Intelligent Assistants for Flexibility Management (Grant Agreement No 957670)

D6.2 Common iFLEX Framework

Date: 2021-08-31

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: D6.2 Common iFLEX Framework_v0.3_INJET.docx
Document version: 1.0
Document owner: **VTT**

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	Anne Immonen, Jussi Kiljander (VTT)	2021-05-03	Initial ToC
0.2	Anne Immonen, Jussi Kiljander (VTT), Ilias Lamprinos (ICOM), Nikos Charitos (ICOM), Dusan Gabrijelcic (JSI), Andrej Krpič (SCOM)	2021-08-16	Description of main components
0.3	Anne Immonen, Jussi Kiljander (VTT)	2021-08-26	Version submitted for internal review
0.4	Louisa Birch Riley (IN-JET), Jemina Leino (Caverion)	2021-08-27	Internal review
1.0	Anne Immonen, Jussi Kiljander (VTT)	2021-08-31	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Louise Birch Riley (IN-JET)	2021-08-26	Approved with minor comments
Jemina Leino (Caverion)	2021-08-27	Approved with minor changes

Index:

List of abbreviations

Abbreviation	Term
A&M (Interface)	Aggregator and Market (Interface)
ADR	Automated Demand Response
AFM	Automated Flexibility Manager
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
API	Application Programming Interface
BEMS	Building Energy Management System
BRP	Balance Responsible Party
DB	Database
DH	District Heat
DR	Demand Response
DRMS	Demand Response Management System (from OpenADR terminology)
DT	Digital Twin
ENTSO-E	European Network of Transmission System Operators for Electricity
FOI	Flexibility Operator Interface
FMI	Finnish Meteorological Institute
HEMS	Home Energy Management System
iFA	iFLEX Assistant
MQTT	Message Queuing Telemetry Transport
OASIS	Organization for the Advancement of Structured Information Standards
oBIX	open Building Information Xchange
RAI	Resource Abstraction Interface
REST API	Representational State Transfer Application Programming Interface
SaaS	Software as a Service
UI	User Interface

1 Executive summary

Task 6.2 *Integration of submodules into a common software framework* is responsible for integration of the components developed in the technical work packages into the common iFLEX software framework. It is based on Task 6.1 *Continuous integration and deployment planning* that defines the continuous integration (and deployment plan) and mechanisms, and Task 2.3 *iFLEX Framework architecture design* that specifies the iFLEX Assistant system architecture. As D2.3 provides the general document for integration by defining the component interfaces at the logical level, T6.2 executes the co-development and integration at the practical level (i.e., software implementation and packaging). Thus, the common iFLEX software framework provides the libraries, scripts and tools for creating application-specific iFLEX Assistant instances.

The common iFLEX framework consists of seven functional components; Aggregator and Market Interface, End-user Interface, Resource Abstraction Interface, Automated Flexibility Management, Digital twin repository, Weather service and Security and privacy interface. Aggregator and market interface and End-user interface enable the interaction between the relevant actors and the iFLEX Assistant, whereas Resource abstraction interface enables iFLEX Assistant to monitor and control relevant aspects in the consumer/prosumer premises. Automated flexibility management provides services for implementing demand-response control and optimization operations. Digital twin repository estimates the impact of a consumer and provides optimal and consumer-centric flexibility management. Weather service provides country-specific data about weather forecasts, and finally Security and privacy interface ensures trust and security between all the elements of iFLEX framework.

The development of the common iFLEX framework is a continuous and iterative process executed in three phases. This document describes the high-level summary of the first phase implementation of the seven functional components. For household specific iFLEX Assistants (Slovenia and Greek pilots), the implementation concentrates on Resource abstraction interface to enable data collection and control of flexible assets via smart meters and Home Energy Management Systems (HEMS), and on mock-up implementation of the End-user Interface to collect feedback on the design from pre-pilot participants and allow co-creation of new functionalities. For apartment buildings targeted iFLEX Assistant (Finnish pilot), the implementation focuses on data collection, flexibility management experiments and testing. Also, the goal is to experiment with the Automated flexibility management and associated Digital twin repository modules, and to deploy and collect feedback via an End-user interface designed for residents of an apartment building.

2 Introduction

2.1 Purpose, context and scope

This demonstrator type deliverable that documents the initial version of the iFLEX Framework.The iFLEX Framework is a software framework for development of intelligent assistants for flexibility and holistic energy management. It consists of the following modules: Aggregator and Market Interface, End-user Interface, Resource Abstraction Interface, Automated Flexibility Management, Digital twin repository, Weather service and Security and privacy interface. The iFLEX Assistant is an application-specific instance of an intelligent assistant developed on top of the common iFLEX Framework.

The iFLEX Framework development is an iterative process implemented in three phases:

- Phase one: A pre-pilot with the Minimum Viable Product of the iFLEX Framework and Assistants will be carried out with selected users.
- Phase two: A small-scale pilot including the iFLEX Framework with full functionality will be validated with small-scale pilot groups.
- Phase three: The improved iFLEX Framework will be deployed and validated in large-scale pilots.

This document describes the common iFLEX Framework in phase one.

2.2 Content and structure

This document is structured as follows. Section 3 describes how the iFLEX Framework is related to use cases and iFLEX architecture, and then the focus of the first implementation phase is introduced. Section 4 introduces the main components of the iFLEX Framework. Concluding remarks are provided in Section 5.

3 Overview

3.1 Relation to use cases

High level requirements for the iFLEX Assistants, specified in D2.1 *Use cases and requirements,* are documented in form of use cases. Based on these use cases, component specific requirements have been engineered in the project. These requirements document the functional and non-functional needs that the iFLEX Assistants developed on top of the iFLEX Framework aim to satisfy. The requirements are collected and managed via the project's issue and project tracking tool, Jira. There are currently 81 requirements collected in Jira. Figure 1 illustrates a snapshot of the requirements collected in Jira.

51–81 of 81 Columns

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due
	IF-64	FN-DTR-03 Household flexibility model	Dusan Gabrijelcic	Dusan Gabrijelcic		PART OF SPECIFICATI...	*Unresolved*	09/Jun/21	30/Aug/21	
	IF-75	Access to individual data on user participation in DR events	Nikos Charitos	Thanasis Papaioannou		RESOLVED	Duplicate	07/Jul/21	26/Aug/21	
	IF-4	FN-DTR-02 Model flexibility of apartment building's heating system.	Jussi Kiljander	Jussi Kiljander		RESOLVED	Duplicate	21/May/21	30/Aug/21	
	IF-3	FN-DTR-03 Model electricity demand of inflexible loads in an apartment building	Jussi Kiljander	Jussi Kiljander		RESOLVED	Duplicate	19/May/21	30/Aug/21	
	IF-5	FN-DTR-01 Model electricity load of a household	Dusan Gabrijelcic	Jussi Kiljander		RESOLVED	Duplicate	21/May/21	30/Aug/21	
	IF-2	System Y performance	*Unassigned*	Jesper Thestrup		REOPENED	*Unresolved*	16/Apr/21	27/Aug/21	
	IF-71	FN-AFM-03 Activate offered flexibility	Jussi Kiljander	Jussi Kiljander		IN PROGRESS	*Unresolved*	15/Jun/21	15/Jun/21	
	IF-69	FN-AFM-01 Provide baseline forecasts	Jussi Kiljander	Jussi Kiljander		IN PROGRESS	*Unresolved*	15/Jun/21	15/Jun/21	
	IF-7	FN-UI-02 - User-defined time and operational constraints	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	21/May/21	26/Aug/21	
	IF-6	FN-UI-01 - Operation mode customisation	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	21/May/21	26/Aug/21	
	IF-9	FN-UI-05 - Automation level customisation	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	21/May/21	26/Aug/21	
	IF-51	FN-AM-09 - Communication of Flexibility Signal	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	
	IF-15	FN-UI-21 - DR event notification	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	21/May/21	26/Aug/21	
	IF-17	FN-UI-08 - Provision of consent for the schedules of dispatchable assets	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	21/May/21	26/Aug/21	
	IF-52	FN-AM-10 - Response to Flexibility Signal (explicit DR)	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	•••
	IF-45	FN-AM-03 - Communication of interest in a new flexibility service	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	
	IF-46	FN-AM-04 - Information on participation in explicit DR actions	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	
	IF-41	FN-DR-09 - Flexibility dispatch	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	
	IF-50	FN-AM-08 - Receiving Flexibility Signal	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	
	IF-70	FN-AFM-02 Flexibility potential	Jussi Kiljander	Jussi Kiljander		IN PROGRESS	*Unresolved*	15/Jun/21	27/Aug/21	
	IF-42	FN-DR-10 - Provide activated flexibility report	Nikos Charitos	Nikos Charitos		IN PROGRESS	*Unresolved*	02/Jun/21	26/Aug/21	

Figure 1: Snapshot of the requirements in Jira.

3.2 Relation to the functional architecture of the iFLEX Framework

This deliverable describes the first phase implementation of the iFLEX Framework and covers thus the whole architecture of the iFLEX Framework, depicted in Figure 2. The focus in this deliverable is on a high-level summary of the first phase implementation and it should be used together with D2.4 *Revised Common architecture of iFLEX Framework*, which provides a more detailed description of the iFLEX Framework Architecture.

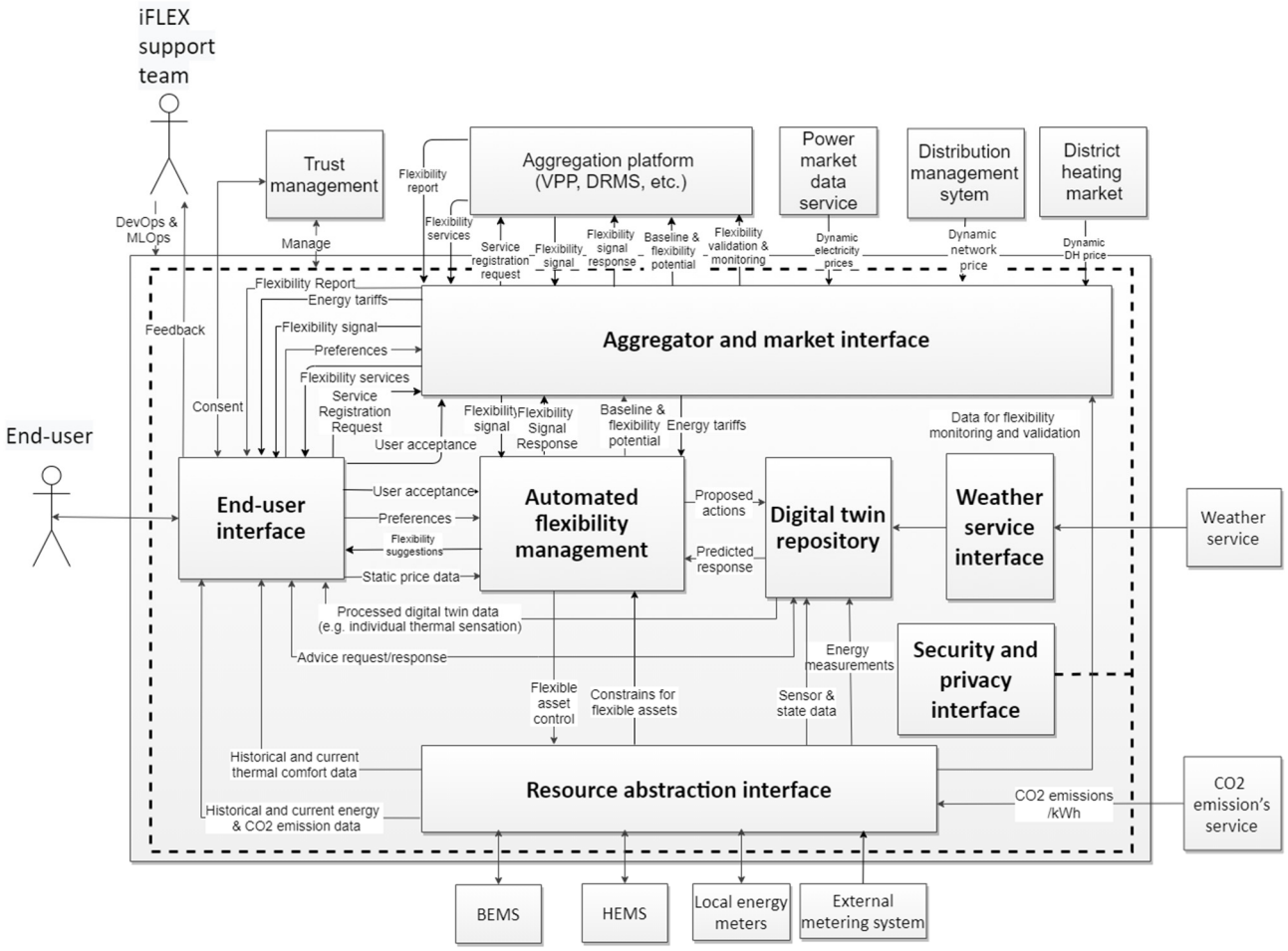

Figure 2: Functional view of the iFLEX Framework.

The iFLEX Framework consists of seven functional components as illustrated in Figure 2. The implementation of the components is documented in this deliverable as follows:

- Section 4.1 introduces 1st phase implementation of the Aggregator and market interface module.
- Section 4.2 describes the End-user interface module.
- Section 4.3 documents the Resource abstraction interface module.
- Section 4.4 introduces the Automated flexibility management module.
- Section 4.5 summarizes the Digital twin repository module.
- Section 4.6 describes the Weather service interface module.
- Section 4.7 summarizes the Security and privacy interface module.

3.3 First phase focus

The focus in the first phase implementation of the iFLEX Framework is to provide tools and libraries for developing the iFLEX Assistants for the pre-pilots. For household specific iFLEX Assistants, to be deployed in Slovenia and Greek pilots, the required functionality focuses on two aspects. First, a Resource abstraction interface to enable data collection and control of flexible assets via smart meters and Home Energy Management Systems (HEMS). Second, mock-up implementation of the End-user Interface to collect feedback on the design from pre-pilot participants and allow co-creation of new functionalities with them.

The first phase implementation of the iFLEX Assistant targeted for apartment buildings (to be deployed in Finnish pilot) focuses on data collection, as well as, on flexibility management experiments and testing. Additionally, the goal of the pre-pilot is to experiment with the Automated flexibility management and associated Digital twin repository modules. Moreover, the goal is to deploy and collect feedback via an End-user interface designed for residents of an apartment building.

To this end, the focus in the first phase has been on implementing new components as well as, extending and integrating baseline components in order to enable the aforementioned functionality to be deployed and tested in the pre-pilots. Moreover, the focus has been on implementing Aggregator and market interface modules to support the full use case to be demonstrated in phases 2 and 3 of the project.

4 Implementation for phase one

The main functional components of the common iFLEX Framework and their interactions are depicted in Figure 3 **Error! Reference source not found.** and described more detailed in the following sub-sections. For each component, the functional description and the description of implementation are provided.

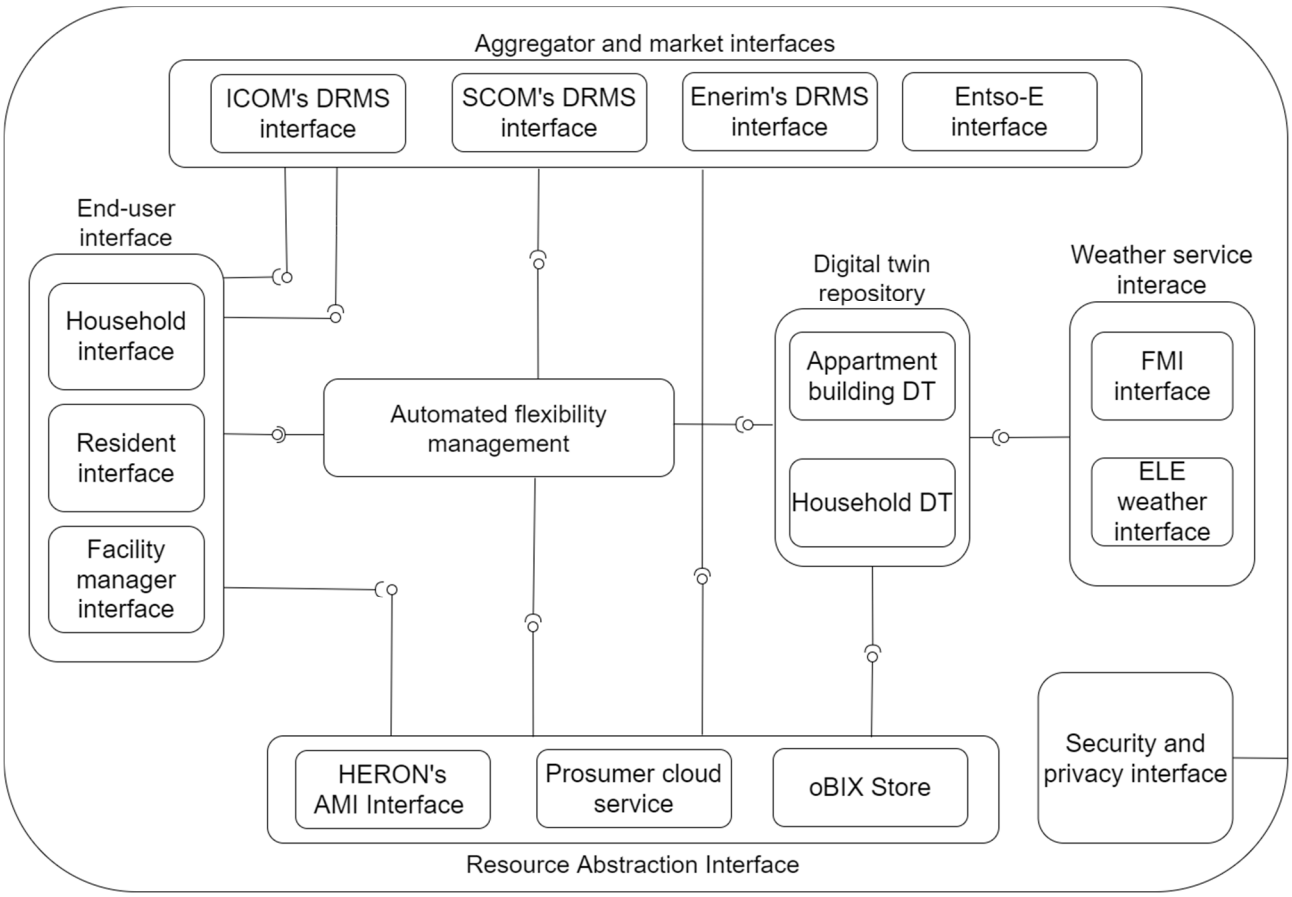

Figure 3: Overview of the iFLEX Framework.

4.1 Aggregator and market interface

The features of the Aggregator and Market interface that are available at the first phase of the project are described in more detail in the dedicated deliverable, namely D4.4 *Initial Market and aggregation interface module*. The main details of each available prototype are provided here.

4.1.1 Demand Response Management System for the Greek pilot

The Demand Response Management System (DRMS) for the Greek pilot is based on the solution provided by ICOM. This solution is able to manage Demand Response (DR) programs, participants and related communications. Apart from supporting the basic operations of scheduling and dispatching DR events, it includes an automated mechanism for optimal selection of DR Programs and Participants' strategies in order to provide reliable and cost-effective flexibility.

Some indicative screenshots of the currently available User Interface (UI) are presented in the following figure (for more details please read D4.4).

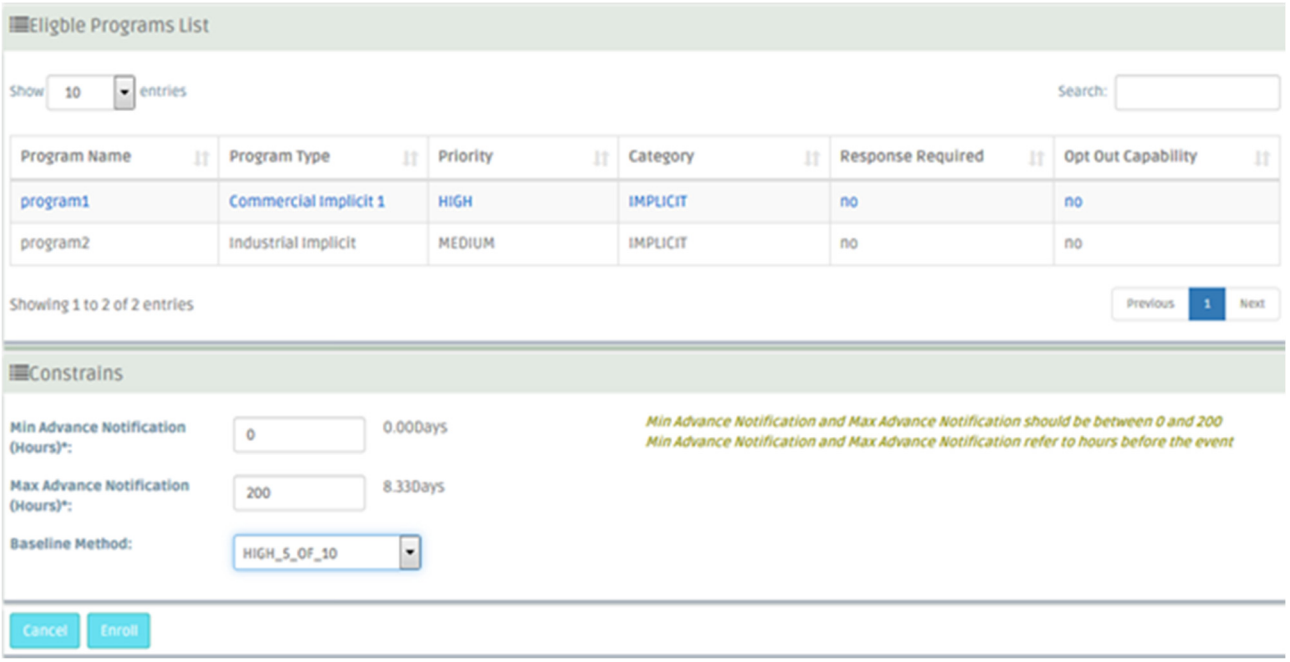

Figure 4: Indicative screenshots of the DRMS available for the Greek pilot at the pre-pilot phase.

4.1.2 Demand Response Management System for the Finnish pilot

The DRMS for the Finnish pilot is based on the solution provided by Enerim. This is a Software as a Service (SaaS) solution that simplifies the complexity of different energy marketplaces, connections and data collection from physical assets, as well as sharing assets or market positions between energy stakeholders. The currently available software, the structure of which is being depicted in the following figure, supports:

- Meter data management
- Consumption and production forecasts
- Balance management (24/7)
- Trading, Elspot, Elbas, and natural gas after market
- Balance Responsible Party (BRP) balance settlement
- Natural gas balance settlement
- Energy invoicing and reporting

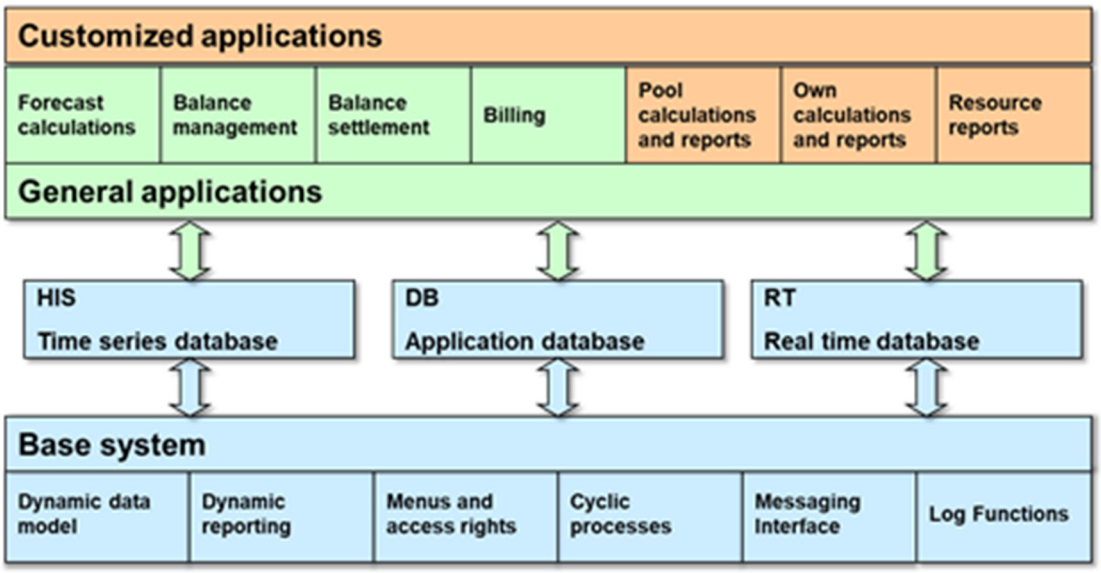

Figure 5: The structure of EnerimEMS SaaS.

4.1.3 Demand Response Management System for the Slovenian pilot

The DRMS for the Slovenian pilot is based on the solution Flexibility Operator Interface (FOI) provided by SCOM. Its primary aim is to facilitate active management of the flexibility (i.e., DR) events. The FOI comprises three groups of components based on the type of client interacting with it. The FOI Admin interface provides Flexibility operators with the means to manage (create, update, delete) participant invitations, household profiles, registrations of home energy gateways for automated energy management, and scheduling (create, browse, update, delete) of the demand response events. The FOI User Page interface provides a registration page for participants. The FOI End-User interface provides the registration API for mobile devices.

4.2 End-user interface

The End-user interface is designed to facilitate consumers and/or prosumers to improve the efficiency and sustainability of their premises or their community premises (building) while participating in energy markets by offering demand and/or production flexibility services. Considering that end users may have no background in energy related terminology and operations, a simple and easily navigable interface is provided through a web or a mobile based interface.

The component enables the information flow between the end user and other iFLEX components. From a functional perspective, the component supports various capabilities towards providing a high-quality user experience for end users. A subset of the functionalities provided through the end-user interface are:

- *Observation of energy production/consumption data:* provides insight on historical and real-time energy related indicators upon end-user request, so that they can have a clear view of their total energy consumption and potential generation, as well as insights into specific assets.
- *Notification of flexibility events:* in cases where the end user chooses the manual instead of the automatic operation for asset management, iFlex Assistant (iFA) suggests schedules for flexible assets. Suggestion may or may not be endorsed by the end user.
- *Visualization of sustainability metrics and goals:* given their participation in previous DR events, the end users have the ability to retrieve and visualize flexibility reports related to their assets.
- *Definition of comfort levels:* end users are capable of configuring utilization of their flexible assets, meaning they can set boundaries in the time frame and asset-level operational constrains.
- *Flexibility policy management:* ability to configure the desired optimization policy for the assets installed in the user's premises. CO2 emissions, energy consumption and energy bill minimization are three non-mutually exclusive user objectives.
- *Residential conditions feedback:* enable end users to provide feedback on building thermal comfort in cases where heating devices are considered as flexible assets and utilized as such.

Availability of the functionalities described above varies per pilot, based on applicability, existing data and resources.

Two distinct solutions are developed to support end-user interaction with the iFA, one applicable to individual end users (Greek and Slovenian pilots) and another to building community users (Finnish Pilot). Those solutions are utilized during phase one of the iFLEX framework implementation:

- The first solution offers a simple and intuitive user experience with the aim to keep the end user engaged. To that end, the user interface is conversational, using first and second person, while keeping the number of screens and interface components at minimum to prevent possible user frustration. Regarding the technology stack, the application is based on React Native[1], a JavaScript Framework for mobile application development, which has the advantage of compiling code down to native iOS/Android code for enhanced performance.
- The second solution is relevant to the building community users, and provides insightful building level data. Electricity and heating consumption along with CO_2 footprint data are available to all users, while each registered user can further observe apartment level thermal comfort data, temperature and humidity.

[1] https://reactnative.dev/

Indicative representation of the current state of the mobile application along with the initial mock-up design is provided below. Main Settings, Automations settings and Automation Schedule screens are illustrated in the Figures 8, 9 and 10.

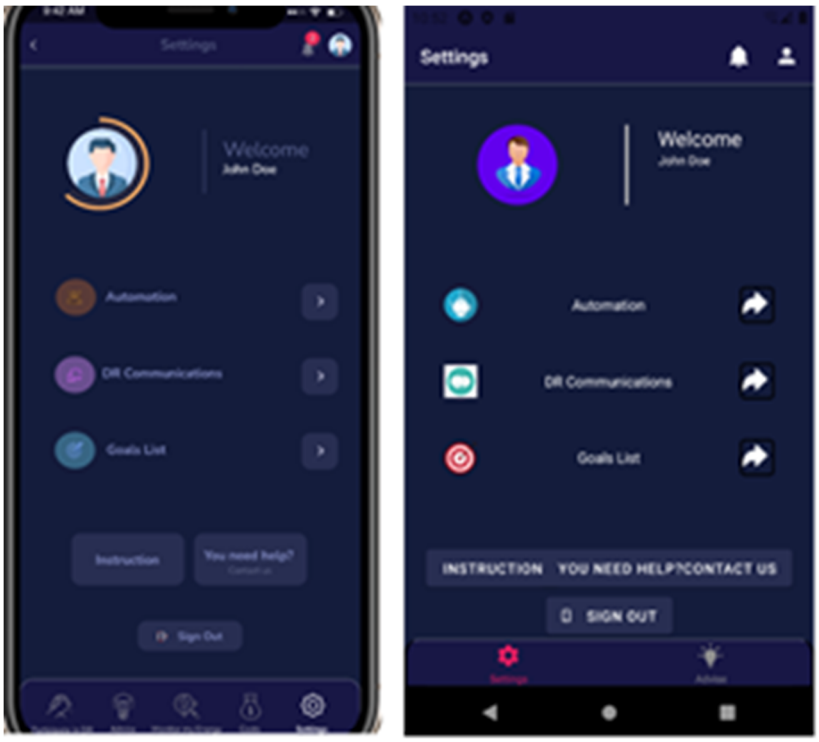

Figure 6: Main Settings Screen.

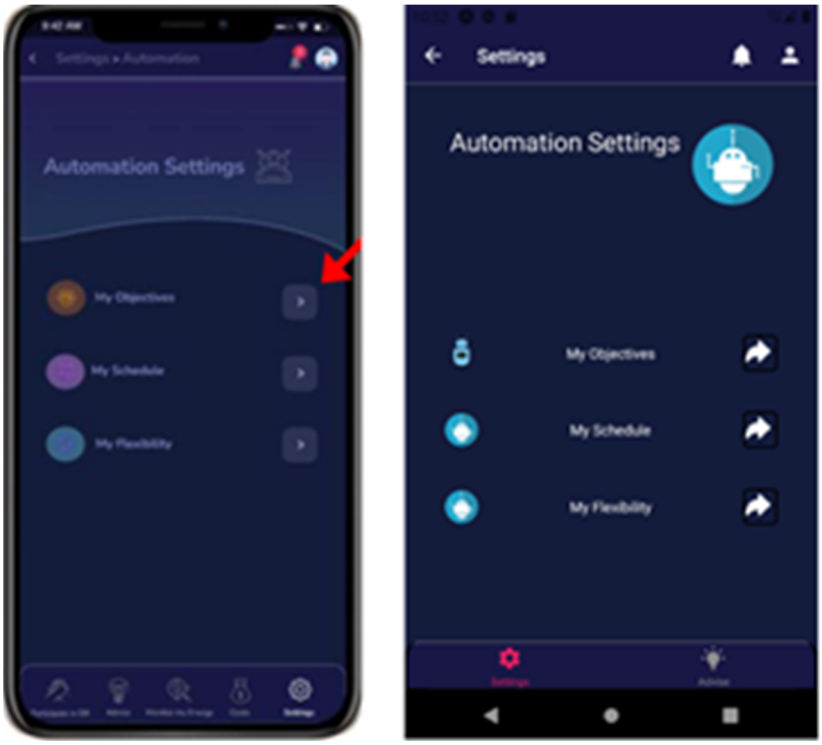

Figure 7: Automation Settings Screen.

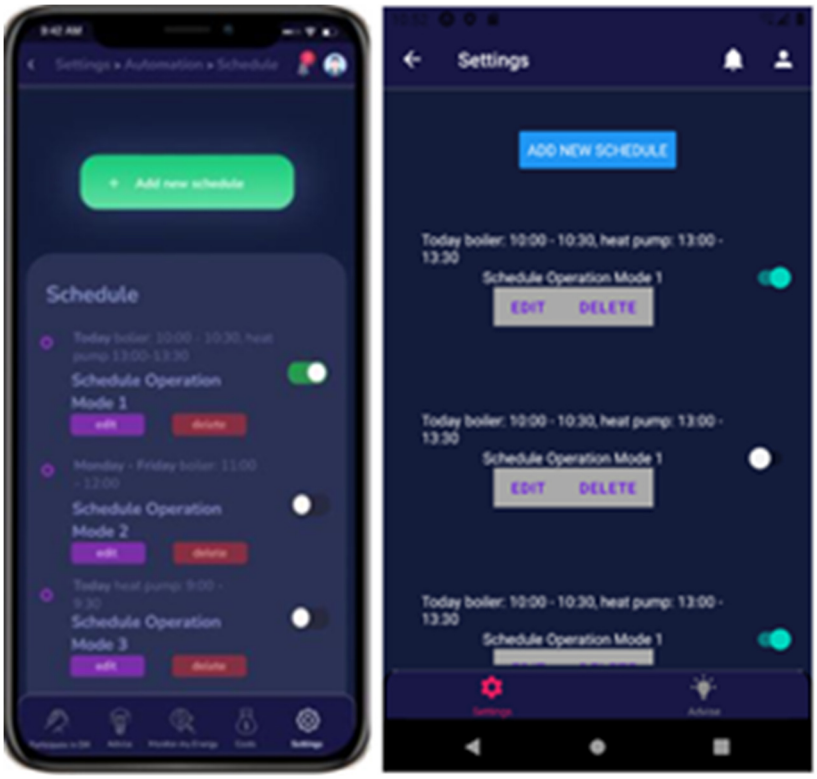

Figure 8: Automation Schedule Screen.

The supported functionalities along with a thorough explanation of the decisions made on the design of the user interface for each of the developed applications are provided in D3.4 *Initial Natural User Interfaces*.

4.3 Resource abstraction interface

The Resource abstraction interface (RAI) component provides interfaces for home energy management systems (HEMS), building energy management systems (BEMS), smart meters, and other external metering/sensor infrastructure. That is, the role of the RAI is to provide iFLEX Assistant with means to monitor and control energy consumption, comfort, and other relevant aspects in the consumer/prosumer premises.

The current RAI component contains three district modules based on existing baseline systems: HERON's AMI interface, Prosumer Cloud Service, and oBIX Store. The oBIX Store module provides means to access data and control resources provided by BEMS via standard Open Building Information Xchange format[2]. It is based on existing baseline implementation hosted by VTT. The oBIX Store is implemented with Java.

The Prosumer Cloud Service (PCS) is a component providing a backend storage service for prosumers. It is capable on one end to collect data from publish-subscribe protocols like MQTT[3] or customize data ingestion systems and on the other to serve the data via REST API to its clients. Used in this way the PCS wraps the diverse resource abstractions and offers them through a unified API. The PCS can be used as a REST API server for backend needs of other components like End User Interface, Automated Flexibility Management, Digital Twin, etc. The PCS comes with integrated security API and framework allowing to implement fain grain security policies governing access to the REST API. The PCS is implemented in Python, Tornado web framework[4] is used as a baseline for REST API, the backend database is MongoDB.[5]

HERON's AMI interface provides means both for real-time monitoring of consumer loads and for remote control of energy assets at the consumer premises. In particular, HERON's AMI provides the following functionality: 1) access to Smart Meter and IoT data (i.e. temperature and occupancy), 2) access to user schedules provided through manual operation, and 3) methods for controlling assets in the end-user premises.

[2] http://docs.oasis-open.org/obix/obix/v1.1/csprd01/obix-v1.1-csprd01.html
[3] See MQTT web page for details: https://mqtt.org/
[4] See Tornado web pages for details: https://www.tornadoweb.org/en/stable/
[5] See MongoDB wep pages for details: https://www.mongodb.com

The first phase implementation of the RAI component is described in more detail in D4.1 *Initial Resource abstraction interface module*.

4.4 Automated flexibility management

The Automated flexibility management (AFM) component provides services for implementing DR control and optimization operations. It achieves this by applying optimization techniques with models from the Digital Twin Repository. More concretely, it provides the following functionality:

- **Baseline forecasting:** AFM component requests forecast from digital twin repository to form a baseline load forecast. Then it publishes the baseline forecast via MQTT.
- **Flexibility estimation:** AFM component predicts minimum and maximum loads utilizing models from digital twin repository together with internal optimization module for different timeslots in order to evaluate the flexibility of the prosumer.
- **DR control and optimization:** AFM component utilizes the digital twin models to predict the demand response for different control options and iterates through them in order to find optimal control strategy. Chosen controls are sent to the Resource Abstraction Interface via MQTT.

The current implementation of the Automated flexibility management module is shown in Figure 9 and consists of:

- **Planning module**: Handles most of the communications to other modules as well as holds most of the operational logic. It formulates the baseline consumption, flexibility estimates and selection of control commands.
- **Resource module:** Acts as a gateway to the models in the Digital Twin Repository. One resource module represents one consumption, generation or flexible asset of the site. In addition, they do have some internal logic as well for translating more abstract command controls sent by the Planning module to more concrete controls sent to the Resource Abstraction Interface. The module is also capable of using optimization module for estimating minimum and maximum loads.
- **Optimization module:** Currently, this module comprises of different heuristic optimization techniques in order to provide rough estimates on flexibility and resource consumption extremities.

The AFM module is written in Python programming language. Data is primarily stored in DataFrame format form Pandas (McKinney, 2010) library. Optimization module, as well as some other data manipulation, is implemented using NumPy (Harris et al., 2020). Communication to other interfaces is done via MQTT protocol, a Python client is implemented by Eclipse Paho project[6].

[6] https://www.eclipse.org/paho/index.php?page=clients/python/index.php

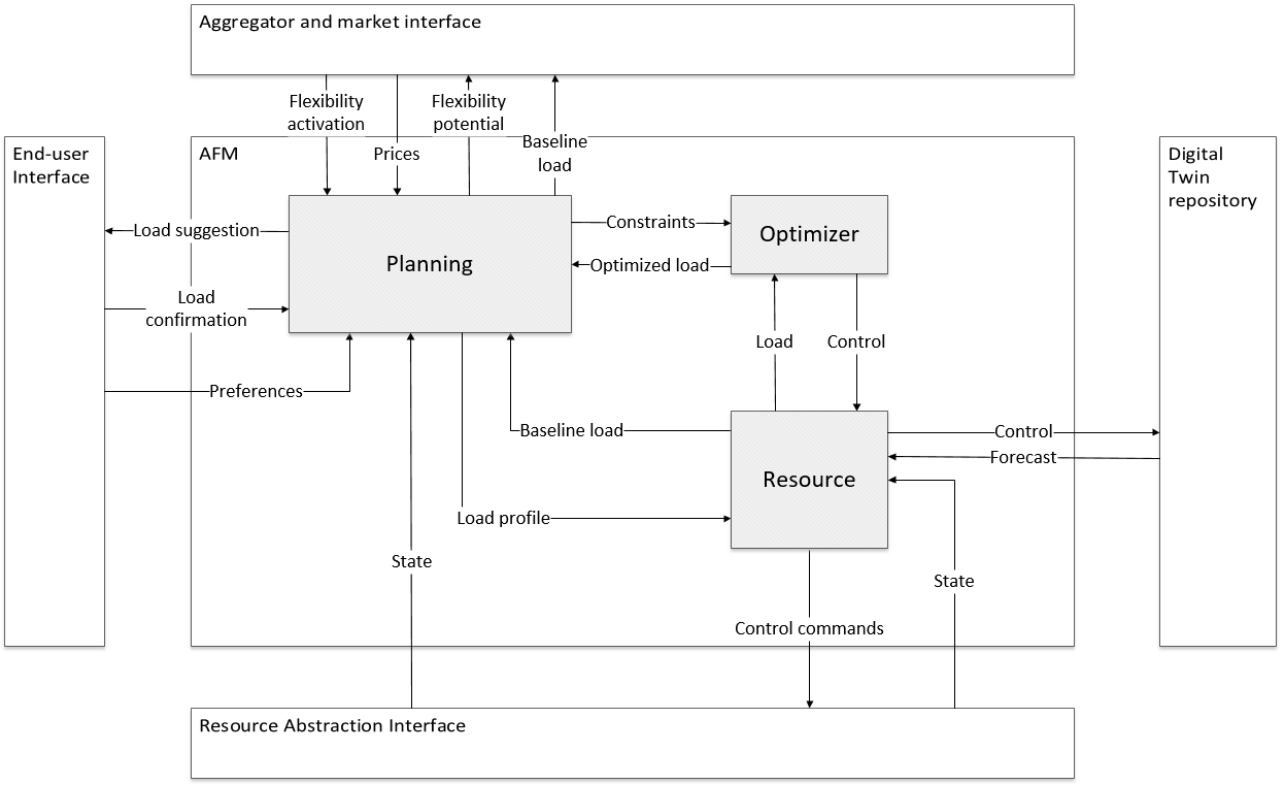

Figure 9: Architecture of the Automated flexibility management module.

The initial version of the AFM component and its architecture is described in more detail in *D3.7 Initial Automated flexibility management module*.

4.5 Digital twin repository

The main purpose of the Digital twin repository is to:

- Estimate the impact of a consumer (i.e., metering point) for flexibility management.
- Provide optimal and consumer-centric flexibility management with model-based planning and control.

To this end, the Digital twin repository module consists of models for predicting and simulating consumer loads, flexibility and potential response to flexibility signals. These models are utilized by the Automated flexibility management module to estimate the baseline, flexibility and response of the consumers. The models are implemented with combination of machine learning and physics-based modelling. The current version of the repository includes digital twins for apartments and houses.

The Digital Twin (DT) for apartment buildings consists of three models as illustrated in Figure 10. It contains two machine learning models for predicting space heating power and base load of the whole apartment building, and a simple physics-based model for estimating the time constant of a building's heating system. The current version of the DT utilizes artificial neural networks for forecasting the baseloads for electricity and district heating. Thermal flexibility models include a room temperature model based on Newton's law of cooling and simple energy consumption models for space heating provided by heat pumps and district heating.

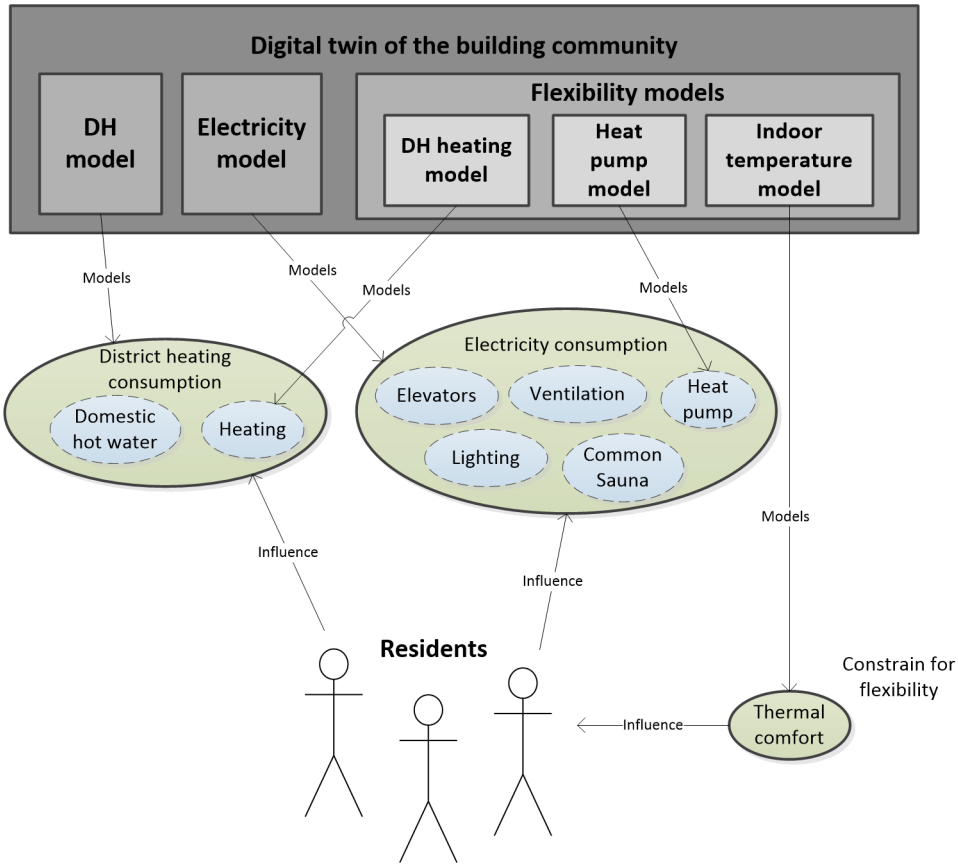

Figure 10: Conceptual representation of the interdependencies among the Digital Twin, measured parameters (green), non-measured parameters (blue) and the residents of the building community.

The digital twin of a house, illustrated in Figure 11, is based on four models: the thermal house model, the electrical model, the flexibility model, and the mobility model. The first basis of the model is explained and presented. The data used for modelling is displayed. This model consists partly of a neural network and partly of a gradient enhancement solution.

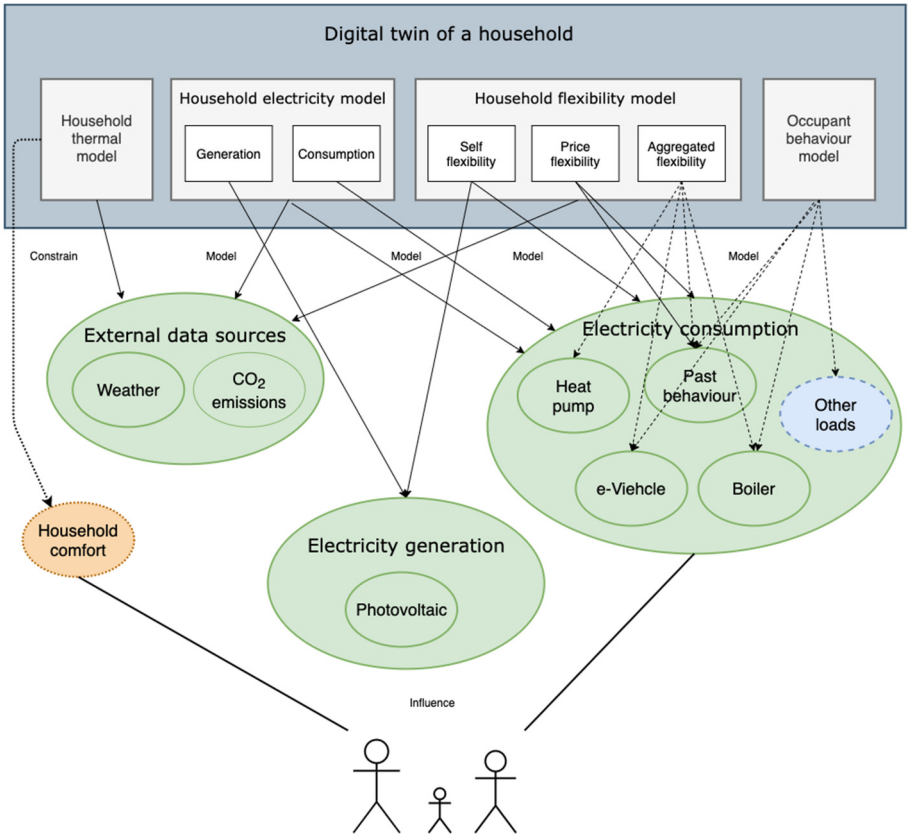

Figure 11: Household modelling, models in grey, measured parameters in green, constrains in orange, non-measured parameters in blue. Full arrows represent phase one focus.

The digital twin repository module is implemented with Python programming language. All ANN models in the repository are implemented with Tensorflow 2.0 (Abadi et al., 2016) on top of its Keras API (Chollet, 2018). Physics-based models are mainly implemented in NumPy (Harris et al., 2020). The predictive data analysis tool scikit-learn (Pedregosa et al., 2011) is used to determine some parameters of physical models and data aggregation. Pandas (McKinney, 2010) is used for analysis and manipulation of time series.

The initial version of the Digital twin repository module is presented in more detail in D3.1 *Initial Hybrid-modelling module*.

4.6 Weather service interface

As the name implies, the role of the Weather service interface module is to provide data about weather forecasts to the iFLEX Framework. In the first phase, this module will be utilized inside the iFLEX Assistants to provide the digital twin repository component with weather forecast data, which is required for predicting the response of weather-related consumption units such as the building's heating and cooling system.

Providers of the weather forecast services are typically country specific. Because of this, the weather service interface module consists of individual modules for different countries. In phase one, the module contains interfaces for accessing weather forecasts provided by the Finnish Meteorological Institute (FMI). Additionally, interfaces for accessing weather forecasts in Slovenia are under development. The FMI provides an open API to access Finland's weather forecast 48 hours in advance. The data is shared according to the standards of the Open Geospatial Alliance[7]. The module providing access to weather forecast data in Finland is integrated to the RAI component via a common database. Figure 12 illustrates the setup for integrating FMI data into the iFLEX Framework. The interface for fetching data into the RAI database is implemented in Java programming language. A client for accessing the weather forecast data via the oBIX RESTful API is implemented in Python to enable easy integration with the Digital twin repository module.

[7] https://www.ogc.org/standards

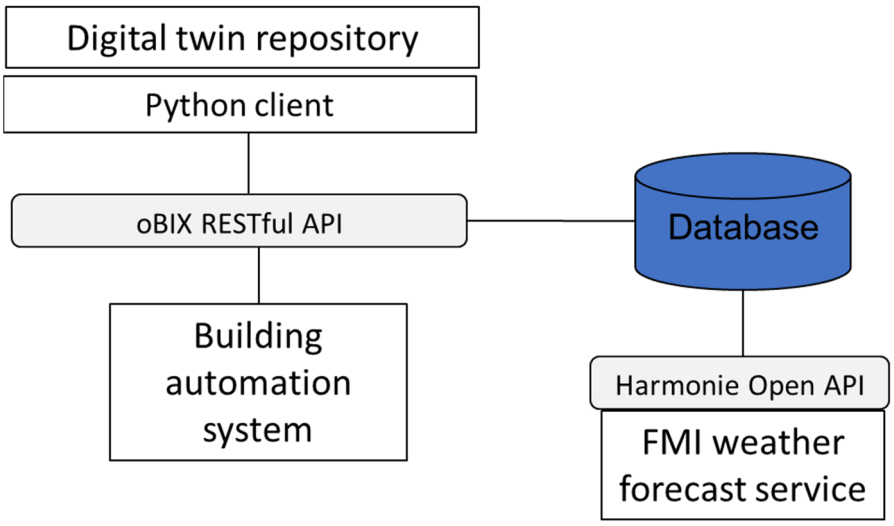

Figure 12: Weather forecast interface module integrated into the Resource abstraction interface component.

In Slovenia, the weather data is integrated as a combination of weather data ingestion service, feeding the Prosumer Cloud Service, and the REST API providing access to forecasted and realized weather data. The ingestion service is implemented in Python, current implementation is further described in deliverable D4.1*Initial Resource abstraction interface module*. The weather data provides information on temperature, radiation and precipitation on 1h basis. The forecast is available for 7 days in advance for 1h granularity. The weather data originates from Slovenian Environment Agency (ARSO)[8] and is provided by ELE.

4.7 Security and privacy interface

The Security and privacy interface provides a number of services and mechanisms needed for implementing the security and privacy requirements as defined in the deliverable D2.1. The following services are currently implemented:

- Trust management module: the module provides a Certification Agency for issuing of digital certificates for services, service managers and end users. According to the application security policies, the certification can be issued according to the role the entity (either a service or user) has in the system. Multiple trust management setups could be prepared, for example one per pilot. The module is based on OpenSSL[9] and implemented with a build automation software Make[10] and parts in a shellscript. An additional component is a component that allows issuing dynamic certificates based of authenticated and authorized access from other system components, like a privacy-compliant, end-user engagement service.
- Privacy-compliant, end-user engagement service: the service provides a web portal for end-user registration and personal account management. During the registration, the certificates for the end user can be dynamically generated and provided. The service allows for utilizing of various invitation methods for registration, like magic numbers[11] provided in the electricity bill, etc. The service provides means to register an additional device under the same account with same privileges in a single trust management setup. The feature could be used for providing access control certificates for mobile devices to be able to access Prosumer Cloud Service REST API
- Access control service: the Access Control Service (ACS) uses the digital certificates issued by the trust management module and internal security policies to provide access control decisions regarding access of system entities to the Prosumer Cloud Service REST API or access to messaging service messages, for example MQTT. The service is performant enough to support large number of requests to the REST API or messaging communication channels. The policies can be fain grain and can control access to REST API HTTP methods (GET, PUT, PULL, etc.) or read and/or write access to

[8] See Slovenian Environmental Agency home page for more details: https://www.arso.gov.si/en/
[9] See OpenSSL web page for details: https://www.openssl.org/
[10] See Gnu Make Wikipage for details: https://en.wikipedia.org/wiki/Make_(software)
[11] https://en.wikipedia.org/wiki/Magic_number_(programming)

MQTT topics. The policies access to resources can be either role or identity based. Through specific procedures, the policies could be updated to manage dynamically issued digital certificates.

- Communication security: all the communication among the security components, system components and resources should be secured. In the baseline implementation, the communication was in general protected by Transport Layer Security -TLS protocol.[12] The TLS protocol provides data origin integrity and authenticity as well confidentiality service of information exchanged in the communication.[13] The baseline approach is used in communication with the Prosumer Cloud Service and presented security components. The challenge of secure communication with the Resource abstraction interface and beyond, towards the devices and actuators and the End-user interface still needs to be worked on and further integrated in the final system setup.

[12] See TLS Wikipages for details: https://en.wikipedia.org/wiki/Transport_Layer_Security
[13] All security terminology used in the document is aligned with the RFC4949 - Internet Security Glossary, Version 2, see the glossary online: https://datatracker.ietf.org/doc/html/rfc4949

5 Conclusion

This is a demonstrator type deliverable that documents the first phase implementation of the iFLEX Framework. The iFLEX Framework presented in this deliverable consist of the components developed in the technical work packages of the project. The components of the framework and their main responsibilities include:

- **Aggregator and market interface**: enables the interaction between the iFLEX Assistant and the market.
- **End-user interface**: enables the consumers/prosumers to interact with the iFLEX Assistant.
- **Resource abstraction interface**: provides iFLEX Assistant with means to monitor and control energy consumption, comfort, and other relevant aspects in the consumer/prosumer premises.
- **Automated flexibility management**: provides services for implementing DR control and optimization operations.
- **Digital twin repository**: estimates the impact of a consumer for flexibility management and provides optimal and consumer-centric flexibility management.
- **Weather service**: provides country-specific data about weather forecasts.
- **Security and privacy interface**: provides trust and security among iFLEX subsystem and services, users and data.

This document provides a high-level description of the whole iFLEX Framework. More details for each functional component of the framework are provided in component specific deliverables referenced throughout this deliverable.

6 List of figures and tables

6.1 Figures

7 References

(Chollet, 2018) Chollet, F. (2018). Deep Learning with Python and Keras. MITP-Verlags GmbH & Co. KG.

(EC, 2007) European Commission (2007). A lead market initiative for Europe. Brussels. COM(2007) 860 final.

(Harris *et al.*, 2020) Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*. https://doi.org/10.1038/s41586-020-2649-2

(McKinney, 2010) McKinney, W. (2010). Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference.

(Pedregosa *et al.*, 2011) Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research.