        "sensor_unit": ""
      }
    ]
  },
  {
    "device_id": 14,
    "device_name": "Razsmernik soncne brez DC napajanja",
    "device_type": "Solar",
    "sensors": [
      {
        "sensor_id": 49,
        "sensor_name": "Current L1",
        "sensor_unit": "A"
      },
      {
        "sensor_id": 50,
        "sensor_name": "Current L2",
        "sensor_unit": "A"
      },
      {
        "sensor_id": 51,
```

```
          "sensor_name": "Current L3",
          "sensor_unit": "A"
        },
        {
          "sensor_id": 52,
          "sensor_name": "Voltage L1",
          "sensor_unit": "V"
        },
        {
          "sensor_id": 53,
          "sensor_name": "Voltage L2",
          "sensor_unit": "V"
        },
        {
          "sensor_id": 54,
          "sensor_name": "Voltage L3",
          "sensor_unit": "V"
        },
        {
          "sensor_id": 264,
          "sensor_name": "Active power",
          "sensor_unit": "W"
        },
        {
          "sensor_id": 47,
          "sensor_name": "Export active energy",
          "sensor_unit": "Wh"
        },
        {
          "sensor_id": 265,
          "sensor_name": "Import active power",
          "sensor_unit": "W"
        },
        {
          "sensor_id": 48,
          "sensor_name": "Export active power",
          "sensor_unit": "W"
        },
        {
          "sensor_id": 142,
          "sensor_name": "Communication error",
          "sensor_unit": ""
        }
      ]
    }
  ],
  "hems_id": "4JDB-OR78-0J8L-WY39",
  "ts": "2022-11-28T19:18:43.523000"
}
INFO    2022-12-01 14:44:57,304 control [106]: -> HEMS 4JDB-OR78-0J8L-WY39 devices state:
INFO    2022-12-01 14:44:57,304 control [107]: {
  "devices_state": [
    {
      "device": 2,
      "state": [
        {
          "bt": "2022-11-28T19:17:41.765000",
          "actuator_id": 284,
          "vs": "off"
        }]}]}
```

The same tool is used below to demonstrate sending of a control message to the household. The message indicates turning of the device 2 actuator 284 on. The message is not sent, -m switch of the tool indicates to generate the message only.

```
xyz:~/delo/src/iflex/ece:> python3 control.py -k certs/si.manager@iflex.p12 -e 4JDB-OR78-0J8L-WY39 -d 2 -a 284 -c on -m
INFO    2022-12-01 14:57:01,391 control [134]: {
  "devices_control": [
    {
      "device": "2",
      "control_device": {
        "actuator_id": "284",
        "command": "on"
      }
    }
  ]
}
```

The measurements are sent to the broker in a form of simple SenML messages as is presented below. The message reports just one measurement in time of the device 3, sensor 160 measurement. The measurement is Import active energy, measured in Wh, with a value of 1,25.


```
{
  "bn": "urn:dev:3:160",
  "t": 1666775763785,
  "n": "Import active energy",
  "u": "Wh",
  "v": 1.25
}
```


3.3.4.1 RAI deployment illustration

The RAI module in the Slovenian pilot is deployed at JSI premises in a local virtualised cloud environment. The deployment uses Linux LXC container virtualisation for production environment RAI provisioning. The deployment contains both a RAI server as well paired MQTT broker. The server uses Python, MongoDB, Mosquitto, Tornado web server and other tools for the RAI implementation. The services are managed in Linux system start-up scripts style. The Tornado web server is used to provide RAI REST API implementation. At JSI we are working on monitoring solutions for deployed services as well as on developing a Kubernetes environment with proper continuous deployment/continuous integration (CD/CI) functionality for dockerized services.

The RAI interface is used in a programmable manner as is shown below. At first necessary data structures are initialised. Then a Python requests module is used to access the server and retrieve via 'households/<household_id>/building' interface household building parameters. The parameters are used to build a Digital Twin of the household, among the other data.

```
pcs_interface = 'https://si.pcs.e5.ijs.si:8443/iflex/v1/%s'
# Debugging interface
#pcs_interface = 'https://si.pcs.e5.ijs.si:8444/iflex/v1/%s'
ks_file = 'certs/si.manager@iflex.p12'
ks = KeyStore.load(ks_file)
headers = requests.utils.default_headers()
headers['User-Agent'] = 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87
Safari/537.36'

hh_id_one = "1dc43de2-8ac1-4a95-82a1-76bc83e937f1"

url = pcs_interface % 'households/%s/building'
r = requests.get(url % hh_id_one,
            cert=(ks.get_certificate_file(),
                ks.get_privatekey_file()),
            verify=ks.get_ca_certificates_file(),
            headers=headers)
pprint(r.json())

{'air_ventilation': 0.35,
 'bt': '2022-11-30T21:54',
 'floor_area': 213.2,
 'floor_height': 2.4,
 'heatpump_model': 'Bosch Compress 6000 AW 9',
 'hems_id': '64JG-U1PL-UUNQ-ZM23',
 'id': '1dc43de2-8ac1-4a95-82a1-76bc83e937f1',
 'pv_data': {'pv_orientation': 'JZ',
          'pv_power': 15.58,
          'pv_power_limit': 13.6,
          'pv_tilt': 30},
 'temperature_set_point': 22.0,
 'thermal_capacitance_per_floor_area': 165000,
 'u_walls': 2,
 'u_windows': 1.2,
 'ventilation_efficiency': 0,
 'walls_area': 240,
 'weather_location': 'Nova cerkev',
 'window_area_south': 29}
```

In a similar manner the measurement data is obtained and then plotted, as presented in Figure 14. The figure presents power measurements in W for all three phases from a household with installed PV and a heat pump.

3.3.4.2 Digital Twin deployment illustration

The Digital Twin is deployed alongside the RAI module. The DT uses the data directly from the RAI database storage which improves performance over obtaining the data through the REST interface. The twin uses JRC PVGIS[9] for modelling irradiation and household solar gains, the heat demand implementation is initially based on JSI/Comsensus flexibility calculator implementation[10]. The twin provides REST interface for loading and storing Machine Learning (ML) models and for obtaining heat and heat consumption related predictions. An example of a such prediction for the same household as presented in Figure 15.

[9] See PVGIS homepage: https://joint-research-centre.ec.europa.eu/pvgis-photovoltaic-geographical-information-system_en
[10] See the Gitlab repository for details: https://gitlab.com/comsensus/flexibilitycalculator

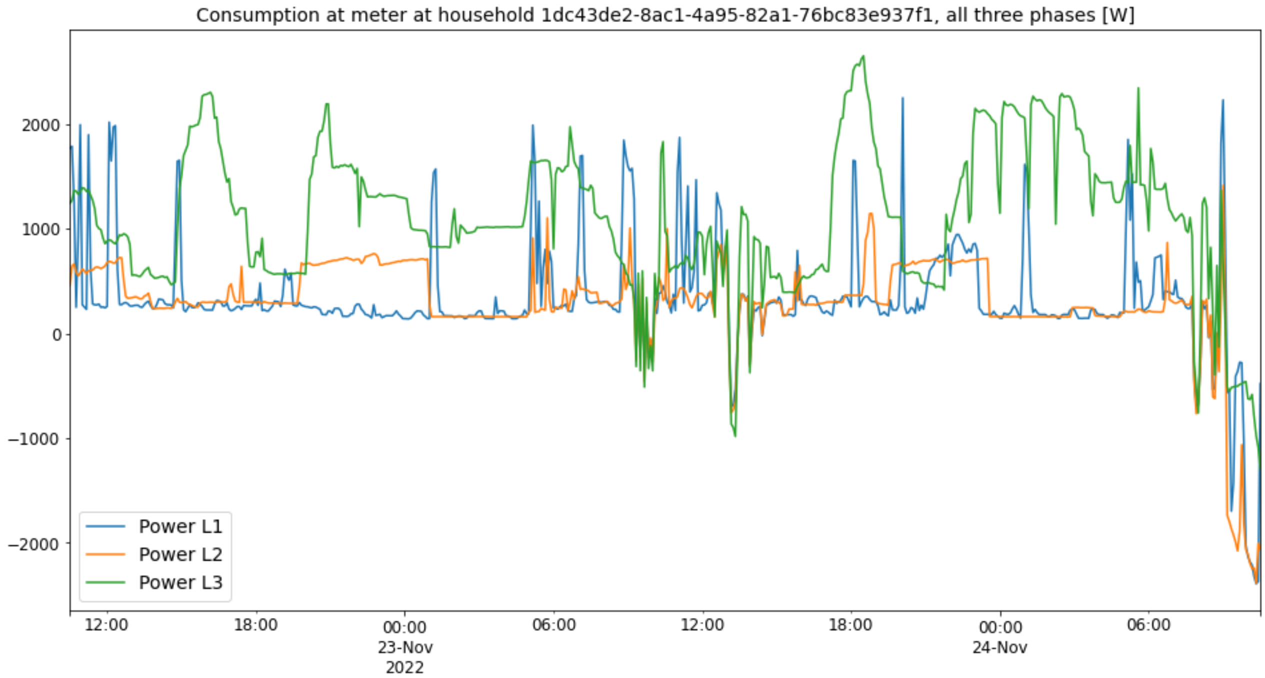

Figure 14: Example of plotted measurement data obtained from a household meter

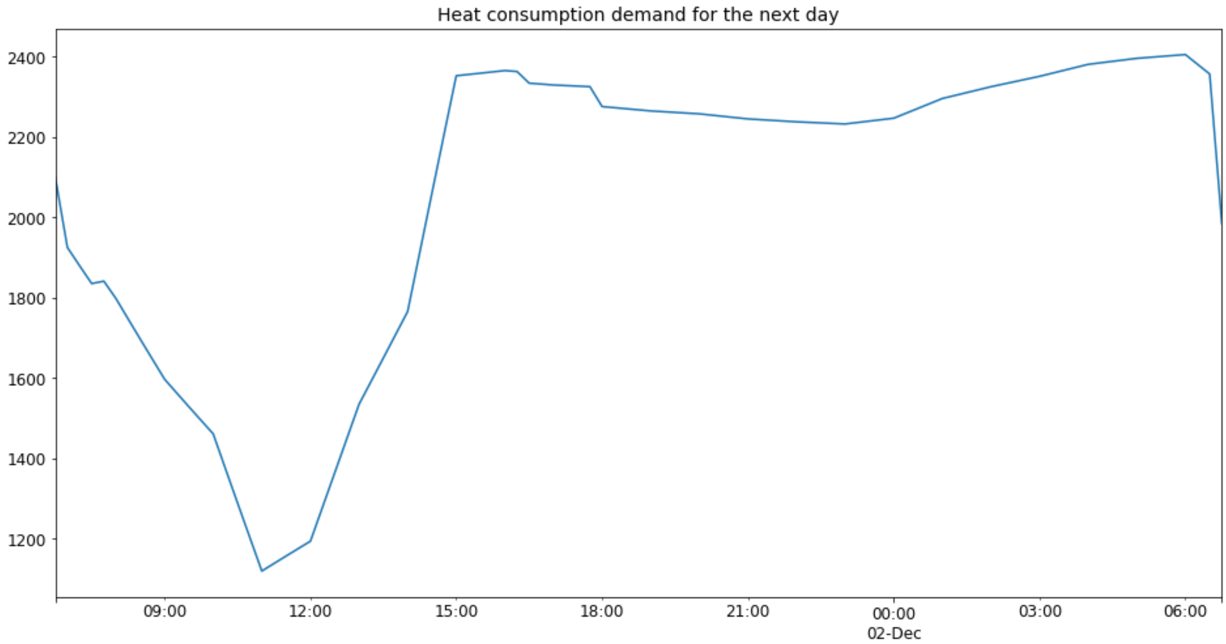

Figure 15: Heat consumption demand for the household with id 1dc43de2-8ac1-4a95-82a1-76bc83e937f1

Modelled COP parameters for the BOSCH Compress 6000 AW 9 are stored in the DT storage and can be used to assess heat pump COP characteristics when the calculations for heat consumption demand are prepared. The Bosch COP model can be obtained from DT API, as shown below.

```
url = rai_interface % 'households/digital_twin/models'
qparams = {'type': 'COP'}
m2url = url + '?' + urllib.parse.urlencode(qparams)
r = requests.get(m2url,
            cert=(ks.get_certificate_file(),
                ks.get_privatekey_file()),
            verify=ks.get_ca_certificates_file(),
            headers=headers)
print(r.ok)
model_data = r.json()
```

The URL in its query part asked for all models of type COP.

```
True
https://si.pcs.e5.ijs.si:8443/iflex/v1/households/digital_twin/models?type=COP
```

Each model is defined with the following keys:

- device_type: heat pump or air-conditioning
- model_type: specific type/model of the device
- type: type of the model
- bt: base time of model creation
- model: the model itself

```
bosch_model = model_data['models'][0]
print(bosch_model.keys())
dict_keys(['device_type', 'model_type', 'type', 'bt', 'model'])
```

The model itself, or only part of the model is presented in the code output below. The model is prepared per temperature degree granulation. At each temperature, based on original graphic characteristics, a polynomial third degree model is calculated.

```
cop_model = bosch_model['model']
print(cop_model)

{'25': {'intercept': 4.776564176564178,
  'coefficients': [0.14886792220125553,
   0.001364746364746358,
   -8.265561598894914e-05]},
```

The COP diagram based on obtained model for four temperatures, namely at 30, 40, 50 and 60 degrees Celsius is shown in Figure 16.

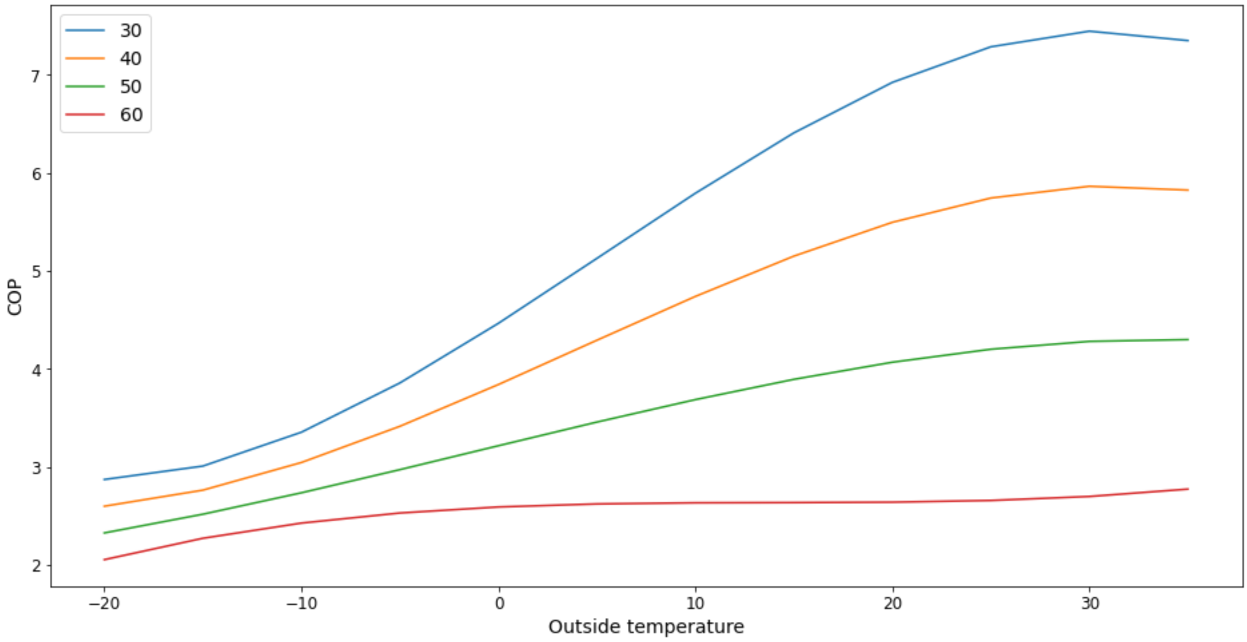

Figure 16: BOSCH Compress 6000 AW 9 heat pump COP parameters model. The curves present characteristics at different inlet water temperature

3.3.5 SCOM Demand Response Management System

SCOM's DRMS prototype will be deployed and the focus of the second phase is on piloting the following functionalities:

- FN-DR-01 – iFA end-users' flexibility potential
- FN-DR-03 – Sending Flexibility Signal
- FN-DR-05 – Information on participation in explicit DR actions
- FN-DR-07 – Interface for parameterizing flexibility services
- FN-DR-09 – Flexibility dispatch
- FN-DR-10 – Provide activated flexibility report

As explained in the project deliverable D4.5 [5] the solution combines SCOM's preexisting DRMS system, whose backend has been integrated with the OpenLEADR, an open-source implementation of the OpenADR2.0 protocol, to communicate with the Aggregation and Market interface component of the iFA. Extensions to this protocol will be utilised to meet iFLEX requirements when necessary. The DRMS prototype encompasses the OpenADR server side (VTN – Virtual Top Node), which is complemented by its client-side counterpart (VEN – Virtual End Node) provided by the Aggregator & Market interface, which is presented in the next section.

3.3.6 Aggregator & Market Interface Module

The Aggregator and Market Interface Module [5] will follow the implementation presented in the Greek pilot, supporting DR communications via OpenADR2.0 protocol – and extensions when necessary – with SCOM's DRMS, as well as interfacing with other iFA's components. It will also provide an interface, enabling the communication of electricity and network tariffs.

The next functionalities shall be supported by the A&M Interface Module [5] by the end of Phase 2 of the Slovenian pilot:

- FN-AM-04 – Information on participation in explicit DR actions
- FN-AM-05 – Communication of flexibility potential
- FN-AM-06 – Access to DR reports to end users
- FN-AM-08 – Receiving flexibility signal
- FN-AM-09 – Communication of flexibility signal

- FN-AM-10 – Response to flexibility signal (explicit DR)
- FN-AM-11 – Communication of network tariffs from external system
- FN-AM-12 – Access to network tariffs to end users
- FN-AM-13 – Communication of electricity tariffs from external system
- FN-AM-14 – Access to electricity tariffs to end users
- FN-AM-17 – Access of flexibility validation data from the RAI
- FN-AM-18 – Communication of flexibility validation data
- FN-AM-19 – Assessment of end user participation in a DR event
- FN-AM-20 – Communication of participation assessment data

3.3.7 User Interface App

The UI for the Slovenian pilot has been developed as a native mobile app, based on the React Native framework[11]. In second pilot phase, it should be enhanced with new features and integrated with the other iFA's components of the Slovenian pilot, thus supporting a wide range of functionalities. More specifically, it should meet the following requirements [3] by the end of this phase:

- FN-UI-01 – Operation mode customization
- FN-UI-02 – User-defined time and operational constraints
- FN-UI-03 – End-user feedback
- FN-UI-04 – Optimization policy selection
- FN-UI-05 – Automation level customization
- FN-UI-07 – Supported system interface languages
- FN-UI-08 – Provision of consent for the schedules of dispatchable assets
- FN-UI-09 – DR notification policy
- FN-UI-10 – Insights into sustainability metrics
- FN-UI-11 – Real-time energy data
- FN-UI-12 – Past energy data
- FN-UI-13 – Flexibility reports
- FN-UI-14 – Insights into energy efficiency
- FN-UI-15 – Customized alerts
- FN-UI-16 – Energy advising service
- FN-UI-17 – Inspection of energy tariffs
- FN-UI-21 – DR event notification
- FN-UI-22 – Presentation of DR event history
- FN-UI-27 – Actual schedules of assets
- FN-UI-28 – Estimation of energy costs
- FN-UI-29 – Savings thanks to iFA's operation
- FN-UI-02 – Easy on-boarding in iFA's usage

Figure 17 shows a mock-up of the screen that presents the daily electricity tariffs' fluctuation to the users. This is amongst the features to be added to the iFA mobile application in second piloting phase.

[11] https://reactnative.dev/

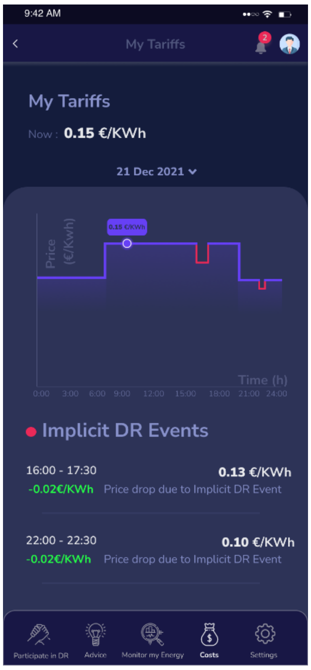

Figure 17: Mock-up of the “My Tariffs” screen of the mobile application

4 Conclusion

Deliverable 6.6 is the fourth [12] [13] [14] commandment document, which is oriented towards prototype implementation and deployment in the second piloting phase, which includes Finland, Greece and Slovenia pilot partners. It focuses on the practical implementation of the functional components listed in deliverable 7.1 Initial Pilot Specification.

In this document, the Finnish pilot partners focus on the deployed results of the function blocks (Resident interface, Facility manager interface, RAI, AFM, DTR and WSI), their communication between iFA and BEMS, the way of capturing environmental/physical data (average humidity and temperature, apartment building electric and heating power consumption, CO_2 presence and other control related parameters/signals) and their further processing to ensure optimal iFA flexibility.

Greek pilot partners present their results, which are related to the implementation of the iFA User Interface application, the existing systems and applications that are extended to support iFA functionalities for the 2nd phase and mocked iFA components. The communication channels mechanisms (Push Notification mechanism, Security Mechanisms, Error Messages to user, application deep-linking) that will be used for integration among HERON's and ICOM's applications will be tested at the end of 2nd piloting phase. Furthermore, ICOM's DRMS is presented in 1st and deployed in 2nd phase of the project, supporting a REST API for flexibility requests as well as communication between iFA and DRMS via OpenADR2.

In the Slovenian second piloting phase, the HEMS installation is deployed, and the first user was equipped with HEMS controller and external power measurements components. The first results of high-level REST API and MQTT communication with HEMS cloud controller were shown. Furthermore, the data collected from HEMS cloud controller were shown in simple user graphical web application (GUI developed in python). First revised iFA framework components for Slovenian second piloting phase were presented and described in detail. For the user interaction with iFA, a mobile application with revised DR functionalities, developed in React Native framework is available to connect with RAI. DR communication will take place via OpenADR2 protocol, supported from OpenLEADR opensource implementation.

5 List of figures and tables

5.1 Figures

5.2 Tables

6 References

[1] “D2.1 - Use cases and Requirements,” 2021.
[2] “D2.3 - Common architectureof iFLEX Framework,” 2021.
[3] iFLEX project, “D3.5 - Revised Natural User Interfaces,” 2022.
[4] IEC, “IEC 62746-10-1Systems interface between customer energy management system and the power management system - Part 10-1: Open Automated Demand Response (OpenADR)”.
[5] iFLEX project, “D4.5 - Revised Market and Aggregation Interface Module,” 2022.
[6] “Revised Pilot Specifications. The IFLEX deliverable D7.2.,” 2022.
[7] “Revised Pilot Specifications. iFlex deliverable D2.4,” 2022.
[8] “Secure prosumer data management module. The iFLEX project deliverable D4.7.,” 2021.
[9] “D4.2 Revised Resouce Abstraction Interface,” 2022.
[10] “D3.2 Revised hybrid-modelling module,” 2022.
[11] “D4.8 - Revised secure consumer data management module,” 2022.
[12] D6.2 Common iFLEX Framework, 2021.
[13] D6.3 Refined Common iFLEX Framework, 2022.
[14] D6.5 Initial application-specific iFLEX Assistant prototype, 2022.
[15] “Initial Pilot Specifications. The iFLEX deliverable D7.1.,” 2021.
[16] “Initial Hybrid-modelling Module. The iFLEX project deliverable D3.1.,” 2021.