D4.10 - Final USM Extensions for Storage Systems

Deliverable ID	**D4.10**
Deliverable Title	**Final USM Extensions for Storage Systems**
Work Package	**WP4**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**30/09/2019**
Status	**final version**
Type	**Prototype**
Lead Editor	**UPB**
Main Contributors	**UPB (Mihai Sanduleac, Marta Sturzeanu, Mihaela Albu), LINKS (Orlando Tovar, Michele Ligios) Libal (Yini Xu), UNINOVA (Vasco Delgado-Gomes), Alperia (Karen Stocker)**

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	2019-06-05	UPB	UPB inputs
0.2	2019-07-10 2019-08-26 2019-07-29	UNINOVA LINKS LIBAL	UNINOVA inputs (ER and Fronius extensions) LINKS inputs LIBAL inputs
0.3	2019-08-29	UPB	Aggregation of all partners contributions
0.3	2019-09-02	UNINOVA	Comments addressed
0.4	2019-09-03	LINKS	Comments addressed
0.5	2019-09-10	UPB	Aggregation of all partners contributions
0.9	2019-09-13	UNINOVA	Reviewed version by UNINOVA
0.92	2019-09-16	LINKS	Reviewed version by LINKS
1.0	2019-08-17	UPB	Final version, ready for submission to the EC
1.0	2019-09-23	UNINOVA	Second reviewed version by UNINOVA
1.0	2019-09-25	UPB	Comments addressed, last minutes correlations and inputs required by UPB
1.0	2019-09-27	UNINOVA	Third reviewed version by UNINOVA
1.0	2019-09-27	LINKS	Second reviewed version by LINKS
1.0	2019-09-30	UNINOVA	Fourth reviewed version by UNINOVA
1.0	2019-09-30	UPB	Second final version, ready for submission to the EC

Internal Review History

Review Date	Reviewer	Summary of Comments
2019-09-13 (v0.5)	Vasco Delgado-Gomes (UNINOVA)	Reviewed (1st): • Deliverable does not match S4G template • Executive summary must be extended. • The acronyms need to be extended in the first appearance
2019-09-12 (v0.91)	Hamidreza Mirtaheri (LINKS)	Reviewed: • Some comments and suggestions
2019-09-16 (v0.92)	Hamidreza Mirtaheri (LINKS)	Approved: • The comments are correctly addressed
2019-09-23 (v1.0)	Vasco Delgado-Gomes (UNINOVA)	Reviewed (2nd): • Comments for deliverable improvement. • Sections numbering and cross-references need to be corrected.
2019-09-30 (v1.0)	Vasco Delgado-Gomes (UNINOVA)	Approved: • Minor comments.

Table of Contents

Executive Summary

D4.10 describes the "Final USM Extensions for Storage Systems" details of the prototypes, developed by the Storage4Grid (S4G) project. The deliverable presents the extension connectors which were developed in the Unbundled Smart Meter (USM) to support the S4G functionalities. The extensions address the interconnection with the GUI for technical users, the integration with the Grid-side ESS in HLUC-1, the integration of electric vehicles in HLUC-2, the integration of ER in HLUC-1 and HLUC-3 and the integration of Hybrid inverter for HLUC-2 and HLUC-3, while instrumentation values for submetering with USM have been also adapted to cope with the real meters and with the needs of the project applications.

Similarly, to other prototypes, this document provides minimal technical documentation necessary to understand the functionalities and structure of the USM prototype with its extension connectors needed to support the S4G system at the field level. The document is an update of D4.9 [S4G-D4.9].

This prototype deliverable has been developed by Task T4.4 – "Unbundled Smart Meter Extensions for Storage Systems".

The document is structured as follows: overview of USM and SMX is presented in chapters 2 and 3, the communication and exchange of information is described in chapter 4, while components are presented in chapter 5. Chapter 6 presents the instructions for installation and deployment, then the following two chapters show software dependencies and Application Programming Interface (API) reference. Conclusions are presented in chapter 9.

1 Introduction

D4.1010 describes the "Final USM Extensions for Storage Systems" prototypes, developed by the Storage4Grid project. Similarly, to other prototypes, this document provides minimal technical documentation necessary to understand the functionalities, structure and deployment instructions for the prototype of interest. More detailed information can be retrieved from related documents summarized in Section 1.3.

1.1 Reading Guide

In order to help readers in quickly finding their extensions concerns of interest in this document, Table 1 provides an overview of the key topics addressed by this and associated documents.

Table 1. S4G Architecture Reading Guide

Topic	Relevant Section
General information regarding the Unbundled Smart Meter, billing, instrumentation values and sub metering with different components	Section 2 - Unbundled Smart Meter Architecture. Prototype Overview
Presentation of main extensions used between the USM and other components	Section 3 – Smart Meter eXtensions – SMX. Prototype Overview
The way in which SMX communicates with high-level application	Section 4 – SMX communication and information exchange with high-level application. Prototype Overview
Sub-components of the prototypes related to USM	Section 5 – Sub-components
How the Technical Graphical Interface, Energy Router, Electrical Vehicle and Energy Storage Systems Extensions behaved	Section 6 – Installation / Deployment instructions
Software dependencies and requirements	Section 7 – Software dependencies and requirements
Application Programming Interface References	Section 8 – API Reference

1.2 Scope

This prototype deliverable has been developed by Task T4.4 – "Unbundled Smart Meter Extensions for Storage Systems".

1.3 Related documents

Table 2. Related documents

ID	Title	Reference	Version	Date
D3.3	Final S4G Components, Interfaces and Architecture Specification	[S4G-D3.3]	0.10	2018-09-24
D4.3	Final User-side ESS control system	[S4G-D4.3]	1.0	2019-06-13
D5.5	Final USM Extensions for Storage Systems	[S4G-D5.5]	1.0	2019-09-10
D4.9	Updated USM Extensions for Storage Systems	[S4G-D4.9]	1.0	2018-08-31
D4.11	Initial Energy Router	[S4G-D4.11]	1.0	2018-11-02

ID	Title	Reference	Version	Date
D4.7	Final Cooperative EV charging station control algorithms	[S4G-D4.7]	1.0	2019-08-28
D4.12	Final Energy Router	[S4G-D4.12]	0.2	2019-09-12
[SmxGuide]	SMX Make yourself guide	SmxGuide	1.6	2018-04-07

2 Unbundled Smart Meter Architecture. Prototype Overview

The USM is composed from (i) a Smart Metrology Meter (SMM) i.e. a certified smart meter suitable to measure, using a trusted/certified method, various parameters of the electricity transferred in the point of meter connection, most important among them being the value of the electrical energy in a given time interval and (ii) a so-called smart meter extension (SMX), hosting the SMX Core and one or more SMX modules. SMX can be seen as a combination of modular software running on a dedicated small-form Personal Computer (PC) (namely the SMX hardware), which can host plug-in components providing added-value services.
The overall, high-level structure of prototype USM Architecture is summarized in Figure 1.

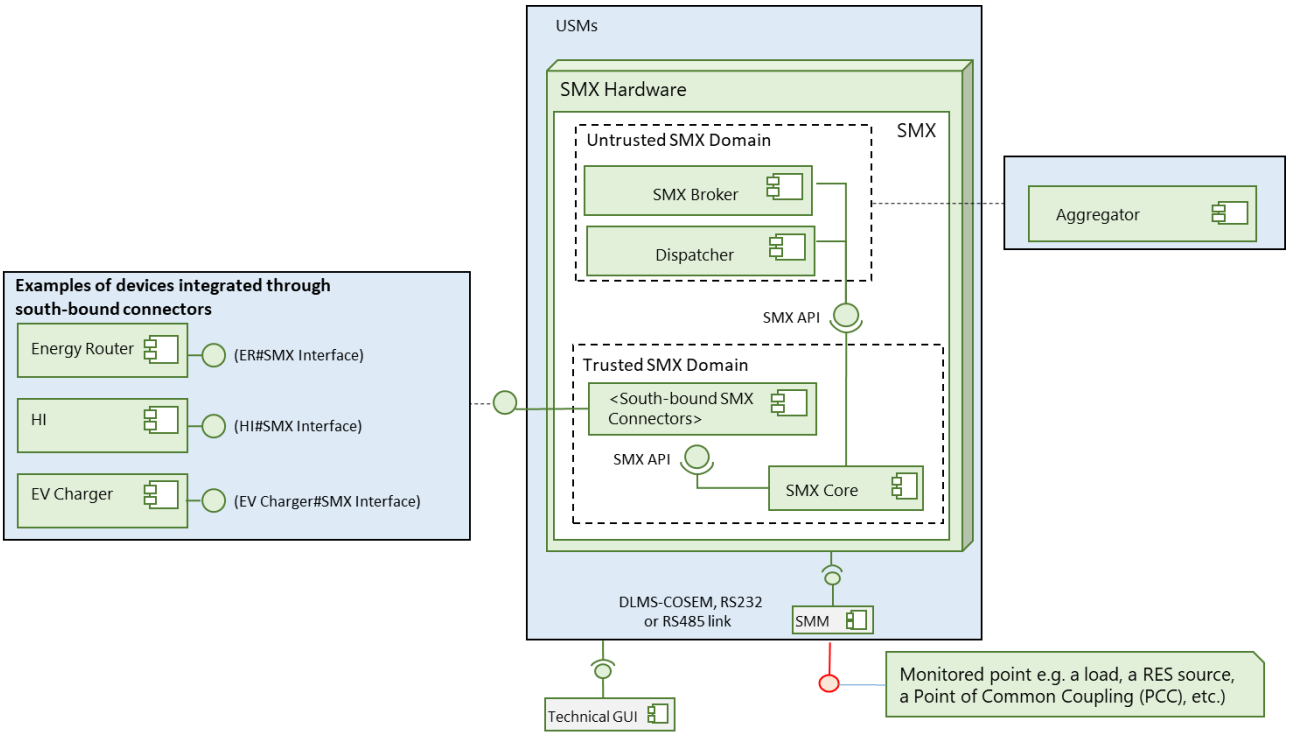

Figure 1. The prototype structure, including an example of SMX instantiation

The sub-components of the prototype are shortly summarized in the following subsections:

2.1 Smart Metrology Meter (SMM)

2.1.1 Billing values in SMM

Electricity meters are deployed for their main functionality: source of information for billing. This is usually made either by reading

- a meter index at a-priori specified intervals of time (e.g. each beginning of month), in case of single tariff or,
- by reading meter indexes associated with different tariffs.

In case of smart energy meters used for active electrical energy, the so-called "billing values" can be reported in digital form and read remotely. These values are usually made available in one of the following forms:

- Indexes of active energy: these are usually indexes which only increase their value until the index recirculates, meaning that it overpasses its maximum value, e.g. 999999.99
- Energies recorded in load profiles related to specific time intervals.

Meters can have separate indexes for each direction of the active and reactive energy transferred through the Point of Common Coupling (PCC) where the meter is installed, meaning separate IdxA+ and IdxA-, measured in [kWh] or [MWh] and separate indexes of reactive energies, meaning separate IdxR+ and IdxR-, measured in [kvarh] or [Mvarh]. Meters can have Indexes associated with each tariff, in case of multiple tariffs, while a list of these indexes can be stored at specified moments, usually at the beginning of a new month.

2.1.2 Instrumentation values in SMM

The following instrumentation values are made available and obtained directly (without additional calculations) or with simple calculations using other values from the SMM (also from those meters on the market which have some of the Smart Meter functionalities[i]) are:

- Voltage value (rms), on each phase, at moment k – U(k) for single-phase meters and U1(k), U2(k), U3(k) for three-phase meters, where U(k) or Ux(k) represents the rms value of the voltage available on the communication interface. To be noted that usually the time window TW for calculating this value is not specified by the meter manufacturer. Usually these values are obtained from the secondary winding of instrument transformers and are expressed in volts [V]. Meters with indirect connection (voltage meter input connected to a voltage transformer (VT) and current meter input connected to a current transformer (CT)) may give the voltage level either as real measurements in the secondary circuits (specific to the standard 57 V range which correspond to 100 V phase-to-phase voltage of VT) or calculated as to correspond to the primary circuit values, i.e. secondary values multiplied by the VT ratio.
- Current value (rms) on each phase, at moment k – I(k) for single-phase meters and I1(k), I2(k), I3(k) for three-phase meters, where I(k) or Ix(k) represents the rms value of the current available on the communication interface.
- Active power value on each phase at moment k – P(k) for single-phase meters and P1(k), P2(k), P3(k) for three-phase meters. To be noted that usually the time window TW for calculating this value is not specified by the meter manufacturer. In case that the SMM transfer quantities to be calculated from other instrumentation values. In case that meters do not provide directly the active power, it is needed to be calculated in SMX based on voltage, current and power factor or angle between voltage and current. Another possibility is to use the active energy indexes at the beginning and at the end of a time period, e.g. 15 or 60 minutes and to calculate in SMX the average power during this period.
- Reactive power value on each phase at moment l– Q(k) for single-phase meters and Q1(k), Q2(k), Q3(k) for three phase meters. To be noted that usually the time window Tw for calculating this Q value is not specified by the meter manufacturer. In case that meters do not provide directly the active power, it is needed to be calculated in SMX based on voltage, current and power factor or angle between voltage and current. Another possibility is to use the active energy indexes at the beginning and at the end of a time period, e.g. 15 or 60 minutes and to calculate in SMX the average power during this period.
- Power factor K
- Frequency f of the network.

2.2 S4G Instrumentation Components

Similar with the main meter used for billing the (AC) electrical energy, presented as SMM, other components of the local system hosting energy storage need also specific instrumentation values. However, the additional instrumentation values may not be necessary enforced by metrology, as being sub-metering information, thus not relevant for the point of common coupling where the contractual relation between the local system and the DSO is sealed. It is important to correlate the real time values (e.g., from the ER) with quasi-real time values (e.g., from the storage system) and instrumentation values (information averaged over TW period).

2.2.1 Sub-metering local ESS

Similar with the SMM, local ESS instrumentation values need to be acquired by SMX through specific interface. A commercial or vendor specific ESS can be integrated by using either a special SW extension which can read the instrumentation values, or another USM having as main role the provision of the electrical data. In S4G project are considered to be used USMs, which bring the same type of data to be read for all these points.
The following data is used in the S4G project:

- The active power $P(t_k)$ on each phase of the meter connection; t_k is the reporting moment, with acceptable synchronization features. The usual reporting rate is given by t_k-t_{k-1}=2 seconds for sending metering data as MQTT messages to the data broker; for logging and offline S4G applications SMX is providing two types of reporting rates in daily files recorded locally: records at each 5 seconds and records at each 1 minute, in case that the active power is not given directly by SMM, calculations as described in section 2.1.2 are made in SMX part of USM, by using available instrumentation values from SMM.
- The rms value of the voltage $U(t_k)$ in the metered point: t_k is the reporting moment, with acceptable synchronization features; the usual reporting rate is given by t_k-t_{k-1}=2 seconds for sending metering data as MQTT messages to the data broker. Daily records are stored locally at a reporting rate of 5 seconds and 1 minute.
- Active energy A+ and A- (in both directions): defined by multiplication of the active power with the reporting time, e.g.: $A+=P*(t_k-t_{k-1})$. The A+ and A- values are automatically calculated inside SMM and provided as results for being used by S4G applications.

2.2.2 Sub-metering EV

EVs charging parameters need also to be read by the S4G system, to help the optimisation of using the storage resources. For this purpose, in S4G project are used USMs, bringing the same type of data to be read for all these points with the help of EV charging connectors. The extension is reading the EV meter in the main USM. The following data are needed for the project:

- The active power $P(t_k)$ on each phase of the meter
- The voltage $U(t_k)$ in the metering point
- Active energy A+ and A- (in both directions)

2.2.3 Sub-metering ER

The ER will measure its electrical data and make it available through the ER SMX South-bound connector. The data is available in the ER is described in section 8.2. Please note that in cases where the ER energy needs to be billed (or any other reason) an USM will be considered.

2.2.4 Sub-metering PV

PV production is present in all pilot sites and is considered to be paired with the storage resources. PV are separate resources which are difficult to be interfaced with SMX, therefore the project is using an USM, as it is a common solution in previous cases.
The following data are needed for the project:

- The active power P(k) on each phase of the meter
- The voltage U(k) in the metered point
- Active energy A+ and A- (in both directions)

2.2.5 Sub-metering Fronius

The Fronius devices used in the project need to be also monitored also in terms of electrical data. While a direct communication with Fronius is presented in section 3.4, in particular cases Fronius device power can be also measured externally by an SMM. This situation can be used to measure commercial inverter input/output when the direct connection is not possible. The situation is in this case an ESS submetering and is described in 2.2.1 section.

3 Smart Meter eXtensions – SMX. Prototype Overview

3.1 Extension regarding the inter-connection with GUI for technical users

During the phase 2, test sites development and installation of the SMXs at the partner's premises, it was identified the need of a technical interface. In the testing stage, obtaining an instantaneous reaction from the analysed connection was important to identify possible errors. Considering this, a Technical Graphical User Interface (GUI) was realized as an additional interface. This includes different tabs which can ease the monitoring of a correct functioning of the SMX and the correct communication between SMM and SMX.
An important tab of the interface is named *LESSAg*. This one is associated to storage systems, and enables the visualization of a planning for the electrical energy storage.

Also, for technical objectives an extension related to Raspberry Pi (SMX) health monitoring was developed. It has been installed on SMXs exploiting a single bash script. The quoted application exposes through a WEB GUI the hardware status and other relevant information (such as CPU temperature and overall CPU and RAM loads). Other than that, it has been developed and installed on each SMX a dedicated script, run by the Unix time-based job scheduler (cron) twice a day, that aims to monitor the disk usage. The monitoring script described generates an automatic email alert towards the IT technicians involved on the Storage4Grid project, in case the disk usage grows over a certain threshold. In particular, it currently raises an alert if the disk free percentage is less than 15% of the overall memory.

3.2 Extension for integration with Grid-side ESS (HLUC – 1)

HLUC-1 targets the advanced prosumer (AP) with energy resilience, having the potential to exchange energy with his neighbours and to provide storage services to the grid on a voluntary basis to increase its income. To provide a storage service, AP can absorb, on request, excess energy from the network, as LESSAg (as AP local agent) can accept requests of storage services from an external entity which orchestrates grid storage resources.

LESSAg can provide Storage as a Service (SaaS) for the grid by using a part of the AP local storage, thus this part of the local storage can be considered as a virtual grid storage resource. In this respect, the grid storage

coordinating entity - such as Grid Side Energy Storage System Controller (GESSCon) or other similar grid integrator, needs to be simulated in HLUC-1 in order to prove the capability of AP for sharing a part of its storage to act as grid storage. Even if the AP local storage which can be offered for grid services is much smaller than directly connected grid storage resources, it is considered that by aggregating many such small storage means it can be obtained an acceptable level of aggregated storage from these Aps, acting as a virtual grid storage.

The partial use of the AP local storage as a virtual grid-side ESS is realized through LESSAg component placed in Edge Layer. The messages which describe ESS grid services, such as charging or discharging the local battery which performs the SaaS, are processed through an extension of LESSAg. In order to simulate the SaaS, in HLUC-1 it is simulated the receival of GESSCon messages, by sending MQTT based JSON data to LESSAg through the SMX broker.

3.3 Extension for integration with EVs (HLUC – 2)

HLUC-2 targets scenarios characterized by growing shares of intermittent power generation from RES while facing with increasing diffusion of electric vehicles.
The extension to connect the SMX with the EVs was designed to face the challenges for efficient management and grid stability generated by the HLUC-2 scenario. In this way, the extension allows monitoring and control of the EVs managed by PROFEV for optimizing, locally, the charging of EVs as well as the charging and discharging of storage systems [S4G-D4.7].

3.4 Extension for integration with ER (HLUC-1 and HLUC-3)

The extension to connect the SMX with the ER was designed to allow the monitor and control of the ERs in HLUC-1 (single-phase) and in HLUC-3 (three-phase). More information about the ER API is described in section 8.2. The deployment scheme of the ER extension is depicted in Figure 2.

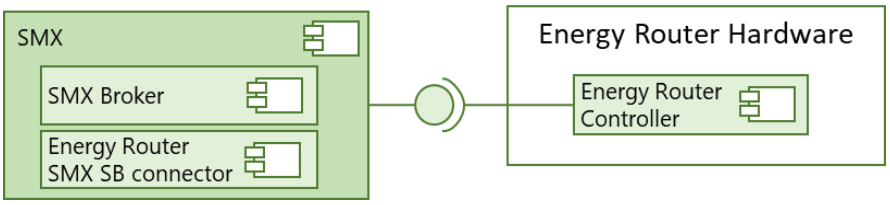

Figure 2. ER extension.

3.5 Extension for integration with Hybrid Inverter (HLUC-2 and HLUC-3)

The extension to connect the SMX with the Hybrid Inverter (i.e., the Fronius System) was designed to allow the monitor and control of the Hybrid inverter in HLUC-2 and HLUC-3. More information about the Fronius API is described in section 8.4. The deployment scheme of the Hybrid Inverter extension is depicted in Figure 3. This connector allows the control of the Fronius Inverter and the ESS connected to it.

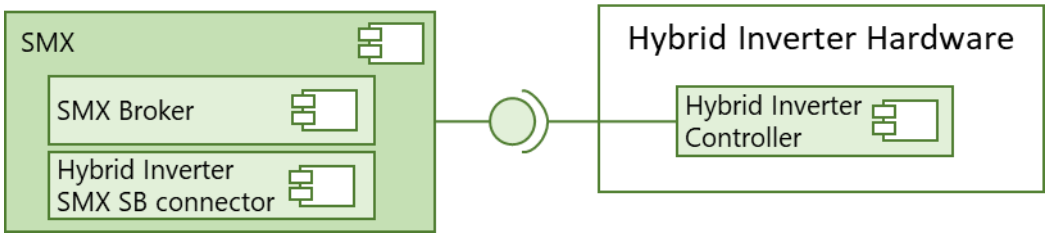

Figure 3. Hybrid Inverter extension.

4 SMX communication and information exchange with high-level application. Prototype Overview

The communication with the high-level components uses MQTT messaging to exchange data between SMX and Aggregator. The Aggregator receives Physical Layer devices data from SMX devices by means of various, heterogeneous wired or wireless local interfaces (ER#SMX interface, EV Charger#SMX interface, HI#SMX interface, SMM#SMX interface, SMX#Aggregator interface). Then, the Aggregator feed the data to higher-level components, through a dedicated, event-oriented interface, namely the Aggregator#Data Broker interface. Finally, Physical Layer devices data is feed directly to the DSF-DWH thanks to the Data Broker#DSF interface. This exchange is detailed in D3.3 [S4G-D3.3], D5.5 [S4G-D5.5] and it can be observed in Figure 4.

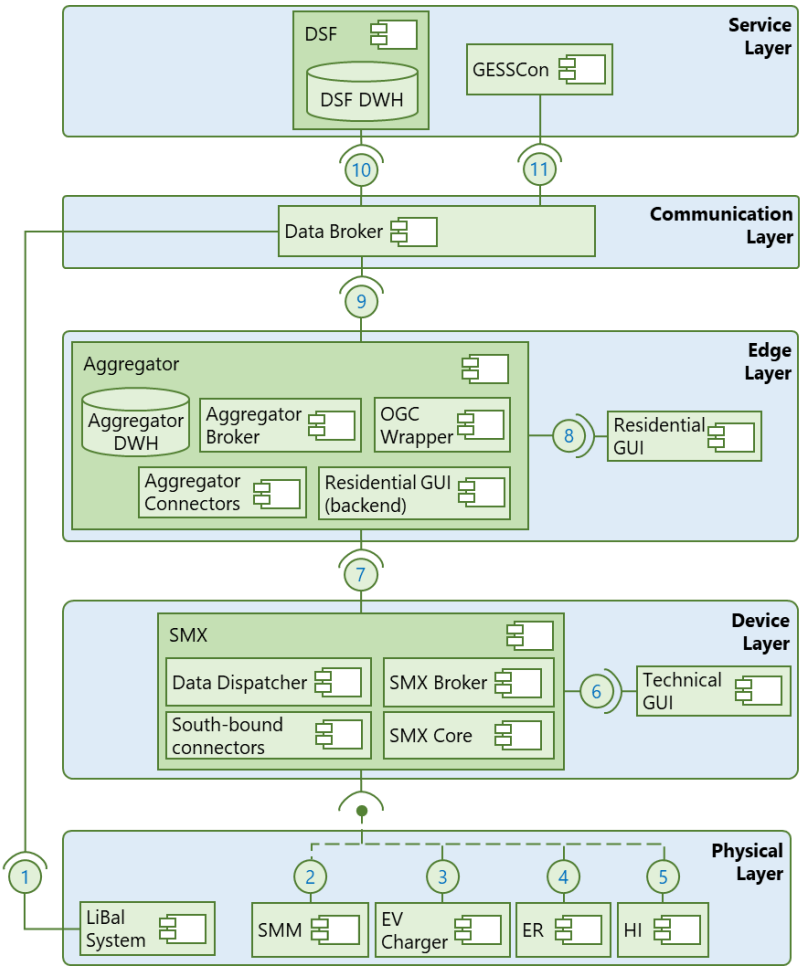

Figure 4. High-level applications connectors on layers

4.1 Minimal requirements for SMX

SMX is a Linux platform that runs on a compact Single Board Computer (SBC). The Raspberry Pi 3 Model B+ SBC has been chosen due to its quad-core architecture running at 1.4 GHz, its memory capacity of 1 GB of RAM and its storage capacity of up to 32 GB on SD-Card. In the S4G project, it is considered as sufficient a 16 GB SD-Card, which allows the installation of the Linux operation system and of the project related applications, while still allowing the recording of a high-resolution log of different variables.

4.2 Advanced requirements for SMX

The advanced requirements for SMX consist in the capabilities of the LESSAg/PROFESS/PROFEV component, a software component in charge of running site-wise ESS control algorithms. It receives in quasi-real-time all available information from local devices: ER which will transmit information on elements connected to its available ports: (the ESS system, PV, DC busbar), the AC load-side energy meters, and the AC PCC with the DSO.

5 Sub-components

5.1 Local Technical GUI SMX South-bound connector

This interface is able to show in real time on an associated web link data recorded by SMX at a certain measurement point by subscription to different topic. The latest version of the Local Technical GUI can be seen in Figure 5.

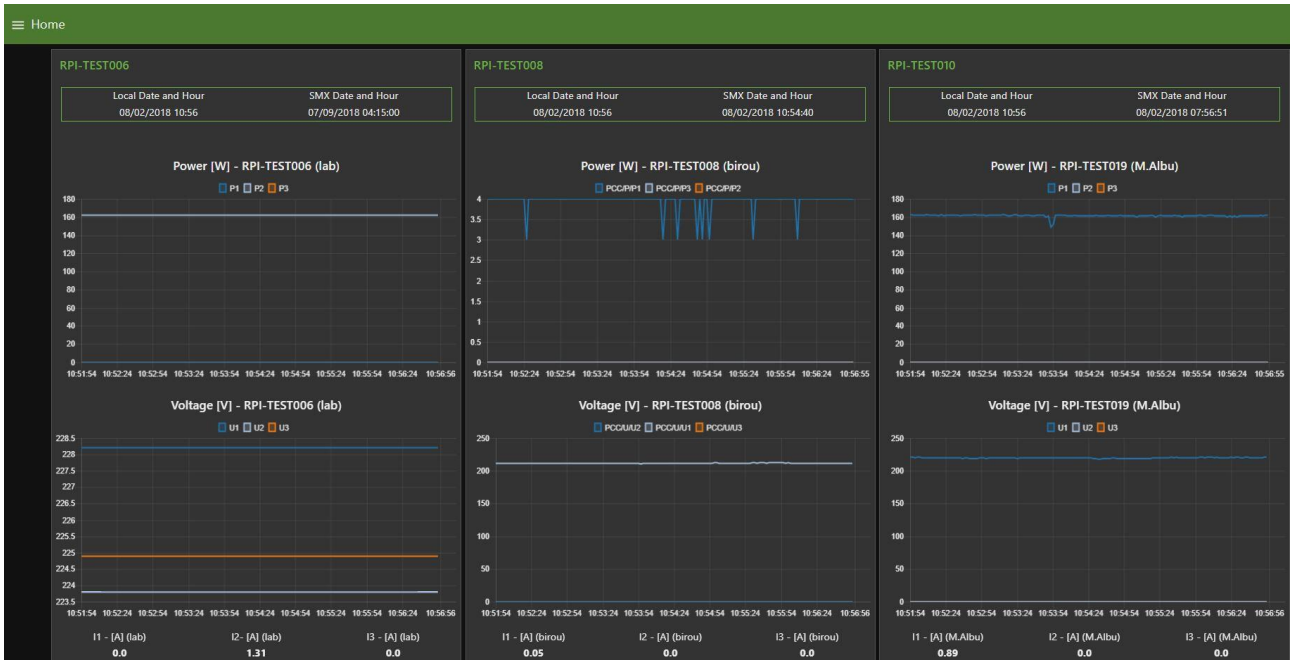

Figure 5. Local Technical GUI

5.2 ER SMX South-bound connector

This connector receives commands (set-points) from the LESSAg and PROFESS components. Moreover, it publishes real-time data in the SMX broker. The ER connector receives the set-points and forward them to the ER controller, through the ERController#ERConnector interface using the IEC61850-90-7 data model. More detailed information about this interaction and interface can be found in section 8.2.

5.3 EV Charger SMX South-bound connector

This connector receives real-time commands (set-points) from the PROFESS/PROFEV exploiting a secure MQTT communication on the cloud data and control broker, parse and build the proper requests towards the EV connector server, further described in D5.5 [S4G-D5.5]. In order to improve the robustness of the developed solution, this connector together with the EV connector server further verify the input provided by PROFESS/PROFEV to prevent wrong command forwarding. Part of the job of the current connector is also the forwarding of the relevant subset of real-time data coming from the Charging Units considered in this project. More detailed information about this interaction and interface can be found in section 8.4.

5.4 Hybrid Inverter SMX South-bound connector

This connector receives commands (set-points) from the PROFESS/PROFEV and publishes real-time data in the SMX broker. The Hybrid Inverter connector receives the set-points and forward them to the Hybrid Inverter controller, through the HIController#HIConnector interface using the Modbus TCP/IP protocol. More detailed information about this interaction and interface can be found in section 8.4.

5.5 Extension connector for receiving grid-side storage services requests

LESSAg is used for supporting the advanced prosumer to perform SaaS for the grid to an aggregating entity such as GESSCon. HLUC-1 needs to demonstrate the ability to receive such SaaS orders and to perform the service. The use of this connector is also presented in D5.5 [S4G-D5.5], where it is mentioned for the simulated grid-side request within a Hardware in the Loop (HIL) environment.
The extension, developed as a software function which has been embedded in the SMXcore package, supports the validation and unpacking of the SaaS request received as MQTT message and saves the information in the real-time database used by LESSAg algorithm.

6 Installation/Deployment Instructions

6.1 Local Technical GUI SMX South-bound connector

The technical GUI was developed using an open-source program developed for Linux distributions called *Node-Red*[ii]. It allows a quick, easy and convenient configuration of an interface by interconnecting some nodes specific to Internet-based Things (IoT) platforms. The basic operating system for *Raspberry Pi* is *Raspbian OS*, and it comes with a pre-installed version of the *Node-Red*. However, it is recommended to update the original version, and an additional module called Node-Red-Dashboard[iii] was used.
In order to install an interface on the SMX following steps are required:

- Connect to your SMX (Raspberry Pi) using *putty.exe* and update node-red using the following command inside putty: <update-nodejs-and-nodered> and then start node-red as a system service using: <sudo systemctl enable nodered.service>. First time it is required to start it manually, so just type: <node-red-start>
- In your web browser, navigate to the node-red page using the SMX IP - you can find out which IP your SMX has using: <hostname –I> (*capital i, not small letter L)* and install node-red-dashboard by clicking on the Menu button (found in the upper-right corner of the interface) >> Manage pallette >> Install nodes >> type in: node-red-dashboard >> install it.
- Import from Clipboard the code for the desired interface (Menu >> Import >> Clipboard >> paste the received code there and click on "import on current flow") or, another option is to connect to your SMX through WinSCP (or similar software), navigate to */home/pi/.node-red/* and delete the file which is named *flows your-SMX-name.json* (if existing), copy the received *.json* file and rename it to have the same name pattern as original deleted one (*flows_your-SMX-name.json*). E.g.: *flows_RPI-TEST005.json*.

- Restart node-red using the following sequence in command line (putty):<node-red-stop>, <node-red-start>.
- Open your interface at hostname:1880/ui. E.g.: 192.168.1.4:1880/ui

6.2 Energy Router SMX South-bound connector

The ER SMX SB connector was implemented using a Raspberry PI 3 Model B[Iliv] (RPI3). The SMX image needs to be installed on the RPI3 using the instructions document available to the consortium [SmxGuide][v]. The Stretch Raspbian version is used as SMX OS.
The developed software used in this prototype can be found in the Storage4Grid Git official repository as SB-SMX-ER-connector[vi].
Copy the 'AuxFiles_v2.0' folder to the SMX and paste it at /home/pi/S4G/. Rename the folder to 'SB_SMX_ErConnector'. Edit the 'ErConnectorConfig.properties' file to comply with the system. To automatically start the ER Connector when the SMX starts, use the instructions available for the consortium[vii] and the 'ErConnector.service' file. Please do not edit the xml file as is a key file for the application.

6.3 EV Charger SMX South-bound connector

The EV Charger SMX South-bound connector is a python3-based software component. The python3 dependencies are pre-installed on each SMX. In order to install this connector, it is enough to copy and paste the related folder on the SMX component. Besides, a dedicated service should be created following the system specifications. Further installation details will be provided in the next version of this deliverable.

6.4 Hybrid Inverter SMX South-bound connector

The Hybrid Inverter SMX SB connector was implemented using a RPI3. The SMX image needs to be installed on the RPI3 using the instructions document available to the consortium [SmxGuide]. The Stretch Raspbian version was used as SMX OS.
The developed software used in this prototype can be found in the Storage4Grid Git official repository as SB-SMX-HI-connector[viii].
Copy the 'AuxFiles_v1.0' folder to the SMX and paste it at /home/pi/S4G/. Rename the folder to 'SB_SMX_HIConnector'. Edit the 'HIConnectorConfig.properties' file to comply with the system. To automatically start the HI Connector when the SMX starts, use the instructions available for the consortium and the 'HIConnector.service' file. Please do not edit the xml file as is a key file for the application.

6.5 Extension connector for receiving grid-side storage services requests

The extension which processes the SaaS requests is embedded as a function in SMXcore, similar with LESSAg. In order to be able to receive the SaaS requests, it is needed to parametrize SMXcore to enable an MQTT connector which subscribes to the SaaS service request topic, i.e., "SMX/LESSAg/SaaS_input". Other relevant MQTT instructions for receiving the MQTT message are:

```
"StartTime": "2019-06-20 00:00:00",
"EndTime": "2019-09-25 00:00:00"
```

where "StartTime" and "EndTime" are the time boundaries defining the period when the message is considered as valid. With these arguments it is possible to protect LESSAg in receiving SaaS requests in other moments than in the accepted period, which should be related with a contractual period for the services.
With these user-related instructions the MQTT client is able to insert the necessary data in the internal database of SMXcore, in order to be used by LESSAg. There is no other instruction needed for installation or deployment.

7 Software dependencies and requirements

Table 3. Software Dependencies

Dependency	License	Role
Apache httpd, version 2.4.26	Apache version 2.0	Used to register and expose web-pages for the GUI, as well as proxying content over SSL channels
OpenIEC61850[ix], version 1.5.0	Apache License 2.0	This software is used to exchange data between the ER Connector and the ER using the IEC61850 protocol.
Jamod[x], version	Apache License 2.0	This software is used to exchange data between the Hybrid Inverter Connector and the Hybrid Inverter using the Modbus protocol.
Java-OCA-OCPP[xi]	MIT	A client and server library of Open Charge-Point Protocol from openchargealliance.org. Used in the "Extensions for integration of local EV charging station"
SMXCore	GPL	Open source Core application running in the trusted domain, implementing the real-time database, the communication with SMM and with all extensions running around SMXCore, by using also a RBAC system to preserve security and privacy for the used data
Node-RED[ii]	Apache License 2.0	Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

8 API Reference

8.1 Local Technical GUI SMX South-bound connector

Table 4. Parameters recorded by SMX and presented on the interface

Read and recorded data	Parameter on interface
SySDate: The date of the system	Yes
SySTime: The time of the system	Yes
Qn: Reactive Power on phase n [kvar]	No
Pn: Active Power on phase n [kW]	Yes
Un: Voltage on phase n [V]	Yes
In: Current on phase n [A]	Yes
Kn: Power factor on phase n [-]	No
Ap: Consumed active electric energy [kWh]	Yes

Am: Produced active electric energy [kWh]	Yes
Rp: Consumed reactive electric energy [kvarh]	Yes
Rm: Produced reactive electric energy [kvarh]	Yes
f: Frequency [Hz]	Yes
SySCpuLoad: Level of charge of the system process unit	No

The Node-RED interface receives by subscribing to different topics as MQTT messages are presented in Figure 6 and Figure 7.

Function

```
var U1 = {payload: msg.payload["1-1-32-7-0-255"] ? msg.payload["1-1-32-7-0-255"]["-2"]:0, topic: "U1"};
var U2 = {payload: msg.payload["1-1-52-7-0-255"] ? msg.payload["1-1-52-7-0-255"]["-2"]:0, topic: "U2"};
var U3 = {payload: msg.payload["1-1-72-7-0-255"] ? msg.payload["1-1-72-7-0-255"]["-2"]:0, topic: "U3"};
var I1 = {payload: msg.payload["1-1-31-7-0-255"] ? msg.payload["1-1-31-7-0-255"]["-2"]:0, topic: "I1"};
var I2 = {payload: msg.payload["1-1-51-7-0-255"] ? msg.payload["1-1-51-7-0-255"]["-2"]:0, topic: "I2"};
var I3 = {payload: msg.payload["1-1-71-7-0-255"] ? msg.payload["1-1-71-7-0-255"]["-2"]:0, topic: "I3"};
var P1 = {payload: msg.payload["1-1-36-7-0-255"] ? msg.payload["1-1-36-7-0-255"]["-2"]:0, topic: "P1"};
var P2 = {payload: msg.payload["1-1-56-7-0-255"] ? msg.payload["1-1-56-7-0-255"]["-2"]:0, topic: "P2"};
var P3 = {payload: msg.payload["1-1-76-7-0-255"] ? msg.payload["1-1-76-7-0-255"]["-2"]:0, topic: "P3"};
var Q1 = {payload: msg.payload["1-1-151-7-0-255"] ? msg.payload["1-1-151-7-0-255"]["-2"]:0,topic: "Q1"};
var Q2 = {payload: msg.payload["1-1-171-7-0-255"] ? msg.payload["1-1-171-7-0-255"]["-2"]:0, topic: "Q2"};
var Q3 = {payload: msg.payload["1-1-191-7-0-255"] ? msg.payload["1-1-191-7-0-255"]["-2"]:0, topic: "Q3"};
var f = {payload: msg.payload["1-1-14-7-0-255"] ? msg.payload["1-1-14-7-0-255"]["-2"]:0, topic: "f"};
var P = {payload: msg.payload["1-1-16-7-0-255"] ? msg.payload["1-1-16-7-0-255"]["-2"]:0, topic: "P"};
var Q = {payload: msg.payload["1-1-131-7-0-255"] ? msg.payload["1-1-131-7-0-255"]["-2"]:0, topic: "Q"};
var Ap = {payload:msg.payload["1-1-1-8-0-255"] ? msg.payload["1-1-1-8-0-255"]["-2"]:0, topic: "Ap"};
var Am = {payload: msg.payload["1-1-2-8-0-255"] ? msg.payload["1-1-2-8-0-255"]["-2"]:0, topic: "Am"};
var Rp = {payload: msg.payload["1-1-3-8-0-255"] ? msg.payload["1-1-3-8-0-255"]["-2"]:0, topic: "Rp"};
var Rm = {payload: msg.payload["1-1-4-8-0-255"] ? msg.payload["1-1-4-8-0-255"]["-2"]:0, topic: "Rm"};
var K = {payload: msg.payload["1-1-33-7-0-255"] ? msg.payload["1-1-33-7-0-255"]["-2"]:0, topic: "K"};
var K1 = {payload: msg.payload["1-1-33-7-0-255"] ? msg.payload["1-1-33-7-0-255"]["-2"]:0, topic: "K1"};
var K2 = {payload: msg.payload["1-1-53-7-0-255"] ? msg.payload["1-1-53-7-0-255"]["-2"]:0, topic: "K2"};
var K3 = {payload: msg.payload["1-1-73-7-0-255"] ? msg.payload["1-1-73-7-0-255"]["-2"]:0, topic: "K3"};
var SMM = {payload: msg.payload["1-1-32-7-0-255"] ? msg.payload["1-1-32-7-0-255"]["-5"]:0, topic: "SMM_Time_Albu"};

return [
    P1.payload ? P1:null,
    P2.payload ? P2:null,
    P3.payload ? P3:null,
    U1.payload ? U1:null,
    U2.payload ? U2:null,
    U3.payload ? U3:null,
    I1.payload ? I1:null,
    I2.payload ? I2:null,
    I3.payload ? I3:null,
    SMM.payload ? SMM:null
];
```

Figure 6. A Node-RED function description

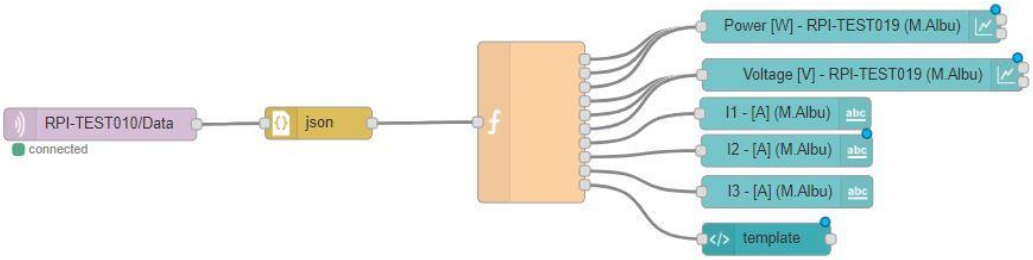

Figure 7. Node-RED structure for an SMX through .json messages

8.2 ER SMX South-bound connector API

PROFESS (or another component, e.g. PROFEV, LESSAg) can send set-points to both ERs using the MQTT topics described in following subsections. To define a set-point, the PROFESS (or another component) should publish in the corresponding topic using the message template described in Figure 8, where "n" should have the set-point parameter value.

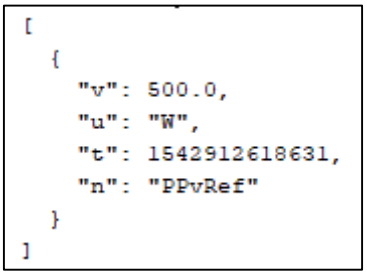

Figure 8. ER SenML JSON message template.

8.2.1 Three-phase ER connector API

Table 5 shows the three-phase ER set-points topics, while Table 6 presents the MQTT topics to receive real-time values of the three-phase ER. More information about the three-phase ER operation can be found in D4.11 [S4G-D4.11].

Table 5. Topics to define three-phase ER set-points.

Parameter	Description	Units	Topic
DERStart	*Three-phase ER operating modes:* 0 = OFF 1 = Standby 2 = PV On 3 = PV Off 4 = Load balancing 5 = PRef Control	N/A	/PROFESS/SMX/EnergyRouterInverter/DRCC1.DERStr.ctlNum
PBatRef	Battery Power Reference	W	/PROFESS/SMX/EnergyRouterInverter/ZBTC1.BatChaPwr.setMag.f
PPvRef	PV Power Reference	W	/PROFESS/SMX/EnergyRouterPV/MMDC1.Watt.subMag.f
QGridRef	ER Reactive Power Reference	VAr	/PROFESS/SMX/EnergyRouterInverter/MMXU1.TotVAr.subMag.f
QaGridRef	ER Reactive Power Phase A Reference	VAr	/PROFESS/SMX/EnergyRouterInverter/MMXU1.VAr.phsA.subCVal.mag.f

QbGridRef	ER Reactive Power Phase B Reference	VAr	/PROFESS/SMX/EnergyRouterInverter/MMXU1.VAr.phsB.subCVal.mag.f
QcGridRef	ER Reactive Power Phase C Reference	VAr	/PROFESS/SMX/EnergyRouterInverter/MMXU1.VAr.phsC.subCVal.mag.f

Table 6. Topics to receive three-phase ER real-time values.

Parameter	Description	Units	Topic
SoC	Battery SoC	%	/ER/SMX/EnergyRouterInverter/ZBAT1.VolChgRte.instMag.f
PBat	Battery Power	W	/ER/SMX/EnergyRouterInverter/ZBTC1.BatChaPwr.setMag.f
PPv	PV Power	W	/ER/SMX/EnergyRouterPV/MMDC1.Watt.instMag.f
PGrid	ER Active Power	W	/ER/SMX/EnergyRouterInverter/MMXU1.TotW.instMag.f
PaGrid	ER Active Power Phase A	W	/ER/SMX/EnergyRouterInverter/MMXU1.W.phsA.instCVal.mag.f
PbGrid	ER Active Power Phase B	W	/ER/SMX/EnergyRouterInverter/MMXU1.W.phsB.instCVal.mag.f
PcGrid	ER Active Power Phase C	W	/ER/SMX/EnergyRouterInverter/MMXU1.W.phsC.instCVal.mag.f
QGrid	ER Reactive Power Phase A	VAr	/ER/SMX/EnergyRouterInverter/MMXU1.TotVAr.instMag.f
QaGrid	ER Reactive Power Phase A	VAr	/ER/SMX/EnergyRouterInverter/MMXU1.VAr.phsA.instCVal.mag.f
QbGrid	ER Reactive Power Phase B	VAr	/ER/SMX/EnergyRouterInverter/MMXU1.VAr.phsB.instCVal.mag.f
QcGrid	ER Reactive Power Phase C	VAr	/ER/SMX/EnergyRouterInverter/MMXU1.VAr.phsC.instCVal.mag.f

8.2.2 Single-phase ER connector API

The same template defined in Figure 8 is used to read and write values in the single-phase ER. Table 7 shows the single-phase ER set-points topics, while Table 8 presents the MQTT topics to receive real-time values of the single-phase ER. More information about the single-phase ER operation can be found in D4.12 [S4G-D4.12].

Table 7. Topics to define single-phase ER set-points.

Parameter	Description	Units	Topic
DERStart	*Single-phase ER operating modes:* 0 = OFF	*N/A*	/LESSAg/SMX/EnergyRouterInverter/DRCC1.DERStr.ctlNum

	1 = Standby (Grid ON) 2 = RPPT (Grid ON) 3 = MPPT (Grid ON) 4 = PV OFF (Grid ON) 5 = Standby (Grid OFF) 6 = RPPT (Grid OFF) 7 = MPPT (Grid OFF) ***8 = BLACK-START***		
PBatRef	Battery Power Reference	W	/LESSAg/SMX/EnergyRouterInverter/ZBTC1.BatChaPwr.setMag.f
PPvRef	PV Power Reference	W	/LESSAg/SMX/EnergyRouterPV/MMDC1.Watt.subMag.f
QGridRef	Reactive Power Reference	VAr	/LESSAg/SMX/EnergyRouterInverter/MMXN1.VolAmpr.subMag.f

Table 8. Topics to receive single-phase ER real-time values.

Parameter	Description	Units	Topic
DERStart	*ER operating modes:* 0 = OFF 1 = Standby (Grid ON) 2 = RPPT (Grid ON) 3 = MPPT (Grid ON) 4 = PV OFF (Grid ON) 5 = Standby (Grid OFF) 6 = RPPT (Grid OFF) 7 = MPPT (Grid OFF) ***8 = BLACK-START***	N/A	/ER/SMX/EnergyRouterInverter/DRCC1.DERStr.ctlNum
PBatRef	Battery Power Reference	W	/ER/SMX/EnergyRouterInverter/ZBTC1.BatChaPwr.setMag.f
PPvRef	PV Power Reference	W	/ER/SMX/EnergyRouterPV/MMDC1.Watt.subMag.f
QGridRef	Reactive Power Reference	VAr	/ER/SMX/EnergyRouterInverter/MMXN1.VolAmpr.subMag.f
SoC_BAT	Battery SoC	%	/ER/SMX/EnergyRouterInverter/ZBAT1.VolChgRte.instMag.f
PBat	Battery Power	W	/ER/SMX/EnergyRouterInverter/ZBAT1.Vol.instMag.f
UBat	Battery Voltage	V	/ER/SMX/EnergyRouterInverter/ZBTC1.ChaV.instMag.f
IBat	Battery Current	A	/ER/SMX/EnergyRouterInverter/ZBTC1.ChaA.instMag.f
PPv	PV Power	W	/ER/SMX/EnergyRouterPV/MMDC1.Watt.instMag.f

UPv	PV Input Voltage	V	/ER/SMX/EnergyRouterPV/MMDC1.Vol.instMag.f
IPv	PV Input Current	A	/ER/SMX/EnergyRouterPV/MMDC1.Amp.instMag.f
PGrid	ER Active Power	W	/ER/SMX/EnergyRouterInverter/MXN1.Watt.instMag.f
UGrid	ER RMS Voltage	V	/ER/SMX/EnergyRouterInverter/MMXN1.Vol.instMag.f
IGrid	ER RMS Current	A	/ER/SMX/EnergyRouterInverter/MMXN1.Amp.instMag.f
QGrid	ER Reactive Power	VAr	/ER/SMX/EnergyRouterInverter/MMXN1.VolAmpr.instMag.f
UDC-link	DC-link Voltage	V	/ER/SMX/EnergyRouterInverter/MMDC1.Vol.instMag.f

8.3 EV Charger SMX South-bound connector

Together with the EV connector server, the current connector parse and forward the messages received through the SIEMENS system towards PROFESS/PROFEV via MQTT exploiting the SenML format. Consequently, the format of all the measurement messages exchanged, generated by the Charging Units, converted by the EV South-bound connector and sent towards PROFESS/PROFEV, can be found in Figure 9.

```
[
  {
    "bn": "chargers/{ChargingUnitID}",
    "n": "ev{InternalSessionID}/SoC",
    "v": SoC_value,
    "t": utcTimestamp
  }
]
```

```
[
  {
    "bn": "chargers/X-Y-Z",
    "n": "ev0/SoC",
    "v": 0.50,
    "t": 1553795978749
  }
]
```

Figure 9. Measurement messages sent to PROFESS/PROFEV converted by the EV South-bound connector

Whenever a new recharge session begins or end it will be sent the MQTT message on the "EV/Range" topic by setting the SenML value "v" to 0 and 1 respectively.
Within a recharge session will be sent periodically the MQTT messages on the "EV/Data" topic by setting the SenML value "v" with the currently estimated SoC in %.

Table 9. Topics to receive data from EV Charger

Information	Units	Topic
EV recharge (begin/end)	Boolean (0/1)	/EV/Range
EV State of charge	%	/EV/Data

The format of all the control messages exchanged towards the Charging Units, generated by PROFESS/PROFEV, received and parsed by the current connector, is presented below in Figure 10:

```
[
  {
    "bver": {version},
    "bn": "chargers/{ChargingUnitID}",
    "n": "ev{InternalSessionID}/P",
    "v": Power_value,
    "t": utcTimestamp
  }
]
```

```
[
  {
    "bver": 5,
    "bn": "chargers/X-Y-Z",
    "n": "ev0/P",
    "v": 2000.0,
    "t": 1553795978749
  }
]
```

Figure 10. Control messages exchanged towards the Charging Units, generated by PROFESS/PROFEV

Whenever PROFESS/PROFEV needs to define a new set-point, it will send the MQTT message on the "EV/Control" topic by setting the SenML value "v" to the power to be injected in Watt.

Table 10. Topics to send data to EV Charger

Information	Units	Topic
EV command (Power)	Watt	/EV/Control

8.4 Hybrid Inverter SMX South-bound connector API

Similar to the ER connector, the Hybrid Inverter connector can receive set-points from different components. It publishes real-time values (Table 11) from the Hybrid Inverter system and receives set-points (Table 12) using also the message template described in Figure 8, where "n" should have the set-point parameter value.

Table 11. Topics to receive Fronius System real-time values.

Parameter	Description	Units	Topic
W_Inverter	AC Power value (inverter output).	W	/Fronius/SMX/W_Inverter
VAr_Inverter	Reactive Power (inverter output).	VAr	/Fronius/SMX/VAr_Inverter
WRtg	Continuous power output capability of the inverter.	W	/Fronius/SMX/WRtg
VArRtgQ1	Continuous VAR capability of the inverter in quadrant 1	VAr	/Fronius/SMX/VArRtgQ1
VArRtgQ4	Continuous VAR capability of the inverter in quadrant 4	VAr	/Fronius/SMX/VArRtgQ4
VARMaxQ1	Setting for maximum reactive power in quadrant 1. Default to VArRtgQ1.	Var	/Fronius/SMX/VARMaxQ1
VARMaxQ4	Setting for maximum reactive power in quadrant 1. Default to VArRtgQ4.	VAr	/Fronius/SMX/VARMaxQ4
PV_DCW	PV DC Power	W	/Fronius/SMX/PV_DCW

ESS_DCW	ESS DC Power	W	/Fronius/SMX/ESS_DCW
ChaSt	Charge status of storage device: 1: OFF 2: EMPTY 3: DISCHARGING 4: CHARGING 5: FULL 6: HOLDING 7: TESTING	N/A	/Fronius/SMX/ChaSt
ChaState	Currently available energy as a percent of the capacity rating.	%AhrRtg	/Fronius/SMX/ChaState
WMax	Setting for maximum power output. Default to I_WRtg.	W	/Fronius/SMX/WMax
Conn_WinTms	Time window for connect/disconnect.	Secs	/Fronius/SMX/Conn_WinTms
Conn	Connection control: 0: Disconnected 1: Connected	N/A	/Fronius/SMX/Conn
WMaxLimPct	Set power output to specified level.	%WMax	/Fronius/SMX/WMaxLimPct
WMaxLimPct_WinTms	Time window for power limit change.	Secs	/Fronius/SMX/WMaxLimPct_WinTms
WMaxLim_Ena	Throttle enable/disable control: 0: Disabled 1: Enabled	N/A	/Fronius/SMX/WMaxLim_Ena
VArMaxPct	Reactive power in percent of I_VArMax.	%VArMax	/Fronius/SMX/VArMaxPct
VArPct_WinTms	Time window for VAR limit change.	Secs	/Fronius/SMX/VArPct_WinTms
VArPct_Ena	Fixed VAR enable/disable control: 0: Disabled 1: Enabled	N/A	/Fronius/SMX/VArPct_Ena
ChaGriSet	Set-point to enable/disable charging from grid: 0: PV (Charging from grid disabled) 1: GRID (Charging from grid enabled)	N/A	/Fronius/SMX/ChaGriSet
V_Grid	AC Voltage Average Phase-to-neutral value (Grid).	V	/Fronius/SMX/V_Grid
W_Grid	AC Power value (Grid).	W	/Fronius/SMX/W_Grid
VAr_Grid	AC Reactive Power value (Grid).	VAr	/Fronius/SMX/VAr_Grid

W_Load	AC Power value (Load).	W	/Fronius/SMX/W_Load

Table 12. Topics to define Fronius System set-points.

Parameter	Description	Units	Topic
WMax	Setting for maximum power output. Default to I_WRtg.	W	/PROFESS/Fronius/WMax
Conn_WinTms	Time window for connect/disconnect.	Secs	/PROFESS/Fronius/Conn_WinTms
Conn	Connection control: 0: Disconnected 1: Connected	N/A	/PROFESS/Fronius/Conn
WMaxLimPct	Set power output to specified level.	%WMax	/PROFESS/Fronius/WMaxLimPct
WMaxLimPct_WinTms	Time window for power limit change.	Secs	/PROFESS/Fronius/WMaxLimPct_WinTms
WMaxLim_Ena	Throttle enable/disable control: 0: Disabled 1: Enabled	N/A	/PROFESS/Fronius/WMaxLim_Ena
VArMaxPct	Reactive power in percent of I_VArMax.	%VArMax	/PROFESS/Fronius/VArMaxPct
VArPct_WinTms	Time window for VAR limit change.	Secs	/PROFESS/Fronius/VArPct_WinTms
VArPct_Ena	Fixed VAR enable/disable control: 0: Disabled 1: Enabled	N/A	/PROFESS/Fronius/VArPct_Ena
ChaGriSet	Set-point to enable/disable charging from grid: 0: PV (Charging from grid disabled) 1: GRID (Charging from grid enabled)	N/A	/PROFESS/Fronius/ChaGriSet

8.5 Extension connector for receiving grid-side storage services requests

Data for providing storage services can be received by this connector as MQTT message in JSON format, in a vector array with the structure data followed presented.

```
[
  {  "n": "ESS_Control",  "t": 1532154800,  "v":0.5,  "u":"kW"  },
  {  "n": "ESS_Control",  "t": 1532158400,  "v":0.8,  "u":"kW"  },
```

{ "n": "ESS_Control", "t": 1532162000, "v":0.6, "u":"kW" },
{ "n": "ESS_Control", "t": 1532165600, "v":0.4, "u":"kW" },
...
]

where each hourly time period has its own object, with the following meaning of the internal structure:

- **n**: "*ESS_Control*" is the specific key for an hourly object describing the storage service.
- **t**: gives the time corresponding to the beginning of the period when storage service is needed for the grid, e.g. 1538704800.
- **v**: *value*, gives the storage order (power to be absorbed or injected in the grid, as grid service), e.g., the value 0.5.
- **u**: *unit*, gives the unit of the power, e.g. "kW".

The dots at the end of the array shows that more objects can be present in the array.
The function takes automatically the received message and translates it in LESSAg data needed to perform the service request received from external agents.

9 Conclusions

This deliverable presents the "Final USM Extensions for Storage Systems" prototype, developed by the Storage4Grid project. Similarly, to other prototypes, this document provides minimal technical documentation necessary to understand the functionalities, structure and deployment instructions for the prototype of interest. The document presents the main extensions needed in the SMX for implementing a complete chain of functionalities to support efficient and optimized used of storage resources and other important components involved in the support process.

Acronyms

Acronym	Explanation
AC	Alternating Current
AP	Advanced Prosumer
API	Application Programming Interface
COSEM	Companion Specification for Energy Metering
CT	Current Transformer
DLMS	Device Language Message Specification
DSO	Distribution System Operator
ER	Energy Router
ESS	Energy Storage System
EV	Electrical Vehicle
GESSCon	Grid Side Energy Storage System Controller
GUI	Graphical user interface
HI	Hybrid Inverter
HIL	Hardware in the Loop
IoT	Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation
LESSAg	Local Energy Storage System Agent
MQTT	Message Queue Telemetry Transport
OCPP	Open Charge Point Protocol
OS	Operating System
PC	Personal Computer
PCC	Point of common coupling
PROFESS	Professional Realtime Optimization Framework for Energy Storage Systems
PROFEV	Professional Realtime Optimization Framework for Electric Vehicles
PV	Photovoltaic
RBAC	Role based access control
RES	Renewable Energy Source

Acronym	Explanation
RMS	Root mean square
RPI3	Raspberry PI 3 Model B
S4G	Storage4Grid
SaaS	Storage as a Service
SBC	Single Board Computer
SCADA	Supervisory control and data acquisition
SenML	Sensor Measurement Lists
SMM	Smart Metrology Meter
SMX	Smart Meter eXtension
SoC	State of Charge
SW	Software
USM	Unbundled Smart Meter
VT	Voltage Transformer
XML	Extensible Markup Language

List of figures

List of tables

References

[i] Mihaela Albu, Mihai Sănduleac, Carmen Stănescu, 2016, Syncretic use of smart meters for Power Quality monitoring in emerging networks, IEEE Transactions on Smart Grid, Volume: PP, Issue: 99
[ii] Node-RED» [online]. Available: https://nodered.org/. [accesed at 05-12-2017].
[iii] node-red-dashboard - Node-RED» [online]. Available: https://flows.nodered.org/node/node-red-dashboard. [accessed at 05-12-2017]
[iv] Raspberry PI 3 Model B, https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, Accessed 31 July 2017.
[v] https://confluence.fit.fraunhofer.de/confluence/display/S4G/SMX+Guide
[vi] SB-SMX-ER-connector project, https://git.repository-pert.ismb.it/storage4grid/ER-connector, Accessed 10 July 2018.
[vii] How to create an automatic script for starting and stopping, https://confluence.fit.fraunhofer.de/confluence/display/S4G/How+to+create+an+automatic+script+for+starting+and+stopping, Accessed 22 June 2018.
[viii] SB-SMX-Modbus-connector project, https://git.repository-pert.ismb.it/vmdg/SB-SMX-Modbus-connector, Accessed 10 July 2018.
[ix] OpenMuc, OpenIEC61850 Overview, https://www.openmuc.org/iec-61850/, accessed 2 September 2019
[x] Java Modbus Library, http://jamod.sourceforge.net/, accessed 2 September 2019.
[xi] https://github.com/ChargeTimeEU/Java-OCA-OCPP