D4.2 - Updated User-side ESS control system

Deliverable ID	**D4.2**
Deliverable Title	**Updated User-side ESS control system**
Work Package	**WP4**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**14/06/2018**
Status	**final**
Type	**Prototype**
Lead Editor	**UNINOVA**
Main Contributors	**UNINOVA (João Martins, Mário Figueiredo, Pedro Pereira, Vasco Delgado-Gomes), FRAUNHOFER FIT (Gustavo Aragón)**

Published by the Storage4Grid Consortium

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731155.

Document History

Version	Date	Author(s)	Description
0.1	2017-12-29	UNINOVA	First draft.
0.2	2018-05-28	UNINOVA	Updated prototype picture and functionalities.
0.3	2018-06-05	FRAUNHOFER	PROFESS modules and API description.
0.4	2018-06-07	UNINOVA	Editing and revision.
0.5	2018-06-14	UNINOVA	Reviewers' comments addressed.
1.0	2018-06-14	UNINOVA	Final version, ready for submission to the EC.

Internal Review History

Review Date	Reviewer	Summary of Comments
2018-06-12 (v0.4)	Gitte Wad Thybo (ENIIG)	Approved: • General minor corrections • Missing acronyms in the list
2018-06-13 (v0.4)	Giovanni Paolucci (ALPERIA)	Approved: • General minor corrections

Table of Contents

Executive Summary

D4.2 – "Updated User-side ESS control system" presents the updated developed prototype implementing some features of the User-side ESS control system and its architecture. For a better understanding the LESSAg is now named Professional Realtime Optimization Framework for Energy Storage Systems (PROFESS)

The architecture of User-side ESS control systems is composed by the north-bound and south-bound connectors, connect through the SMX-EB. The PROFESS component receives and sends all the necessary information through the SMX-EB, interacting with the field-related devices, according to its algorithms decisions.

The PROFESS is a software component running site-wise ESS control algorithms. It receives in quasi-real-time all available information from local devices (ESS systems, PV energy meters, load-side energy meters, EV chargers, etc.). In S4G, PROFESS works also together with GESSCon, a global ESS controller service, linking the charging/discharging profiles for the next 24h sent by GESSCon with the internal optimal control model of PROFESS. In some applications (e.g. related to distributed energy management) it can be configured to act as the main local EMS.

In the following deliverable (D4.3 – "Final User-side ESS control system"), the final algorithms will be implemented in the PROFESS. These algorithms will implement cooperative charging strategies, and they will also focus on minimizing energy cost and optimizing self-consumption.

1 Introduction

D4.2 describes the "Updated User-side ESS control system" prototype, developed by the Storage4Grid project. The control system will be shared by two intelligent devices: the SMX of the USM, and the intelligence of the ER, or, in more traditional cases where the ER is not available, the ESS itself. These components share the ESS control according to the local ESS and RES resources: individual local ESS, hybrid inverter system, and ER system.

The interaction between the PROFESS and the cloud components is detailed in this deliverable. The PROFESS receives quasi-real-time data from the cloud components, namely GESSCon and DSF. More detailed information can be retrieved from related documents summarized in Section 1.2.

1.1 Scope

This prototype deliverable has been developed by Task T4.1 – "User-side ESS control". One last update of this prototype is expected to be released as D4.3 – "Final User-side ESS control system" (M30).

1.2 Related documents

ID	Title	Reference	Version	Date
[D2.1]	Initial Storage Scenarios and Use Cases	D2.1	1.1	2017-06-08
[D2.5]	Initial Lessons Learned and Requirements Report	D2.5	1.0	2017-05-30
[D3.1]	Initial S4G Components, Interfaces and Architecture Specification	D3.1	1.0	2017-09-15
[D4.8]	Initial USM Extensions for Storage Systems	D4.8	1.0	2017-08-31
[SmxGuide]	SMX Make yourself guide	SmxGuide	1.6	2018-04-07

2 Initial User-side ESS Control System Prototype Overview

The overall high-level structure of D4.2 - "Updated User-side ESS Control System" prototype is shown in Figure 1.

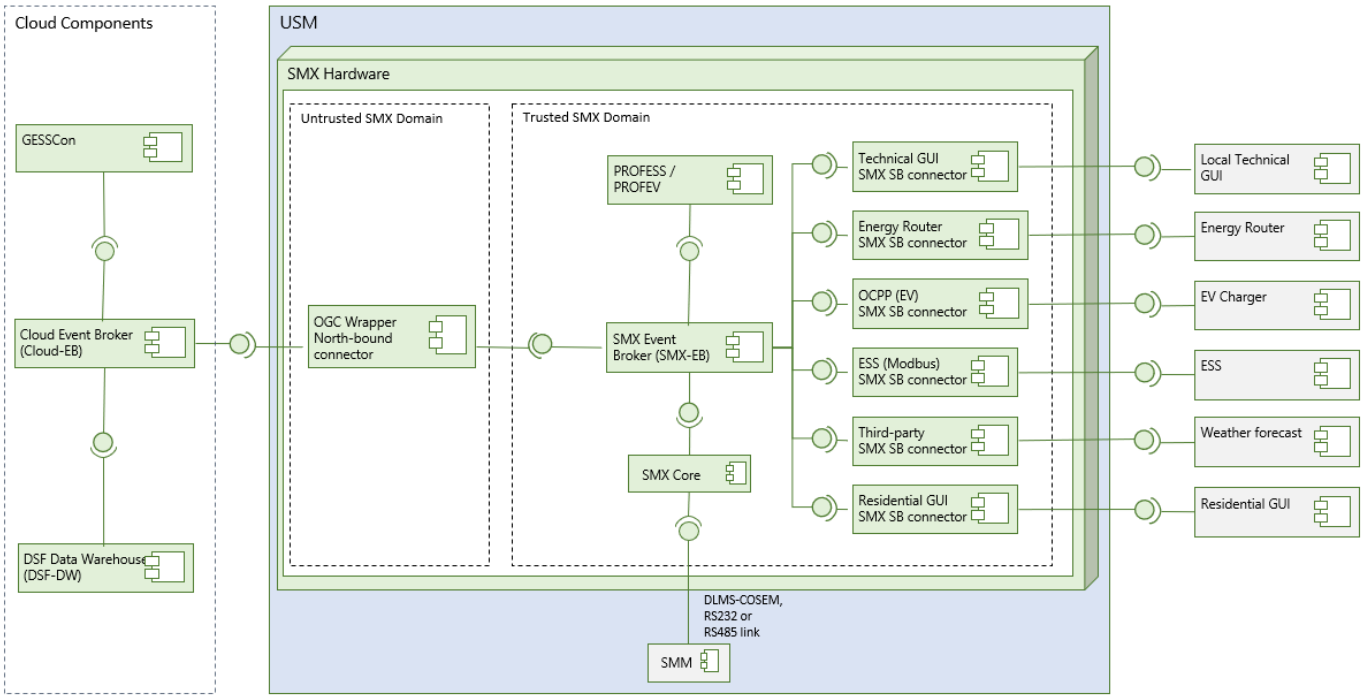

Figure 1. Prototype diagram.

2.1 PROFESS

The PROFESS is a software component running site-wise ESS control algorithms. It receives in quasi-real-time all available information from local devices (ESS systems, PV energy meters, load-side energy meters, EV chargers, etc.). In S4G, PROFESS works also together with GESSCon, a global ESS controller service, linking the charging/discharging profiles for the next 24h sent by GESSCon with the internal optimal control model of PROFESS. In some applications (e.g. related to distributed energy management) it can be configured to act as the main local EMS. The PROFESS architecture is show in Figure 2.

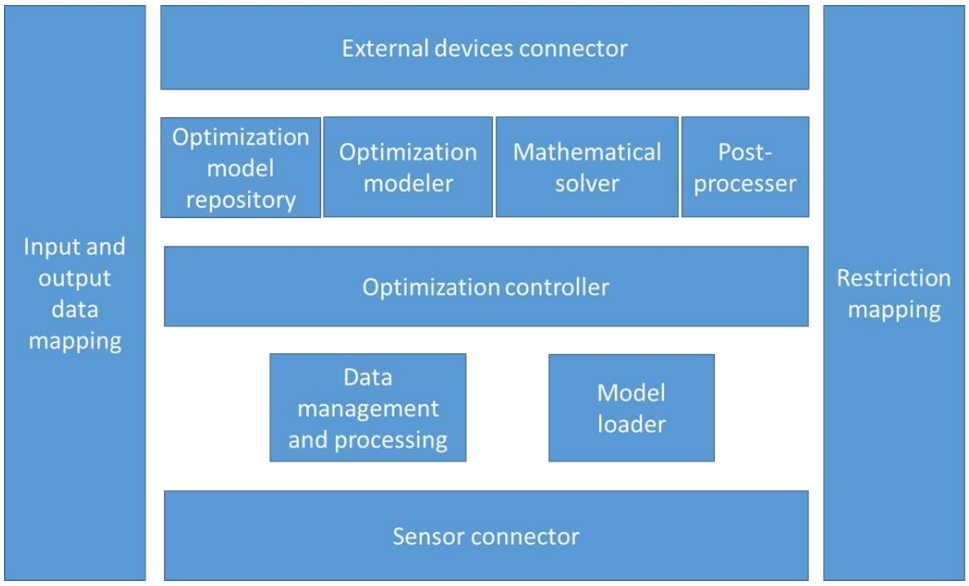

Figure 2. PROFESS architecture.

The following subsections describe in more detail, each of the PROFESS's modules.

2.1.1 Sensor connector

The *sensor connector* (SC) connect sensors delivering the input information necessary for the optimal control. In S4G, the sensor connector receives the information of the different smart meters sent by the SMX through the SMX-EB broker. The SC allows also the use of cloud services in the optimization. In this case, the SC has connectors to these services reading continuously their inputs. An example are weather forecast services, which are used for the prediction of PV generation.

2.1.2 External devices connector

The *external device connector* (EDC) prepares and links the optimal control results to their corresponding output channel. These results can be obtained through a RESTful API or MQTT.

2.1.3 Data management and processing and model loader

The inputs obtained from the SC are used consequently by the *data management and processing* module (DMP) to generate some predictions. The DMP manages the input data from sensors, pre-process it if necessary, links an implemented algorithm through the *model loader* module (ML), evaluates it and executes a post-processing of the resulted prediction.
PROFESS uses the complex event processing and machine learning framework (CEML) in charge of pre-processing, learning, evaluating and deploying data from machine learning algorithms[i,ii,iii]. CEML connects to algorithm implementations using the ML. The ML contains the implementation of machine learning algorithms.

2.1.4 Input and output data mapping

The *input and output data mapping* (IODM) maps inputs or outputs channels to the corresponding information context. In this way, PROFESS knows e.g. if an input corresponds to consumption measurements and can map it internally to the corresponding load profile prediction. In the same way, PROFESS can map the result of an optimization process to a certain output that will be used for some energy element control, e.g. charging/discharging of an ESS.

2.1.5 Optimization controller

The optimization controller (OC) orchestrates the inputs from the predictions, charges the corresponding optimization model and links all this information to a mathematical solver. Besides, the OC controls the frequency in which each optimal control is calculated and manages solver error events.

2.1.6 Restriction mapping

The *restriction mapping* module enables the input of different objective functions or constraints to be mapped to the optimization model.

2.1.7 Optimization modeller

The *optimization modeller (OM)* applies optimization modelling tools to define mathematically the optimization problem and to link a *mathematical solver (MS)* or algorithm in charge of the calculation of the modelled problem. Through the available documentation of the tools, the use of different optimization models is simplified for the user.

2.1.8 Mathematical solver

The mathematical solver (MS) calculates the optimization problem modelled through the optimization modeller. The MS searches for an optimal solution and reports errors if the solution is not found. Different open source solvers can be used together with the PROFESS, allowing the calculation of a variety of optimization problems from linear, mixed-integer to non-linear problems. In Professional Realtime Optimization Framework for Electric Vehicles (PROFEV) another version of PROFESS for electric vehicles, stochastic optimization problems are also calculated using stochastic solvers or via dynamic programming.

2.1.9 Post-processor

This module generates and manages event-based messages, which will be exchanged among distributed devices, e.g. in a more complex cooperative energy management scenario.

2.2 Optimization algorithms

Mathematical optimization is a very useful tool for decision making in a wide range of application fields such as operation research, design, engineering, as well as energy management. Main idea of optimization is minimization (maximization) of a cost (benefit) function expressed in terms of control variables and parameters that determine the system to be optimized. This function is usually referred to as objective function. Furthermore, in many optimization problems, there are some limitations for the values that the optimization variables can take. These limitations are expressed with constraint functions, which are also combination of parameters and variables.

Optimization problems are classified in three different categories: function types, variable domains, parameter certainty. An optimization problem is classified as linear problem if all of constraint and objective functions are linear. Similarly, optimization problems can be quadratic or non-linear. Control variables could be continuous or discrete. Problems are classified as integer or binary problems according to the specific type of discreteness. Furthermore, the problems that include both continuous and discrete variables are referred to as mixed problems e.g. mixed integer problems. Finally, some system parameters are deterministic while some parameters include randomness in nature. The problems that have random parameter are called as stochastic optimization problems.

An example of the use of the optimal control by PROFESS is explained in Figure 3, where a house is equipped with PV and an Energy Storage System (ESS). The goal is to achieve an optimal control of the ESS using an

optimization model. The optimization goal is defined mathematically in the model through an objective function, as for example the maximization of the self-consumption. It means, that the control algorithm will try to optimize the charging and discharging of the ESS, in order to maximize internally the consumption of the energy generated by the PV. The model also requires some restrictions expressed mathematically. One example constitutes the State of Charge (SoC) of the battery that cannot go under 20% (longer lifetime) and up 100%. Information about the system is inserted into parameters of the model. Some of these parameters are predictions required for the optimization. In this example, the required predictions are for PV generation and load profile for the next day. Finally, the model, requires some variables to be controlled in the optimization process. These variables will contain the output information for the ESS about its charging or discharging control.

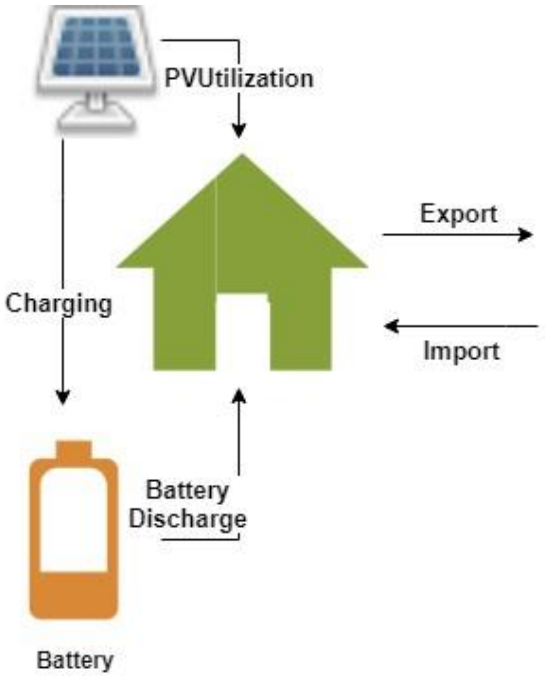

Figure 3. Residential scenario

3 Installation/Deployment instructions

To prepare a SMXCore image it is necessary to follow the instructions document available to the consortium [SmxGuide]. PROFESS uses a container architecture which simplifies its deployment. Images of the software are uploaded into an open space of Docker Hub. The user has to download the images and make them run into the final platform (SMX). Any change of the settings can be achieved through the PROFESS API.
Another possibility for the deployment is by using the open source Linksmart repository[iv,v]. The code can be cloned from there. In that case, developers can change the code if necessary, then build it and run into the platform.

4 Software dependencies and requirements

No special hardware requirements are needed to implement the Updated version of the User-side ESS control. This D4.2 prototype has the software dependencies described in Table 1.

Table 1. Software dependencies.

Dependency	License	Role
Docker and docker-compose for Raspbian	Apache License 2.0	Docker is used to facilitate the PROFESS installation.
Raspbian, version "Jessie with Desktop"	GNU General Public License (GPL) 2.0	The operating system of the Raspberry PI, used as SMCore and ER controller operating system.

5 API Reference

5.1 PROFESS

As shown in Figure 4, the PROFESS allows different operations to enable the control of the field-related ESS devices. Moreover, it allows the operation of the algorithms, and to insert and remove data.

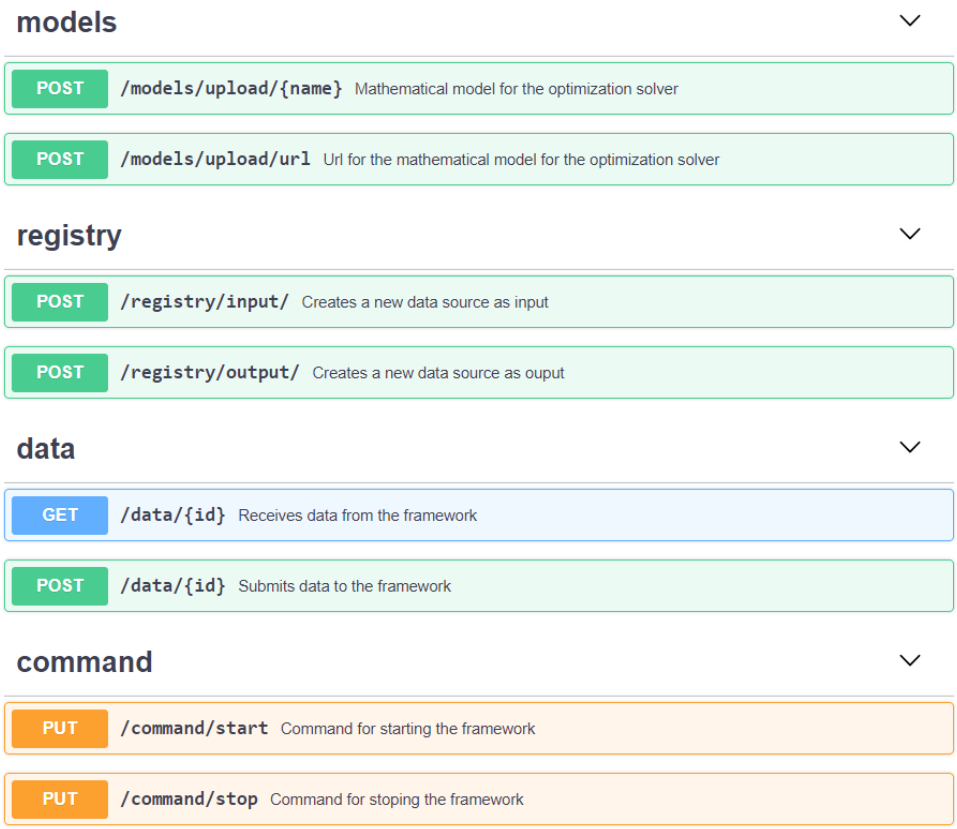

Figure 4. PROFESS API.

6 Conclusions

This document presents all the necessary information regarding the D4.2 - "Updated User-side ESS control system" prototype, developed by the Storage4Grid project. The final version of this document will be available in M30. The current developed modules will be extended and updated in order to provide more features, necessary for the Storage4Grid project.

Acronyms

Acronym	Explanation
API	Application Program Interface
CEML	Complex Event Processing and Machine Learning Framework
DMP	Data Management and Processing Module
DSF	Decision Support Framework
EB	Event Broker
EDC	External Device Connector
EMS	Energy Management System
ER	Energy Router
ESS	Energy Storage System
GESSCon	Grid ESS Controller
IODM	Input Data Mapping
LESSAg	Local ESS Agent
ML	Model Loader Module
MQTT	Message Queuing Telemetry Transport
MS	Mathematical Solver
OC	Optimization Controller
PROFESS	Professional Realtime Optimization Framework for Energy Storage Systems
PROFEV	Professional Realtime Optimization Framework for Electric Vehicles
RES	Renewable Energy Sources
RESTful	Representational state transfer
S4G	Storage4Grid
SC	Sensor Connector
SMM	Smart Metrology Meter
SMX	Smart Meter eXtension
SMX-EB	SMX- Event Broker
SoC	State of Charge
USM	Unbundled Smart Meter

List of figures

List of tables

References

[i] D. Bonino, M. T. D. Alizo, A. Alapetite, T. Gilbert, M. Axling, H. Udsen, J. A. C. Soto, and M. Spirito, "*ALMANAC: Internet of things for smart cities*," Proc. - 2015 Int. Conf. Futur. Internet Things Cloud, FiCloud 2015 2015 Int. Conf. Open Big Data, OBD 2015, pp. 309–316, 2015.
[ii] J. Á. Carvajal Soto, M. Jentsch, D. Preuveneers, and E. Ilie-Zudor., "*CEML: Mixing and moving complex event processing and machine learning to the edge of the network for IoT applications*," IoT 2016, pp. 103–110, 2016.
[iii] D. G. and E. I.-Z. J. A. Carvajal Soto, F. Tavakolizadeh, "*An online machine learning framework for early detection of product failures in the Industry 4.0*," Int. J. Prod. Res., vol. CRC Press, no. Submitted, 2018.
[iv] PROFESS Repository. [Online]. Accessed 29-May-2018, https://code.linksmart.eu/scm/spf/optimization-framework.git.
[v] LinkSmart® Sensor Platform, Optimization Framework. [Online]. Accessed 29-May-2018, https://code.linksmart.eu/projects/SPF/repos/optimization-framework/browse