D4.1 - Initial User-side ESS control system

Deliverable ID	**D4.1**
Deliverable Title	**Initial User-side ESS control system**
Work Package	**WP4**

Dissemination Level	**PUBLIC**

Version	**1.0**
Date	**30/08/2017**
Status	**final**
Type	**Prototype**

Lead Editor	**UNINOVA**
Main Contributors	**Anabela Pronto, João Martins, Ricardo Gonçalves, Pedro Ferreira, Vasco Delgado-Gomes (UNINOVA), Carlo Andrea Pigato (ISMB), Mihai Sanduleac (UPB), FRAUNHOFER, Rasmus Rode Mosbæk (LIBAL)**

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	2017-06-19	UNINOVA	First draft and initial inputs
0.2	2017-07-17	UPB	Contribution
0.3	2017-07-24	ISMB	Contribution on predictive algorithm
0.4	2017-07-28	ISMB	Revision of contribution on predictive algorithm
0.5	2017-07-31	UNINOVA	Prototype implementation results. Template update (prototype)
0.6	2017-08-22	UNINOVA	Version for internal review
1.0	2017-08-30	UNINOVA	Final version, ready for submission to the EC

Internal Review History

Review Date	Reviewer	Summary of Comments
2017-08-28 (v0.6)	Gitte Wad Thybo (ENIIG)	Approved: • General minor corrections • Missing acronyms in the list
2017-08-30 (v0.6)	Rasmus Rode Mosbæk (LIBAL)	Approved: • General minor corrections • Comments on next steps in terms of LESSAg algorithm development

Table of Contents

Executive Summary

D4.1 – "Initial User-side ESS control system" presents the initial developed prototype implementing some features of the User-side ESS control system.

The interaction between an emulated Cloud Component, a LESSAg, and the ER controller are presented in this deliverable. The User-side ESS control system is composed by the LESSAg and the LEVChAg (when EVs are present).

The Cloud Component that will interact with the LESSAg is the GESSCon, but since its developed have not started yet, in this deliverable, it is considered a simple application that issue commands to the LESSAg. The SMX interacts with the ER using an ER Connector. This Connector is deployed on the trusted zone of the SMX and exchanges information with the SMX Core using the Trusted EB. The LESSAg is deployed on the untrusted zone of the SMX and receives commands from the Cloud Components through a VPN, and send it to the SMX using the LESSAg EB.

For testing purposes, it was implemented an algorithm in the ER Connector to manage the unbalanced loads in the AC grid. This algorithm connects to the RTDB to receive information from the SMM and balances the phases accordingly. To initiate the algorithm, the LESSAg receives a "LoadBalancing" command.

In the following deliverable (D4.2 – "Updated User-side ESS control system"), other algorithms will be implemented both in the LESSAg and in the ER. The GESSCon and LESSAg interaction will use selected data standards and communication technologies, and will be detailed in the following deliverables. Some predictive algorithms are also planned to be implemented.

1 Introduction

D4.1 describes the "Initial 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 LESSAg and the ER controller is detailed. The LESSAg receives instructions from the Cloud Components (GESSCon). 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". Further updates of this prototypes are expected to be released as D4.2 – "Updated User-side ESS control system" (M18) and 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	0.5	2017-08-09
[D4.8]	Initial USM Extensions for Storage Systems	D4.8	0.9	2017-08-29

2 Initial User-side ESS Control System Prototype Overview

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

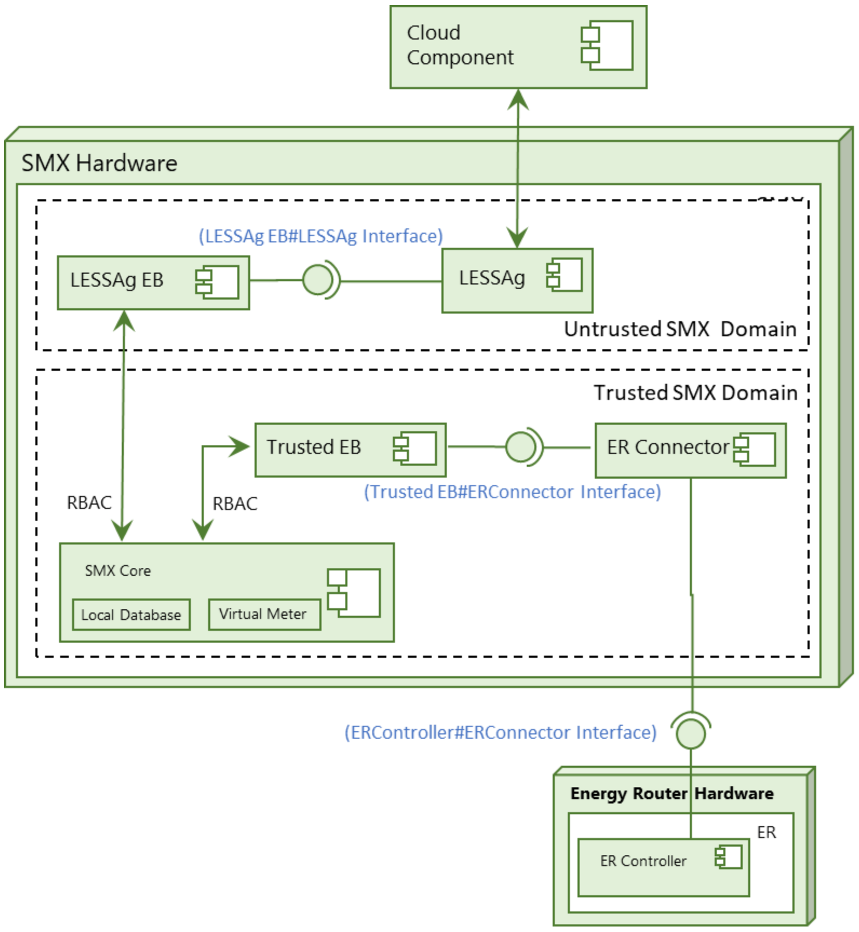

Figure 1. Prototype diagram.

The LESSAg receives a command from the Cloud Component (simulating the GESSCon) through a web service. The LESSAg receives the command, and according to the actor responsible, the task is forward to the respective component. In this prototype, the commands are only received by the ER Connector, because it is the only component which has subscribed the Trusted EB (implemented using the SMX MQTT broker).

The SMX interacts with the ER Controller using the ER Connector. This Connector is deployed on the trusted zone of the SMX and exchanges information with the SMX using the Trusted EB (MQTT broker). The Connector exchanges information with the ER Controller using the ERController#ERConnector interface. In this first version of the prototype, it was used the IEC 61850[i] standard. It is foreseen for the next version to use the IEC61850-90-7[ii], which is more oriented for the communication with power converters and RES.

For testing purposes, it was implemented an algorithm in the ER Connector to manage the unbalanced loads in the AC grid. This algorithm connects to the Trusted EB to receive information from a Virtual Meter and balances the phases accordingly, sending set-points to the ER controller. To initiate the algorithm, the LESSAg receives a "LoadBalancing" command. This interaction is described in Figure 2. Please note that for simplification purposes the LESSAg EB, the SMX Core, and the Trusted EB components are only represented by the EB component.

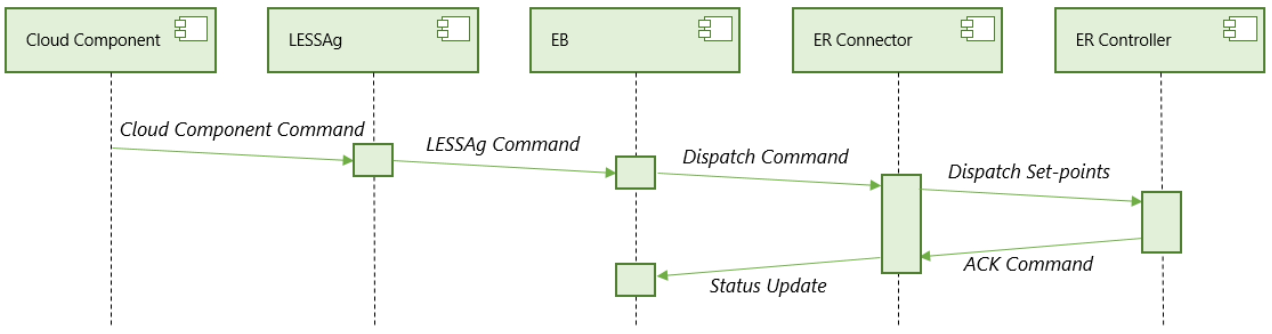

Figure 2. UML sequence diagram of the prototype.

It is worth to refer that the ER Connector will subscribe to a service in the Trusted EB to receive the LESSAg command through the LESSAg EB. Moreover, the Connector will also receive periodically the SMM measurements through the Trusted EB. In this prototype, the SMM was emulated using a virtual meter application of the SMX Core. The real-time measurements are received in the Connector, in order to provide data to the load balancing algorithm. The sub-components of the prototype are shortly summarized in the following subsections.

2.1 Cloud Component

This component is responsible to issue commands to the LESSAg. In this prototype, just one command was tested: load balancing in the AC grid. The command is sent using the following JSON message:

```
{
"Timestamp" : "2017-07-31T12:57:45Z",
"SMX" : {
        "IP" : "192.168.0.102",
        "Port" : "80"
        },
"Command" : "LoadBalancing"
}
```

In the following versions of this deliverable, it will be implemented the GESSCon#LESSAg interface. For more information regarding the interfaces is available in D3.1.

2.2 LESSAg

This component is responsible to provide the web service to receive the message from the Cloud Component and send it to the Reconnector using the EB#ERConnector interface.

2.3 Event Brokers

There are two EBs: the LESSAg EB and the Trusted EB. In this prototype, only one EB was used to speed up the development. Basically, the LESSAg EB is an untrusted EB, which connects to the SMX, and according to its permissions, the SMX can block it or allow its commands. In this way, the data in the SMX Core is secure from third party applications attacks.

2.4 Energy Router Connector

This component receives commands from the Trusted EB. The Connector will execute the necessary actions to answer the command. In this prototype, the Connector receives the virtual meter values and send set-points to the ER Controller, through the ERController#ERConnector interface, using the IEC61850 data model.

2.5 Energy Router Controller

This component is responsible for the communication of the ER. The ER Connector interacts with this component using the IEC61850 protocol, enabling the ER management.

2.6 Virtual Meter

This component was not developed in this prototype, but it was necessary to simulate real meter data and provide active and reactive power values to the ER Connector.

3 Installation/Deployment instructions

The SMX was implemented using a BeagleBone Black Industrial[iii], and the ER controller was implemented using a Raspberry PI 3 Model B[iv]. A SMX image needs to be installed on the BeagleBone and updated. The Raspberry PI image used was a Raspbian Jessie version. After these images have been correctly installed and update, it is necessary to deploy the developed software. The SMX is directly connected to the ER Controller, as shown in Figure 3.

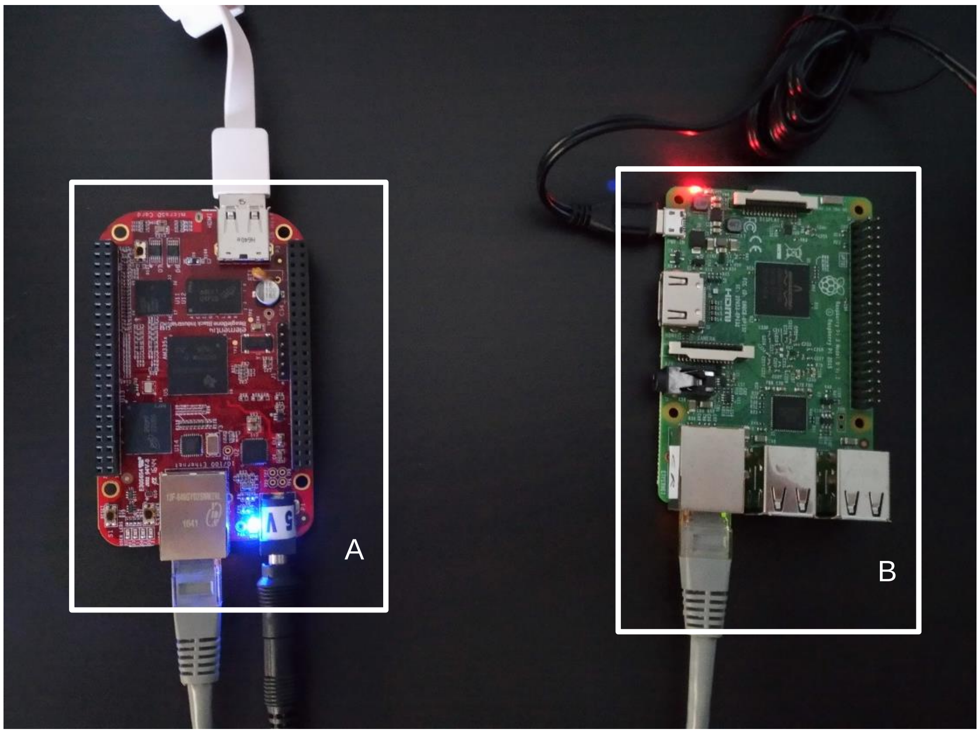

Figure 3. D4.1 prototype architecture (A – SMX; B – ER Controller).

4 Software dependencies and requirements

No special hardware requirements are needed to implement the initial version of the User-side ESS control. It is necessary an USB-Ethernet adapter to enable the connection of the development PC with the SMX hardware, since both BeagleBone and Raspberry PI have only other Ethernet port. This D4.1 prototype has the software dependencies described in Table 1.

Table 1. Software Dependencies

Dependency	License	Role
OpenIEC61850, version 1.4.0	Apache License 2.0	The software is used to exchange date between the ER Connector and the ER, using the IEC61850 protocol.
Raspbian, version "Jessie with Desktop"	GNU General Public License (GPL) 2.0	The operating system of the Raspberry PI, used as ER controller.

5 API Reference

5.1 LESSAg

As previous mentioned, in this D4.1 initial version, the LESSAg receives a command in the JSON format (as described in section 2.1). The API for the LESSAg is

void lEssAgCommand(String command)

5.2 EBs

Both Event Brokers are implemented using a MQTT broker, and use the publish/subscribe method.

5.3 ER Controller

Since the ER Controller uses the IEC61850 and it is implemented using the OpenIEC61850, its API is available online[v].

6 Conclusions

This document presents all the necessary information regarding the D4.1 - "Initial User-side ESS control system" prototype, developed by the Storage4Grid project. An updated version will be presented in M18 and M30. The current developed modules will be extended in order to provide more features, necessary for the Storage4Grid project.

Acronyms

Acronym	Explanation
ACK	Acknowledgment
DSF	Decision Support Framework
EB	Event Broker
ER	Energy Router
ESS	Energy Storage System
EV	Electric Vehicle
GESSCon	Grid ESS Controller
JSON	JavaScript Object Notation
MQTT	Message Queue Telemetry Transport
LESSAg	Local ESS Agent
LEVChAg	Local EV Charging Agent
RES	Renewable Energy Sources
RTDB	Real-time database
S4G	Storage4Grid
SMM	Smart Metrology Meter
SMX	Smart Meter eXtension
UML	Unified Modelling Language
USM	Unbundled Smart Meter
VPN	Virtual Private Network

List of figures

List of tables

References

[i] IEC 61850:2017 SER - Communication networks and systems for power utility automation, https://webstore.iec.ch/publication/6028, Accessed 28 August 2017.
[ii] IEC TR 61850-90-7:2013 - Communication networks and systems for power utility automation - Part 90-7: Object models for power converters in distributed energy resources (DER) systems, https://webstore.iec.ch/publication/6027, Accessed 28 August 2017.
[iii] BeagleBone Black, http://beagleboard.org/black, Accessed 31 July 2017.
[iv] Raspberry PI 3 Model B, https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, Accessed 31 July 2017.
[v] OpenIEC61850 1.4.0 API, https://www.openmuc.org/iec-61850/javadoc/, Accessed 23 August 2017.