*local **E**lectricity retail **M**arkets for **P**rosumer smart grid p**OWER** services*

Deliverable nº:	**D4.3**
Deliverable name:	**Communications Test Plan**
Version:	**1.0**
Release date:	**21/03/2016**
Dissemination level:	**PU**
Status:	**Draft**
Authors:	Schneider Electric - Cristóbal Cordobés Berraquero eSmart - Bjørn Asvard Olsen
Contributors:	CITCEA-UPC - Pau Lloret Gallego

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement No 646476.

Document history:

Version	Date of issue	Content and changes	Edited by
0.1	10/02/2016	Draft version	Cristóbal Cordobés
0.2	02/03/2016	Added information	Pau Lloret
0.3	16/03/2016	Update sections and tests	Bjørn Asvard / Cristóbal Cordobés
0.4	21/03/2016	Complete Version	Cristóbal Cordobés
0.5	22/03/2016	Review version	Mónica Aragués
0.6	30/03/2016	Peer Review version	Stig Ødegaard

Peer reviewed by:

Partner	Contributor
Esmart	Stig Ødegaard

Deliverable beneficiaries:

WP / Task	Responsible
WP3	UPC
WP5	eSmart Systems
WP7	Schneider Electric

Table of contents

Abbreviations and Acronyms

Acronym	Description
CA	Consortium Agreement
DoA	Description of Action (annex I of the Grant Agreement)
EC	European Commission
GA	Grant Agreement
PC-A	Project Coordinator-Administrative
PC-T	Project Coordinator-Technical
PMC	Project Management Committee
PO	Project Officer
QM	Quality Management
TMT	Technical Management Team
ToC	Table of Contents
WP	Work Package
WPL	Work Package Leader

Tables

Executive summary

This document reflects the test plan that the APIs defined in deliverable D4.2 must pass.

For each use case defined in the project, a series of tests have been developed where it will be defined each step taken in the preparation of the test and the expected results. The final results will come reflected on these deliverables, once they have completed the development of the systems involved.

The test and verification results will be documented in the Communication Test Report.

1 Test objectives

The aim of this section is to define the objectives to be achieved during the tests, which are:

- To ensure that the pilot or platform where the EMPOWER architecture is going to be tested, operates appropriately. This means ensuring that all the equipment works as it is expected in stand-alone mode, as well as connected to other devices. The tests must also verify the equipment is able to send and receive the required signals.
- To ensure the metering, communications and control system work suitably.
- To ensure that the market cloud responds as expected and interacts properly with the control and metering cloud.

So, there are two steps in the test performance. One is to check that each element operates as it is desired. The other is to ensure that the system as a whole performs as specified. The entire system is tested in the laboratory and in pilots. In both, the following units or sub-systems can be identified:

- Customer premises
    - Customer appliances
    - Local controller
    - Meter
- Distributed Energy Resources
    - Distributed generation
    - Energy storage
    - EV charging station
    - DER controller
- SCADA+HMI
- SESP platform
    - Control cloud
    - Market cloud
    - Metering cloud

2 Methodology

The methodology adopted for performing the tests is based on first classifying the different tests and later identifying the specific objective of each cluster of tests.

2.1 Tests classification

The tests can be classified according to their purpose, distinguishing in each case the elements or sub-systems they concern. Basically, there will be 3 types of tests:

1. *Standalone tests* – They concern a single sub-system and concentrate on standalone functionality. They will be applied to those sub-systems where the design and/or operation is unique to the system. In our case, at least the SESP and the SCADA fit into this category. However, it may be necessary to test more subsystems as standalone.
2. *Communication tests* – They aim to test and validate the data exchanges between sub-systems, that allow to operate them co-ordinately. A given sub-system may have a number of different interfaces depending on its role in the complete system.
3. *System integration tests* – They validate the whole operation of the system in a laboratory scaled platform and in each pilot site.

2.2 Tests targets identification

The specific targets for each cluster of tests presented in Section 2.1 are detailed below.

2.2.1 Standalone tests

Test name	Targets	Associated task and deliverable
SESP– Standalone tests	Test the performance of the SESP.	Task 4.4 Deliverable 4.5
SCADA– Standalone tests	Test the performance of the SCADA.	Task 4.4 Deliverable 4.5

Table 1.*Stand alone Tests*

2.2.2 Communication tests

Test name	Targets	Associated task and deliverable
SCADA – DER tests	Test the performance of the communications between the SCADA and the DER. This includes: ▪ Establish communications (EV, Solar, Storage) ▪ Read status (EV, Solar, Storage) ▪ Read metered data (EV, Solar, Storage) ▪ Ensure remote commands (EV, Solar, Storage)	Task 4.4 Deliverable 4.5
SCADA – SESP tests	Test the performance of the communications between the SCADA and the SESP. This includes: ▪ Read Periodic data from DER device (EV, Solar, Storage) ▪ Read data on demand from DER device (EV, Solar, Storage) ▪ Send on/off command to DER device (EV, Solar, Storage) ▪ User Validation for EV charging point (EV)	Task 4.4 Deliverable 4.5
Prosumer – SESP tests	Test the performance of the communications between the prosumer and the SESP. This includes: ▪ User Validation ▪ Read Meter values on demand ▪ Read Periodic values	Task 4.4 Deliverable 4.5

Table 2.*Communication Tests*

2.2.3 System integration tests

Test name	Targets	Associated task and deliverable
Laboratory tests	Emulate different scenarios to validate the operation of the whole system in a scaled platform.	Task 5.5 Deliverable 5.5
Pilot tests	Validate the operation of the whole system in each specific pilot site. Tests will be pilot-specific.	Task 7.3 Deliverable 7.3

Table 3.*Integration tests*

3 API Communication Tests

3.1 Validation Template Description

Description of the tests performed on every the scenarios to validate them. The following template will be used for every one of the tests.

Test Name	XXXXXXX
Test Code	*Coding for test type (**EV.XXX** for electric Vehicle, **GE.SL.XXX** for solar generation and **ST.XXX** for Storage)*
Actors	XXXXXXX
Test Description	XXXXXXX
Steps	1. XXXXXXX 2. XXXXXXX
Expected Results	XXXXXXX
Final Results	XXXXXXX

Table 4.*Validation Template*

3.2 DER – SESP Test

3.2.1 Electric Vehicle charging station

All technical information from the charging stations will be sent to the SESP in a periodically manner.

SCADA will initialize the process for reading messages from the DER, and will act as a gateway between all EV charging stations and SESP, and all communication to and from SESP will take place via SCADA.

The following systems and tools will be used when performing the API communication tests:

- Schneider Citect SCADA
- SESP
- Azure Event Hub test tool

3.2.1.1 EV Periodic reading

Test Name	EV Periodic Reading
Test Code	***EV.01***
Actors	Scada System, EV charging station, Modbus controller, SESP
Test Description	This test will verify that the periodic reading request is sent from SCADA to the EV charging station, and that the EV charging station will return the reading message back to SCADA. SCADA will route the message to the SESP. The same test procedure will be used by the following variables: - Active power (charging power) - Reactive power - Voltage - Current - Charging point state
Steps	1. SCADA send a request message to EV charging station. 2. EV charging station will return a Modbus message back to SCADA. 3. SCADA will create a JSON message based on the Modbus message received by EV charging station and send the JSON

	message to SESP (Azure Event Hub).
Step results	1. An ACK message should be returned from the EV charging station. 2. The reading message will be received by SCADA and stored in the SCADA internal database. 3. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Expected Results	The reading value can be read using the SESP client.
Final Results	

Table 5.*EV Periodic test*

3.2.1.2 EV on-demand reading

All technical information from the EV charging stations will normally be sent to the SESP in a periodically manner. In addition, the SESP can request for such information at any time if needed (on-demand). This section describes such an on-demand test scenario.

The following systems and tools will be used when performing the API communication tests:

- Schneider Citect SCADA
- SESP
- Azure Logic App status tool
- Visual Studio (or another Azure Service Bus queue viewer).
- Azure Event Hub test tool

Test Name	**EV on-demand reading**
Test Code	***EV.02***
Actors	Scada System, EV charging station, Modbus controller, SESP
Test Description	This test will verify that the on-demand request is sent from SESP to the EV charging station, and that the EV charging station will return the reading message back to SCADA, who is responsible for transforming and routing the message back to the SESP. The same test procedure will be used by the following variables:

	- Active power (charging power) - Reactive power - Voltage - Current - Charging point state
Steps	1. SESP will send a request message to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the request message to the EV charging station. 4. EV charging station will return a Modbus message back to SCADA. 5. SCADA will create a JSON message based on the Modbus message received by EV charging station and send the JSON message to SESP (Azure Event Hub).
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the EV charging station. 4. The reading message will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Final Results	

Table 6.*EV on demand test*

3.2.1.3 Remote control signals

A “charge ON/OFF command” can be sent from SESP to a EV charging station in relation with a demand response plan.

Test Name	**EV Remote**
Test Code	***EV.03***
Actors	Scada System, EV charging station, Modbus controller, SESP

Test Description	This test will verify that the control signal is sent from SESP to the EV charging station, and that the EV charging station changes its state (e.g. from "charging" to "not charging").
Steps	1. SESP will send a control signal to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the control signal to the EV charging station. 4. EV charging station will change its state and return the response back to SCADA. 5. SCADA will request for a new reading of the EV charging station state, and send this as a JSON message to the SESP.
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the EV charging station. 4. The new EV charging station state will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes, and the new state of the charging station should be available for the SESP operator.
Final Results	

Table 7.*EV control signal test*

3.2.2 Generation resources

3.2.2.1 Periodic Readings

All technical information from the generators and devices installed will normally be sent to the SESP in a periodically manner.

Communications will be checked in order to start operation tests between SCADA and field devices. It is necessary to distinguish between photovoltaic (PV) and wind (WI) generation and establish the correct communication.

Test Name	**Generation Periodic Reading**

Test Code	***GEN.XX.01*** ***(where XX is PV or WI in regards to type of generation)***
Actors	SCADA System, Solar or wind, Generator, Generation Modbus controller, SESP
Test Description	This test will verify that the periodic reading request is sent from SCADA to the Generation Controller / inverter, and that this unit will return the reading message back to SCADA. After that, SCADA will route the message to the SESP. The same test procedure will be used by the following variables: - Active power - Reactive power
Steps	1. SCADA sends a request message to the controller situated in the main switchboard. 2. Controller will return a Modbus message back to SCADA. 3. SCADA will create a JSON message based on the Modbus message received generation controller and send the JSON message to SESP (Azure Event Hub).
Step results	1. An ACK message should be returned from the generation controller. 2. The reading message will be received by SCADA and stored in the SCADA internal database. 3. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Expected Results	The reading value can be read using the SESP client.
Final Results	

Table 8.*Generation periodic test*

3.2.2.2 On-demand readings

All technical information from the generators and devices installed will normally be sent to the SESP in a periodically manner. In addition, the SESP can request for such information at any time if needed (on-demand). This section describes such an on-demand test scenario.

The following systems and tools will be used when performing the API communication tests:

- Schneider Citect SCADA
- SESP
- Azure Logic App status tool
- Visual Studio (or another Azure Service Bus queue viewer).
- Azure Event Hub test tool

Test Name	**Generation on-demand Reading**
Test Code	***GEN.XX.02*** ***(where XX is PV or WI in regards to type of generation)***
Actors	SCADA System, Solar or wind, Generator, Generation Modbus controller, SESP
Test Description	This test will verify that the on-demand request is sent from SESP to the generator controller, and this controller will return the reading message back to SCADA, who is responsible for transforming and routing the message back to the SESP. The same test procedure will be used by the following variables: - Active power - Reactive power
Steps	1. SESP will send a request message to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the request message to the generation controller. 4. Generation controller will return a Modbus message back to SCADA. 5. SCADA will create a JSON message based on the Modbus message received by generation controller and send the JSON message to SESP (Azure Event Hub).
Expected Results	1. It will be possible to see a message on the Azure Service bus queue.

	2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the generation controller. 4. The reading message will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Final Results	

Table 9.*Generation on demand test*

3.2.2.3 Remote control signals

A “generation ON/OFF command” or similar control signal must be sent from SESP to the generation controller in relation with a demand response plan.

Test Name	**Generation Remote Command**
Test Code	***GEN.XX.03*** ***(where XX is PV or WI in regards to type of generation)***
Actors	SCADA System, Solar or wind, Generator, Generation Modbus controller, SESP
Test Description	This test will verify that the control signal is sent from SESP to the generation controller, and that the corresponding generator controller changes its state (e.g. from “ON” to “OFF”).
Steps	1. SESP will send a control signal to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the control signal to the proper generation controller. 4. This controller will change its state and return the response back to SCADA. 5. SCADA will request for a new reading of the generator state, and send this as a JSON message to the SESP.
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App.

	3. An ACK message should be returned from the generation controller. 4. The new generator state will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes, and the new state of the generator should be available for the SESP operator.
Final Results	

Table 10.*Generation control signal test*

3.2.3 Storage

The following systems and tools will be used when performing the API communication tests:

- Schneider Citect SCADA
- SESP
- Azure Logic App status tool
- Visual Studio (or another Azure Service Bus queue viewer).
- Azure Event Hub test tool

3.2.3.1 Periodic Readings

All technical information will normally be sent to the SESP in a periodically manner.

This will be valid for external Storage different from Storage situated in Generation Resources.

Test Name	**Storage Periodic Reading**
Test Code	***STO. 01***
Actors	SCADA System, Storage Battery, Modbus controller, SESP
Test Description	This test will verify that the periodic reading request is sent from SCADA to the Storage Battery controller, and that this controller will return the reading message back to SCADA. After that, SCADA will route the message to the SESP. The same test procedure will be used by the following variables:

	- Charging current - Discharging current - State of charge
Steps	1. SCADA send a request message to the controller. 2. Controller will return a Modbus message back to SCADA. 3. SCADA will create a JSON message based on the Modbus message received by the controller and send the JSON message to SESP (Azure Event Hub).
Step results	1. An ACK message should be returned from the controller. 2. The reading message will be received by SCADA and stored in the SCADA internal database. 3. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Expected Results	The reading value can be read using the SESP client.
Final Results	

Table 11.*Storage periodic readings test*

3.2.3.2 On-demand readings

All technical information from the storage controllers will normally be sent to the SESP in a periodically manner. In addition, the SESP can request for such information at any time if needed (on-demand). This section describes such an on-demand test scenario.

Test Name	**Storage on-demand Reading**
Test Code	***STO.02***
Actors	SCADA System, Storage Battery, Modbus controller, SESP
Test Description	This test will verify that the on-demand request is sent from SESP to the battery controller, and this controller will return the reading message back to SCADA, who is responsible for transforming and routing the message back to the SESP. The same test procedure will be used by the following variables: - Active power

	- Reactive power
Steps	1. SESP will send a request message to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the request message to the Storage battery controller. 4. The controller will return a Modbus message back to SCADA. 5. SCADA will create a JSON message based on the Modbus message received by the controller and send the JSON message to SESP (Azure Event Hub).
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the controller. 4. The reading message will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Final Results	

Table 12.*Storage on demand test*

3.2.3.3 Remote control signals

A “charge ON/OFF command” can be sent from SESP to the battery controller in relation with a demand response plan.

Test Name	**STO Remote control signal**
Test Code	***STO.03***
Actors	SCADA System, Storage Battery, Modbus controller, SESP
Test Description	This test will verify that the control signal is sent from SESP to the Storage battery controller, and that battery state has already changed (e.g. from “charging” to “not charging”).
Steps	1. SESP will send a control signal to an Azure service bus queue.

	2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the control signal to controller. 4. The controller will change its state and return the response back to SCADA. 5. SCADA will request for a new reading of the controller state, and send this as a JSON message to the SESP.
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the controller. 4. The state will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes, and the new state of the battery should be available for the SESP operator.
Final Results	

Table 13.*Storage remote control signal test*

3.2.4 Load Resources

The following systems and tools will be used when performing the API communication tests:

- Schneider Citect SCADA
- SESP
- Azure Logic App status tool
- Visual Studio (or another Azure Service Bus queue viewer).
- Azure Event Hub test tool

3.2.4.1 Periodic Readings

All technical information from load resources controllers and devices installed will normally be sent to the SESP in a periodically manner.

Communications will be checked in order to start operation tests between SCADA and field devices.

Test Name	Load Resourcing Periodic Reading
Test Code	***LR.01***
Actors	SCADA System, Load Resource Modbus controller, SESP
Test Description	This test will verify that the periodic reading request is sent from SCADA to the load resource controller, and that this controller will return the reading message back to SCADA. After that, SCADA will route the message to the SESP. The same test procedure will be used by the following variables: - Active power - Reactive power
Steps	1. SCADA sends a request message to the controller. 2. Controller will return a Modbus message back to SCADA. 3. SCADA will create a JSON message based on the Modbus message received generation controller and send the JSON message to SESP (Azure Event Hub).
Step results	1. An ACK message should be returned from the controller. 2. The reading message will be received by SCADA and stored in the SCADA internal database. 3. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Expected Results	The reading value can be read using the SESP client.
Final Results	

Table 14.*Load Resource periodic reading test*

3.2.4.2 On-demand readings

All technical information from the generators and devices installed will normally be sent to the SESP in a periodically manner. In addition, the SESP can request for such information at any time if needed (on-demand). This section describes such an on-demand test scenario.

Test Name	Load Resources on Demand Reading

Test Code	***LR.02***
Actors	SCADA System, Load Resource Modbus controller, SESP
Test Description	This test will verify that the on-demand request is sent from SESP to the controller, and this controller will return the reading message back to SCADA, who is responsible for transforming and routing the message back to the SESP. The same test procedure will be used by the following variables: - Active power - Reactive power
Steps	1. SESP will send a request message to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the request message to the controller. 4. Load resource controller will return a Modbus message back to SCADA. 5. SCADA will create a JSON message based on the Modbus message received by the controller and send the JSON message to SESP (Azure Event Hub).
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the controller. 4. The reading message will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes.
Final Results	

Table 15.*Load Resource on demand test*

3.2.4.3 Remote control signals

The SESP will have the possibility to send messages to a load resource. Either types of messages are a request for variables, or control signals for demand response (power off / power on). A “load resource ON/OFF command” or similar control signal must be sent from SESP to the load resource controller in combination with a demand response plan.

Test Name	**Load Resources Remote control signals**
Test Code	***LR.03***
Actors	SCADA System, Load Resource Modbus controller, SESP
Test Description	This test will verify that the control signal is sent from SESP to the generation controller, and that the corresponding controller change its state (e.g. from “ON” to “OFF”).
Steps	1. SESP will send a control signal to an Azure service bus queue. 2. A dedicated Azure Logic App will pick up the message and send it to the SCADA web service. 3. SCADA transforms and send the control signal to the proper controller. 4. This controller will change its state and return the response back to SCADA. 5. SCADA will request for a new reading of the generator state, and send this as a JSON message to the SESP.
Expected Results	1. It will be possible to see a message on the Azure Service bus queue. 2. A positive response from the SCADA REST API to the Azure Logic App. 3. An ACK message should be returned from the controller. 4. The new generator state will be received by SCADA and stored in the SCADA internal database. 5. The JSON message is received in Azure Event Hub and picked up by the SESP backend processes, and the new state of the Load Resource should be available for the SESP operator.
Final Results	

Table 16.*Load Resource control signal test*

4 Conclusions

This document defines a list of tests that must be complied in order to approve the final solution of communication systems.

5 References

- Empower Project - Deliverable D4.2: Functional and technical documentation of relevant API-functions
- Azure Event Hub (azure.microsoft.com)
- Azure IoT Hub (azure.microsoft.com)
- Protocol Specification for Modbus exchanges between external system and charging Station – www.Schneider-electric.com
- Technical Note – unSpec Logging in SolarEdge Inverters (Susnpec Alliance) - www.sunspec.org/