Small or medium-scale focused research project (STREP)

Energy Demand-Aware Open Services for Smart Grid Intelligent Automation

SmartHG

EU FP7 Project #317761

Deliverable D3.3.1

Third Year Design of Home Intelligent Automation Services

Deliverable due on : M36
Output of WP : WP3
WP Responsible : HMTI

Consortium

Participant Organization Name	Participant Short Name	Country
Sapienza University of Rome	UNIROMA1	Italy
Aarhus University	AU	Denmark
IMDEA Energía	IMDEA	Spain
A. V. Luikov Heat and Mass Transfer Institute of the National Academy of Sciences of Belarus	HMTI	Belarus
ATANVO GmbH	ATANVO	Germany
Panoramic Power	PANPOW	Israel
Solintel	SOLINTEL	Spain
SEAS – NVE	SEAS	Denmark
Kalundborg Municipality	KAL	Denmark
Minskenergo	MINSKENG	Belarus
Develco Products A/S	DEVELCO	Denmark

Document Information

Version	November 17, 2015, 23:05
Date	November 17, 2015
Contributors	UNIROMA1, AU, IMDEA, HMTI, ATANVO, PANPOW, DEVELCO
Reason for release	Third year review
Dissemination level	Public (PU)
Status	Final

Project title	Energy Demand-Aware Open Services for Smart Grid Intelligent Automation
Project acronym	SmartHG
Project number	317761
Call (part) identifier	FP7-ICT-2011-8

Work programme topic addressed	
Challenge	**6**: *ICT for a low carbon economy*
Objective	**ICT-2011.6.1** *Smart Energy Grids*
Target Outcome	**d**) Home energy controlling hubs that will collect real-time or near real-time data on energy consumption data from smart household appliances and enable intelligent automation.

Project coordinator	Enrico Tronci
E-mail	`tronci@di.uniroma1.it`

Contents

List of Acronyms

API Application Programming Interface

COP Coefficient of Performance

DAPP Demand Aware Price Policies

DB&A Database and Analytics

DBService Database Service

DSO Distribution System Operator

EBR Energy Bill Reduction

EDN Electric Distribution Network

ESS Energy Storage System

EUMF Energy Usage Modelling and Forecasting

EUMF-H Energy Usage Modelling and Forecasting for Homes

EUMF-K Energy Usage Modelling and Forecasting for Control

EUR Energy Usage Reduction

EUR-H Energy Usage Reduction for Homes

EUR-K Energy Usage Reduction for Control

GIAS Grid Intelligent Automation Service

HAN Home Area Network

HECH Home Energy Controlling Hub

HIAS Home Intelligent Automation Service

IAS Intelligent Automation Service

IBR Inclining Block Rate

JSON JavaScript Object Notation

JWT JSON Web Token

MILP Mixed-Integer Linear Programming

PEV Plug-in Electric Vehicle

TLS Transport Layer Security

T&D Transmission and Distribution

ToU Time of Usage

UC Use Case

UML Unified Modeling Language

HMAC Hash-based Message Authentication Code

JOSE Javascript Object Signing and Encryption

JWS JSON Web Signature

JWT JSON Web Token

JWE JSON Web Encryption

JWA JSON Web Algorithm

JWK JSON Web Key

dbservice Database Service

smcservice SMC Service

OAuth2 OAuth 2.0

UC Use Case

URI Uniform Resource Identifier

ORM Object-Relation Mapping

TLS Transport Layer Security

SmartAMM Smart Automatic Meter Management

CSP Communication Service Provider

ESP Energy Service Portal

Executive Summary

Retrospect During the second year, all the Home Intelligent Automation Services (HIASs) underwent a complete re-design. The Energy Usage Modelling and Forecasting (EUMF) and Energy Usage Reduction (EUR) services have been split in two different services to be used in two different scenarios. Finally, communication protocols used by smart meters inside homes make sensor measurements available to Intelligent Automation Services (IASs).

Present Achievements The third year version of the HIASs consisted in one last redesign for the Energy Bill Reduction (EBR) service and for the open communication protocols. On the other hand, the second year versions of the other HIASs (namely, Energy Usage Reduction for Control (EUR-K), Energy Usage Reduction for Homes (EUR-H), Energy Usage Modelling and Forecasting for Homes (EUMF-H) and Energy Usage Modelling and Forecasting for Control (EUMF-K)) resulted to be satisfactory, thus only a few improvements (mainly in the Web interfaces) were carried out.

As for the EBR service, in this last year version it directly aims at minimising the overall costs for the Distribution System Operator (DSO)/energy retailer. Namely, the costs which are minimised by EBR are: hardware (Energy Storage System (ESS) to be installed on each home), software (EBR instances to be installed on each home), energy, Transmission and Distribution (T&D) and CO_2 emissions costs. This is obtained by suitably driving the charge/discharge of an ESS (and possibly of a Plug-in Electric Vehicle (PEV) being plugged at home). As for distribution costs reduction, the EBR works in synergy with the Demand Aware Price Policies (DAPP) service (see Deliverable D4.3.1).

Regarding the design of open protocol for IASs, the focus was on easing the integration between IASs in the SmartHG ecosystem. The SMC Service (smcservice) has been extended to embrace all IASs, which required the Use Case (UC) diagram in the iteration of the previous year to be updated. This has resulted in a data model of the smcservice and a design of a token structure.

Impact

As it is shown in Deliverable D5.3.1, the final version of EBR (also working in synergy with the Grid Intelligent Automation Services (GIASs) and especially with DAPP) fulfils the requirements of energy bill reduction, by allowing a meaningful net saving for the DSO/retailer (able to also cover all hardware, software, energy, T&D and CO_2 emissions costs).

The design of the open protocol for home devices enables seamless communication between smart home appliances and the Database Service (DBService). Moreover, the design of the open protocol allows the developers to upgrade remotely the software of the

Home Energy Controlling Hub (HECH). A infrastructure platform based RESTful principles and a infrastructure service for enabling web-based smart grid services to interact securely and with consent from the residential consumer and DSO.

Chapter 1

Retrospect

In this chapter we briefly recall the main achievements obtained in the second year version of the SmartHG Home Intelligent Automation Services (HIASs) design, which was described in Deliverable D3.2.1. The activities in the second year of the project resulted in a complete re-design of all HIASs.

The Energy Usage Modelling and Forecasting (EUMF) service has been split in two different services to be used in two different scenarios: EUMF-H is used to offer a medium-term (many days) energy demand forecasting service to residential users, EUMF-K is used to offer a fast, short term (a few hours) energy demand forecasting to other Intelligent Automation Services (IASs). Also the Energy Usage Reduction (EUR) service has been split in two different services: EUR-H is used to offer an energy usage visualization service, whilst EUR-K is used to support residential users in evaluating trade-offs between home retrofit options (e.g., increase thermal insulation or increase heat pump Coefficient of Performance (COP)) aiming at improving home energy efficiency. The Energy Bill Reduction (EBR) service has been re-designed to directly control selected home appliances (namely, an Energy Storage System (ESS) and a Plug-in Electric Vehicle (PEV)), so as to be transparent to residential users. Finally, communication protocols used by smart meters inside homes make sensor measurements available to IASs. Such protocols had to be completely re-designed with respect to their first year version, as a new SmartHG partner is providing home devices different from the ones planned in the first year.

Chapter 2

Introduction

Work Package 3 (WP3) is devoted to the design of the SmartHG Home Intelligent Automation Services (HIASs). The goal for such services is to reduce energy costs and consumption at the residential level. To this aim, they require as input data coming from Grid Intelligent Automation Services (GIASs) output, residential users power consumption, homes geographic location, homes-related information, etc. As an output, their goal is to provide either a technology to directly control home appliances, or information to assist residential consumers to use electricity in an effective way. Furthermore, such goals must be achieved while preserving consumer privacy and confidentiality. From a functional point of view, HIAS control loops are the inner loops of the overall SmartHG functional schema, see highlighted part of Figure 2.1. From an architectural point of view, HIASs are those highlighted in the overall architectural schema of Figure 2.2. This deliverable illustrates the design of HIASs and the communication protocol between them and the home devices. The prototypes built basing on such designs are then used for the evaluation phase described in D5.3.1. In the following, we briefly introduce each of the HIASs which we re-designed in the third and final year, namely Energy Bill Reduction (EBR) and communication protocols. As for the other HIASs, we have the following. The second year versions of Energy Usage Reduction for Control (EUR-K) and Energy Usage Modelling and Forecasting for Control (EUMF-K) resulted to be satisfactory,thus no more work is needed. As for Energy Usage Reduction for Homes (EUR-H) and Energy Usage Modelling and Forecasting for Homes (EUMF-H), they were completed by fixing missing parts in the corresponding Web services (see Deliverable D3.3.2).

The EBR service aims at supporting residential users in *saving on their energy bill*. In this last year version, this goal is achieved as follows. Residential users have to pay a very low fixed (say monthly) tariff to participate to SmartHG Platform. Such tariff will be very low compared to their current average energy bill, and here we will conservatively suppose it is 0 EUR. Instead, the saving w.r.t. the overall costs (hardware, software, energy, Transmission and Distribution (T&D), CO_2 emissions) is achieved by the Distribution System Operator (DSO)/energy retailer, by installing the EBR service on each home, together with an Energy Storage System (ESS). By suitably driving the charge/discharge of such an ESS (and possibly of a Plug-in Electric Vehicle (PEV) being plugged at home), EBR is able of minimising the costs stemming from energy usage and CO_2 emissions. As for T&D costs reduction, the EBR works in synergy with the Demand Aware Price Policies (DAPP) service (see Deliverable D4.3.1): it receives in input the individualised power profile P output by DAPP and also minimises a "fine" coming from going outside P. In this way, both the DSO and the energy retailer may obtain a saving: the DSO

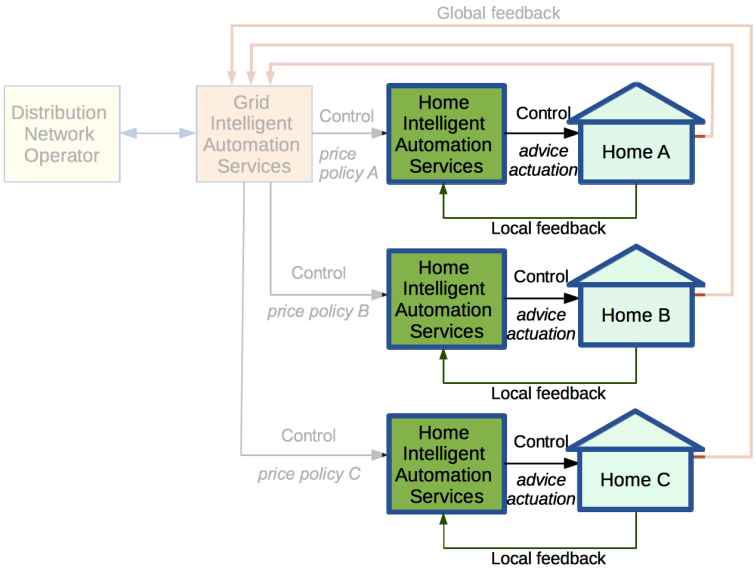

Figure 2.1: Functional schema of SmartHG HIASs.

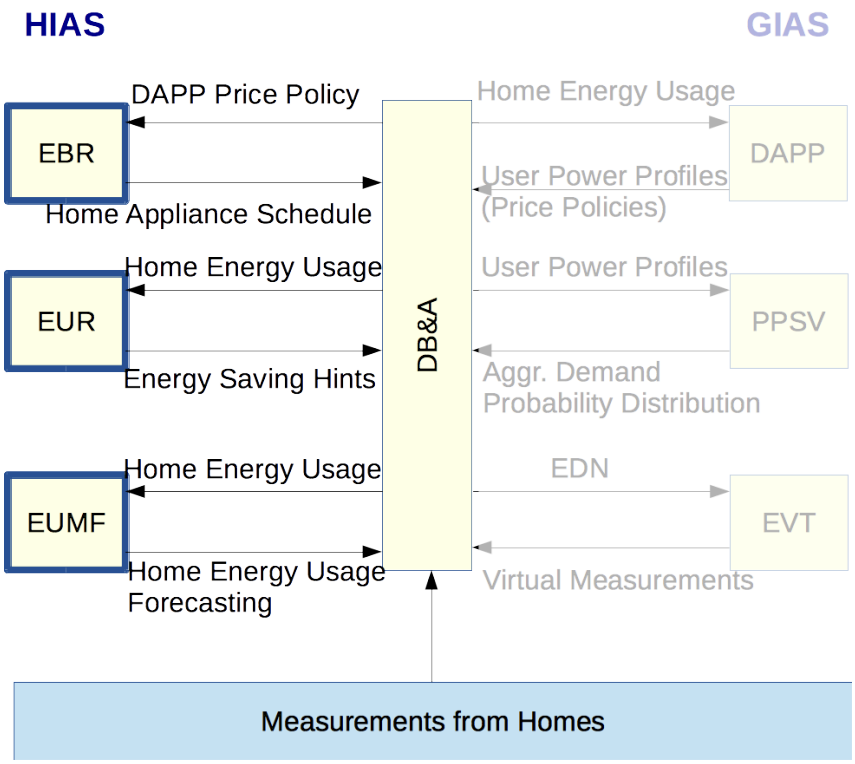

Figure 2.2: SmartHG HIASs architecture.

by T&D investment deferral (if EBR keeps the demand inside the power profiles output by DAPP, Electric Distribution Network (EDN) substations are less stressed and last longer) and the energy retailer by arbitrage and CO_2 emissions reduction. Finally, as for hardware costs, these consists in the smart meters (which are a fixed cost) and in the ESS costs. In order to choose the fittest characteristics of the ESS of each home (depending on the historical power profile of each home on a sufficiently long period, e.g., one year), a suitably modified version of EBR may be used to directly output, for each home h, the suitable capacity and power rate for the ESS to be installed on h.

Regarding the design of open protocol for home devices, the Home Area Network (HAN) is a dedicated network connecting home devices. In the SmartHG project, HAN devices are communicating with the Intelligent Automation Services (IASs) via DEVELCO home automation middleware called Smart Automatic Meter Management (SmartAMM). The SmartAMM Communication System consists of two core components: the Communication Service Provider (CSP) and the SmartAMM Application Programming Interface (API). The CSP handles the communication with the remote SmartAMM gateways, and the API library is used by both the CSP and optionally systems wishing to utilize the CSP.

We have implemented a customized SmartAMM server backend to handle the metering home devices' telegrams and post them to the Database and Analytics (DB&A).

2.1 Motivation

The main motivations for developing final versions of HIASs are the following.

- The ever increasing complexity of both energy tariffs and home appliances fine tuning often prevents residential users to be able to save in the electrical energy bill, by actually using energy when it costs less. This sets the urge for automated methodologies and technologies able to lower down users energy bill.
- Home appliances complexity also prevents residential users to actually use energy in a responsible and optimised way. This sets the urge for automated methodologies and technologies able to assist residential users in this task.
- Such services require in input a forecast for the home power demand in the next few hours, in order to perform their tasks in an optimised way.
- To connect the smart main meters to provide data to the Database Service (DBService).
- To interact remotely with the Home Energy Controlling Hub (HECH).
- To individualised the composition of the services for the particular customer such that the customer can pick out the wanted services as needed.
- To allow the IAS developers to interface to other services as time goes by, hence a transparent change requires a standardised way of communicating.

2.2 Objectives

With respect to the motivations in Section 2.1, the main goals for final versions of HIASs are the following.

- To develop effective and efficient services to aid residential users in reducing their electricity bill.
- To develop effective and efficient services to aid residential users in saving electric energy.
- To develop effective and efficient services providing a forecast for the home power demand in the next few hours, in order to aid the other services in performing their tasks in an optimised way. Such a forecast should be invisible to the user who will only experience our SmartHG Platform benefits.
- To develop effective and efficient communication protocols between all home smart meters and IASs.
- To update the HECH remotely.
- To ease the integration between IASs and securely exchange information among services under the resource owner's control.
- To use RESTful principles to achieve this, where the communication is based on HTTP communication protocol.
- To exchange information securely in the SmartHG ecosystem is an underlying requirement, thus the design of the communication between IASs has to add security per default.

2.3 Achievements

All services have been finalised as follows.

Energy Bill Reduction (EBR). SmartHG Home Services have been finalised by gluing together EBR, EUR-K, and EUMF-K. They work together with SmartHG Grid Intelligent Automation Services (GIASs) (Grid Services) in order to minimise the energy and CO_2 costs for the energy retailer and the distribution costs for the DSO, while trying to satisfy the power demand coming from all home appliances. Moreover, it is also possible to choose the fittest capacity and power rate for the ESSs to be installed on each home, in order to minimise hardware costs for the energy retailer.

The EBR service directly drives selected home appliances, namely ESSs and PEVs, acting as a controller for them. Limitations of previous year prototypes have been overcome by taking adequate solutions, described in Section 3. The final release of EBR implements more precise models for both PEV and ESS. More in detail, not only ESSs but also PEVs can be used also for satisfying other house loads, as current PEVs available on the market allows this behaviour. More details on final EBR design are given in Section 3.

In order to have more precise house parameters identification and house demand forecast, we have improved the usage of EUR-K and EUMF-K services by EBR. In their final releases, EUR-K and EUMF-K are functional to the EBR service, while EUR-H and EUMF-H are left as a support tool for residential users. Final versions of these services are essentially the same as second year versions, with improvements for integration with

Table 2.1: Mapping between SmartHG tasks inside WP3 and chapters of this deliverable

Task	Task Name	Chapters
T3.1	Design and Development of Open Standard Internet-based communication between Home Devices and IASs	Chapter 4
T3.2	Design and Development of Open Standard Internet-based communication between IASs	Chapter 5
T3.3	Design and Development of home EUMF service	see Annex of Deliverable D3.3.1
T3.4	Design and Development of home EBR service	Chapter 3
T3.5	Design and Development of home EUR service	see Annex of Deliverable D3.3.1

EBR and for the amount of data considered for forecasting, gathered from SmartHG test-beds. For this reason, we do not need to go into detail for the design of Energy Usage Reduction (EUR) and Energy Usage Modelling and Forecasting (EUMF) services.

Design of open protocol for home devices. The main achievements are the following.

- **Re-design of Communication Protocols**: The communication protocols between home devices and DBService have been re-designed in order to integrate the residential main smart meters.
- **Re-design of the HECH**: A new protocol has been developed to connect remotely with the HECH.

The design of protocol for IASs. It takes advantage of the HTTP [1] and OAuth 2.0 [2] as stated in the previous year's deliverable. The focus of this year's iteration has been on easing the integration between IASs in the SmartHG ecosystem. The concept of the SMC Service (smcservice) is extended to be embraced by all IASs. This year's achievements therefore includes extended Use Cases (UCs) for the smcservice which incorporates the HECH and IASs developers, as well as the DSO and residential consumer with OAuth 2.0 (OAuth2) terminology [2]. Using the UCs, the data model for the smcservice is designed. The report illustrates the inclusion of the OAuth2 framework through a concept sequence diagram. Furthermore, it shows the construction and exchange of the web tokens using the JWT RFC [3].

2.4 Outline

This deliverable is organised as follows. Chapters 3, 4 and 5 describe the advances in the design of EBR, the open protocol for home devices and the open protocol for IASs, respectively. The overall results of this deliverable are summarised in Chapter 6. Moreover, Section 6 describes in detail the advance of third year year in GIASs design and discusses their impact on the global project objectives. Finally, Table 2.1 shows the correspondence between SmartHG tasks inside WP3 and sections of this deliverable.

Chapter 3

EBR Service Design Description

3.1 EBR Overview

The main goal of the Energy Bill Reduction (EBR) service is to minimise the energy bill of a given home. To this aim, we design the EBR service to be a controller for an Energy Storage System (ESS) and/or a Plug-in Electric Vehicle (PEV) installed in the home. The control signals issued by EBR consists in charge/discharge commands for both the ESS (at any time) and the PEV (when actually plugged). In order to minimise the energy bill, EBR read in input the overall power demand from the home, the current state of charge of the ESS and/or of the PEV, and the costs of energy. Since we want to minimise the costs for the energy retailer (which have to install the ESSs in the residential homes), such costs are split in the energy costs from the day-ahead energy market and in the costs due to CO_2 emissions. When EBR is used in conjunction with Demand Aware Price Policies (DAPP), the costs to be minimised also encompass a penalty for when the overall power demand is outside the power profiles output by DAPP.

3.2 EBR Specification

In this section we describe the input-output behaviour of EBR. Details for the actual algorithms are given in the Annex of D3.3.1.

We first define the notation we use (which is the same used in Deliverable D4.3.1). T is a finite set of contiguous time-slots, all having the same duration. A *power profile* is a function $P : T \to \mathbb{R}$ (being $\mathbb{R}$ the set of real numbers). A *power profile tube* (or *region*) is a pair of power profiles (P_l, P_h) defined over the same set of time-slots T, such that $P_l(t) \leq P_h(t)$ for all $t \in T$. A power profile P *follows* a power profile tube (P_l, P_h) if and only if $P_l(t) \leq P(t) \leq P_h(t)$ for all $t \in T$.

3.2.1 EBR Input and Output

Here we describe in detail input and output for EBR (for a high-level view, see Figure 3.1).

EBR consists on an infinite loop, in which, for each time-slot t (one hour in our experiments), the following input is read:

- the state of charge (in kWh) of the ESS;
- the state of charge (in kWh) of the PEV (0 if the PEV is not plugged in t);

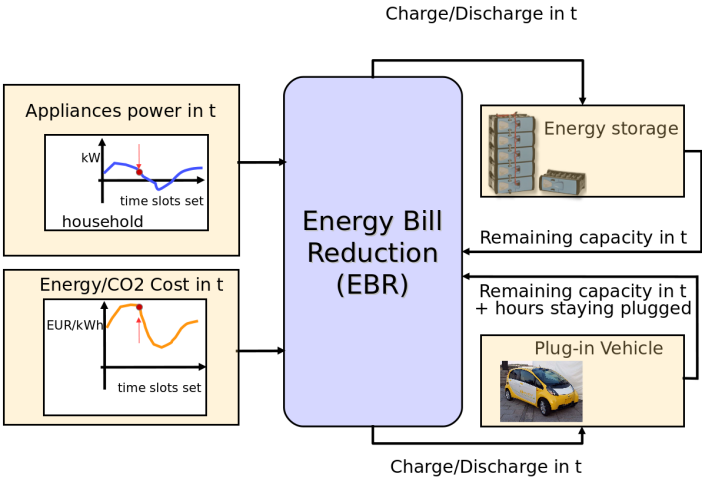

Figure 3.1: EBR input and output (inside the control loop).

- if the PEV has been just plugged, the number of hours it will stay plugged;
- the overall power demand (in kW) coming from all other home appliances;
- the energy and CO_2 costs for the retailer (in EUR/kWh);
- the lower and upper bounds to be enforced in t (in kW), from the DAPP output power profile for the current home.

Note that the number of hours the PEV will stay plugged must be provided by the user when he/she plugs the PEV, whilst all other inputs are automatically read from sensors.

Furthermore, EBR reads the following as configuration inputs (i.e., this information is read once and for all, instead of on each time-slot t):

- the capacity of the ESS and of the PEV (in kWh);
- the minimum and maximum allowed power rate of the ESS and of the PEV (in kW);
- the round-trip efficiency for both the ESS and the PEV;
- the minimum and maximum allowed power demand (in kW), from the home energy contract.

Finally, The output of EBR is a charge/discharge command (in kW) for the ESS and a charge/discharge command for the PEV (when it is plugged).

3.2.2 EBR Algorithm

The input-output behaviour described in Section 3.2.1 is achieved by setting up, for each time t, a Mixed-Integer Linear Programming (MILP) problem. Such MILP problem is based on a forecast of the home appliances demand in the a number N of future time-slots after t. Such forecast is computed using the Energy Usage Modelling and Forecasting for Control (EUMF-K) service, basing on the appliances demand recorded in the last M time-slots. Thus, the MILP problem is defined so as to minimise both the energy and CO_2

costs for the energy retailer plus distribution costs (keeping into account that energy going outside the DAPP power profile must be payed more) under the following constraints, for each of the N future time-slots (including t):

- the PEV may be charged or discharged only when present;
- the ESS [PEV] state of charge at the next time-slot is obtained by applying a charge/discharge action in the current time-slot to the ESS [PEV] capacity at the current time-slot, by also considering the round-trip efficiency (the starting ESS [PEV] state of charge at time-slot t is the one read from ESS [PEV] sensors);
- the resulting ESS [PEV] state of charge must always be within 0 and the ESS [PEV] capacity;
- the power actually required by the home results from the appliances demand plus the charge/discharge action for the ESS [PEV] and the power production from renewable sources (to be subtracted);
- the power computed in the previous point must be below the maximum contractual power of the home.

Once the MILP problem has been created, it is solved by means of a MILP solver (either CPLEX of GLPK). Finally, since the required actions at time t for the ESS and the PEV are decision variables in the MILP problem, the value for such actions is extracted from the solution of the MILP problem returned by the MILP solver.

3.2.3 EBR for Retailer: Battery Dimensioning

We note that EBR may be easily modified in order to aid the energy retailer the fittest dimensions (capacity and power rate) for the batteries to be installed in each home. To this aim, it is sufficient to feed EBR with the whole power demand (including also the PEV historical demand for re-charging) for a sufficiently log time span (e.g. one year), plus two additional inputs:

- the cost of ESS capacity (in EUR/kWh);
- the cost of ESS power rate (in EUR/kW).

Furthermore, the input capacity and power rate for the ESS is not needed any more. Instead, this will be the output of this version of EBR.

3.2.4 EBR Scenarios

We note that EBR may be used under different *scenarios*, depending on the following:

BU : only the ESS is controlled. To this aim, it is sufficient to customise EBR passing 0 as PEV capacity.

BC : both ESS and PEV are controlled, but the PEV may only be charged. To this aim, it is sufficient to customise EBR passing 0 as PEV minimum power rate.

BD : full control for both ESS and PEV.

PC : only the PEV is controlled, and may only be charged. To this aim, it is sufficient to customise EBR passing 0 as ESS capacity and 0 as PEV minimum power rate.

PD : only the PEV is controlled, and may also be discharged. To this aim, it is sufficient to customise EBR passing 0 as ESS capacity.

Finally, different versions of EBR may be obtained by considering the following objective functions to be minimised in the MILP:

- Cost of the ESS plus cost of energy and CO_2. This is the modality described in Sect. 3.2.3, used by the retailer to compute the fittest ESS dimensioning for each home.
- Energy and CO_2 cost. This is the default modality when DAPP is not used (e.g. because peak shaving is not a problem on the given Electric Distribution Network (EDN)).
- CO_2 cost only. This is useful to get an upper bound to the possible reduction of the costs stemming from CO_2 emissions.
- Energy and CO_2 cost, plus distribution costs (keeping into account that energy going outside the DAPP power profile must be payed more). This is the default modality when DAPP is used.

Chapter 4

Design of Open Protocol for Home Devices

4.1 HECH design

This section describes the technologies which have been used to design the open protocol for home deceives to integrate the main meter electricity data with the Database Service (DBService) and to connect remotely with the Home Energy Controlling Hub (HECH).

4.1.1 DEVELCO ZigBee MEP card for electricity meters

The ZigBee module serves as a Smart Energy gateway between the wireless ZigBee network and the Echelon Electricity Meter Power line communication. The MEP device (see Figure 4.1) is capable of collecting meter readings from up to 16 different meter units in one system. The meter units can e.g. be different kinds of meters like water, gas and heating connected to the wireless Zigbee network established and managed by the MEP device. The ZigBee module is based on a DevCom 04HP module with a microcontroller and a low power radio from Freescales Flexis series. Furthermore, a data flash for firmware update over the air and data storage is also included. Communication between meter and ZigBee unit is implemented by a MEP interface. The MEP module is mounted under the standard meter cover and is compatible with both single and poly phase meters.

Figure 4.1: DEVELCO MEP card for electricity meters

4.1.1.1 Mirroring ZigBee meters

Mirroring ZigBee meters in a SE network is done using dynamic end point. The dynamic endpoints support enables the creation of endpoints inside an Energy Service Provider device to support the attributes data availability of a sleeping metering device during its

low power mode. The dynamic endpoint supports the use of the ZigBee Cluster Library simple meter optional commands Create Mirror and Remove Mirror requests and its responses. A dynamic endpoints application includes:

- Mirror Endpoint capability discovery process
- Mirror Creation Request
- Mirror Initialization through Write Attributes Command
- Mirror Updates through ZCL Reporting
- Mirror Remove Request

The discovery process is initiated by the ZigBee Meter Device. The process to internally create or destroy a mirror, and the mirror capability discovery process will be described with more detail in the subsequent sections. The following figure shows a ZigBee Meter Device connected to a Mirror endpoint in the Energy Service Portal (ESP). During the connection time, the Meter Device will update its attributes values into the ESP every time it wakes up (ZCL Reporting). If any other devices need the Meter information it can be obtained from the Mirror endpoint in the ESP which is not a low power device.

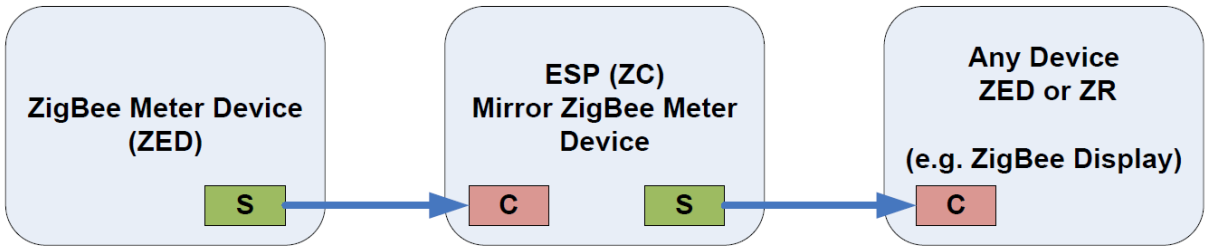

Figure 4.2: Mirror End Point Example

4.1.1.2 Mirror Discovery Process

In order to create a mirror it is necessary to find a device with mirror capability and the endpoint where the SE Metering cluster is implemented. The ZigBee meter has to support a discovery process to find the ESP mirror end point. Typically this discovery task is performed as described below. 1) Broadcast Match Descriptor Request to find Basic cluster
2) Collect Match Descriptor Responses
3) Unicast request to each short address in the list of Match Descriptor Responses to Read Physical Environment Attribute until a device with mirror capability is found.
4) Unicast Match Descriptor Request to find SE Metering cluster
5) Upon Match Descriptor Response request mirror endpoint creation to the SE Metering client cluster.
The sequence diagram in Figure 4.3 shows the discovery process in detail.

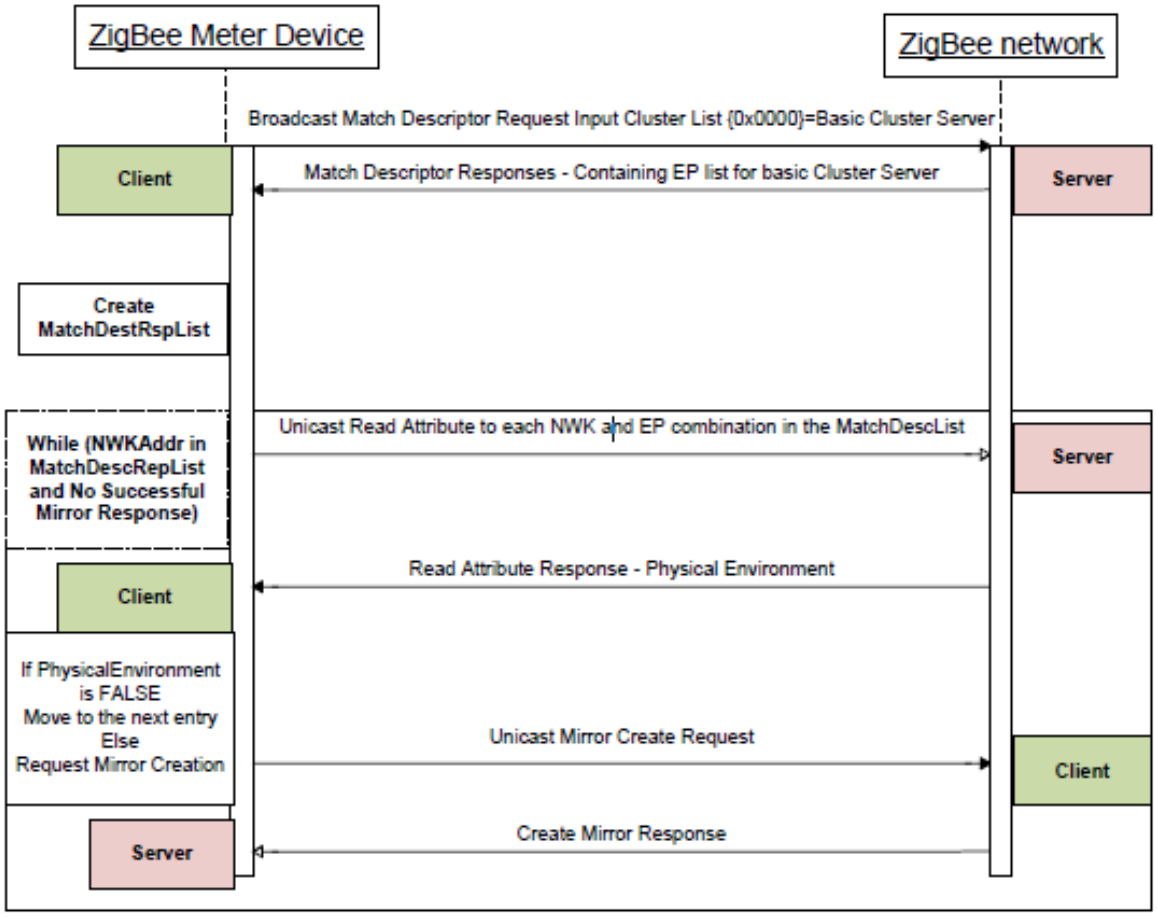

Figure 4.3: Mirror Discovery Process Sequence diagram

4.1.1.3 Echelon Meter MEP interface

The MEP port has 12 uplink tables of 509 bytes (MT45), and 5 downlink tables of 24 bytes (MT15). These tables are utilized for transferring commands and data between the Echelon NES System and the MEP card. The following sections describe the protocol for supporting the communication via the MEP port.

4.1.1.4 MEP Port Communication Protocol

The flow of data between the server and the ZigBee Gateway is typically in the form of request/response messages, though some messages from the Gateway to the server are unsolicited such as periodic meter reports (if enabled). To minimize the time to wait for a response to a request the Urgent Data (On-Demand) transfer model should be used by the MEP communication protocol (the MEP device has only control over the Uplink path. Thus, Urgent Data can only be guaranteed in that direction).

4.1.2 DEVELCO SmartAMM Server

The SmartAMM server consist of 3 parts “Core”, ”Back ends”, ”Network Dispatchers”. Figure 4.4 shows the structure of the server.

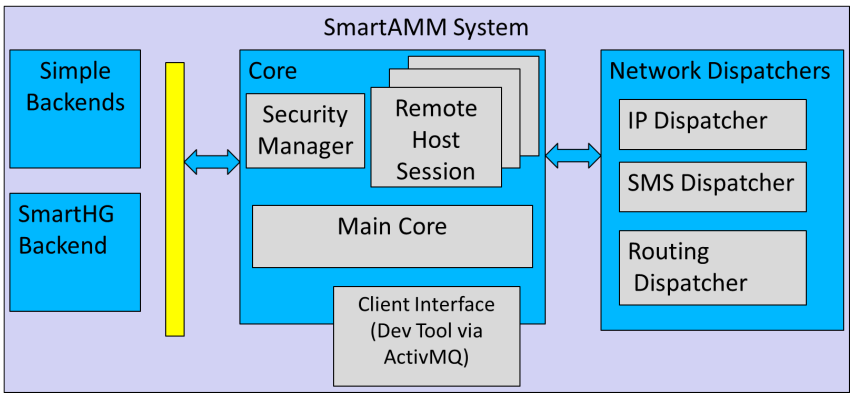

Figure 4.4: The structure of SmartAMM server

4.1.2.1 Network Dispatcher

The network dispatcher contains of 3 different dispatchers "IP", "SMS" and "Routing".

- IP dispatcher that handles TCP/IP connections with all the GW in the network. Informs the core that it has a connection with a GW. Forwards events from the different GW and decode the header of the incoming SmartAMM messages.
- SMS dispatcher handles 2-ways communication with the GW's via SMS communication. This is normally used to configuration of a GSM GW.
- Routing dispatcher handling communication to the new Squid.link gateway the support 3 different Sub module like ZigBee, ZWave and Wireless MBus. The dispatcher also handles Online/Offline state for the GW. If no keep alive has been received in a user defined interval the GW is the marked as offline.

4.1.2.2 Core

- Creates a remote host session when it receives information from the dispatcher that a GW in online.
- If security is enabled the "Security manager" will then negotiate a unique security key with the GW. If this is a success the new remote host section is created an all communication is encrypted using the security key. The old remote host session will then be terminated. Back end will then be informed that a security session is created ready for communication.
- The core creates remotes host sessions for each GW in the network.
- Client tool interface is for debugging and enable the user to monitor all the communication between the GW, Core and back end.
- If no communication is registered for 24 hours the remote host session will be terminated.

4.1.2.3 Back end

- User defined application code is implemented in the back end. The back end register to the core and subscribe to x number of gateways that it will receive and send data to.
- The back end has a SmartAMM protocol lib that can en/decode all incoming messages from the server.
- The back end has logging functionality and can store all incoming data into a SQL data base og user defined file format.

4.1.3 The DBService Backend Design

This is a customized backend implementation which interface with the SmartAMM server to receive telegrams and forward only the relevant data to the DBService. Figure 4.5 shows a Unified Modeling Language (UML) structural diagram of its connection with SmartAMM sever and the DBSevice.

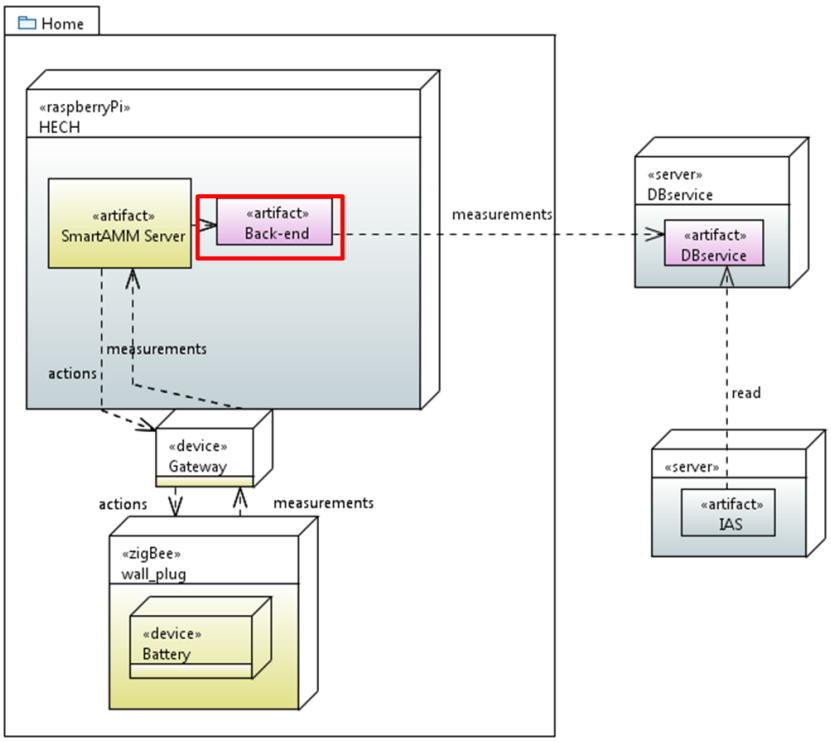

Figure 4.5: UML Deployment diagram with SmartHG UML profile annotations of the backend tool

4.1.4 HECH Remote Access Design

In SmartHG we designed a tunneling method to allow us to upgrade the HECH software remotely. This consists of the HECH creating a reverse SSH tunnel to a chosen server, thus alleviating the issue of the HECH being inaccessible from the outside since it will be located inside a private LAN. Developers can then make access through the server and

login to the desired HECH. Report IR IR3.1.3.2 Prototype of Open Protocol for Home Devices shows the implementations.

Chapter 5

Design of Open Protocol for IAS

5.1 Use Cases for the Service Market Controller

In IR4.1.2.2 of Deliverable D4.2.1, a Use Case (UC) diagram were presented together with four use cases for the SMC Service (smcservice). These were represented as initial design elements to display the externally visible functionality to its actors. In this internal report, the UCs have been extended to include abstract actors which allow for, not only the process of granting an Intelligent Automation Service (IAS) provider access to the Database Service (dbservice) through the smcservice, but also the IAS to IAS data access control. The abstraction of actors expresses a distinct interface for the considered users. It is therefore possible to reuse this because of the generality in expressing settings with different actors. For instance, the superuser of the smcservice might be the Distribution System Operator (DSO) in some domains, but it might also be a cooperation of DSOs in another setting.

Generally, a use case is defined to be a coherent unit of an externally visible functionality that expresses the relationships with its actors. It can be created for different implementation levels whereas the system level is the top view. A system level UC include high-level implementation decisions and be presented both in a formal and informal manner. The UC diagram illustrated in Fig. 5.1 presents the system-level view and has three base actors:

- **a client** which requests data access to a resource provider,
- **a resource provider** which represents an actor that generate or provide data. The Home Energy Controlling Hub (HECH) provides data to the dbservice, while the dbservice provides data to the other IAS,
- **a smcservice superuser** which represents the owner of smcservice, and
- **resource owner** which represents either the residential user or IAS developer depending on the ownership of the data. Meter data are considered owned by the residential consumer, while data produced by an IAS is considered owned by the IAS developer.

The UC diagram consists of five primary UCs where either one or more actors are provided with a measurable value. It does not include exception behaviour for the sake of brevity. The use cases are described in the following sections.

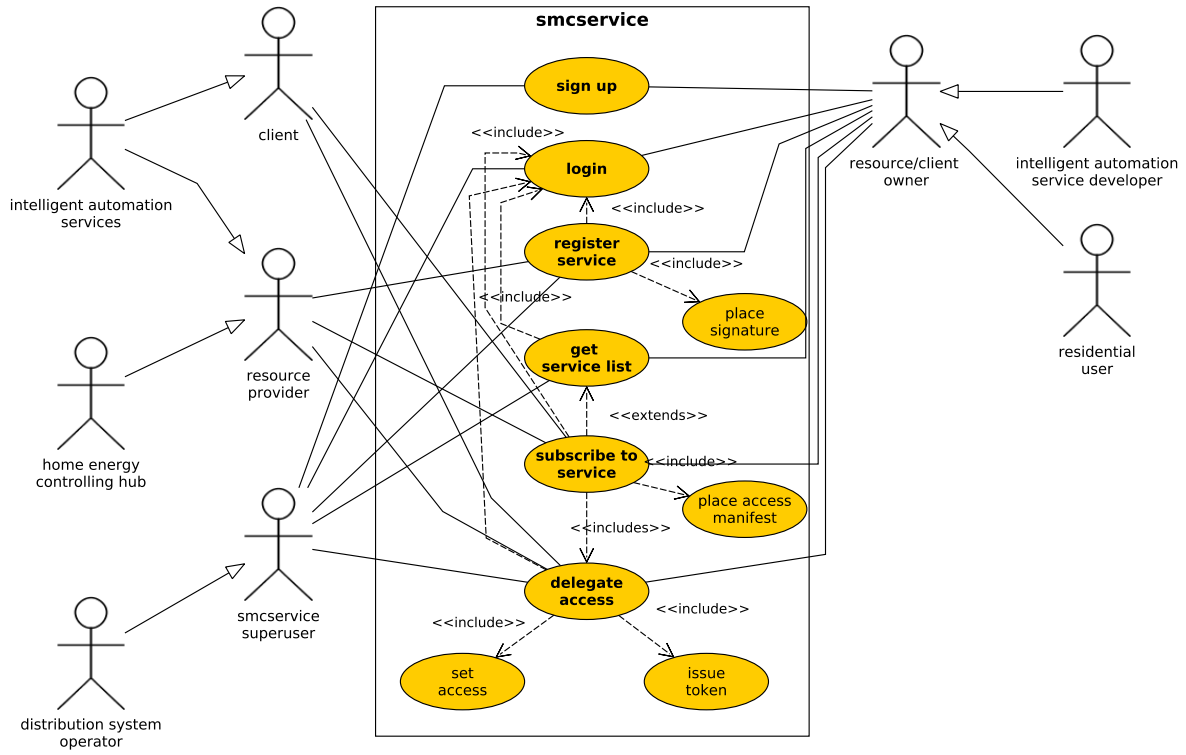

Figure 5.1: Use case diagram of the SMC service.

5.1.1 Sign up

SMC-UC1 - Sign up

Name: Sign up
Identifier: SMC-UC1

- Resource owner/client provides e-mail, password, address, postal code and city, where address, postal code and city are optional.
- System verifies that the e-mail is not already in the database.
- Smcservice user verifies the authenticity of the resource owner. This is can be done through a third party authentication system, but details about this is beyond the scope of the system.
- System creates an account for the resource owner.

The resource/client owner represents either the residential user or the service developer. In this use case, it is described as the residential user signs up directly, but this could also be the smcservice superuser that does it indirectly. Smcservice superuser would in many cases have external systems where the residential user would be listed. For instance, if the DSO has the residential user as customer already, then they would be able to fill in the address, postal code and city.

5.1.2 Login

SMC-UC2 - Login
Name: Login **Identifier:** SMC-UC2 • Resource owner/client provides credentials to the system. • Smcservice superuser verifies the authenticity of the resource owner. • Resource owner/client is logged in.

In order for the resource owner/client to use the system, credentials must be provided. The credentials are verified by the smcservice superuser. The login credentials can either be directly linked with an email and password or a token system, where the system can verify on authenticity on the same level as the manual provide email and password.

5.1.3 Register service

SMC-UC3 - Register service
Name: Register service **Identifier:** SMC-UC3 • Resource/client owner is required to login (follows SMC-UC2). • Resource/client owner provides the following: – Name – Uniform Resource Identifier (URI) to RESTful Application Programming Interface (API) – Public key – Description • The smcservice superuser can review these settings.

This UC includes the SMC-UC2 UC. With granted access, the resource/client owner is able to register a service. The service is characterized by four fields: name of the service, the URI to the RESTful API, a public key for encrypting an access token, and a description of the service.

5.1.4 Get service list

SMC-UC4 - Get service list
Name: Get service list **Identifier:** SMC-UC4 • Resource/client owner is required to login (follows SMC-UC2). • Resource/client gets the list of services registered in the system. The list provides information about name and description of the service.

In order for the resource/client owner to subscribe to a service, a service discovery must be made. The list of services is managed by the smcservice superuser.

5.1.5 Subscribe to service

SMC-UC5 - Subscribe to service
Name: Subscribe to service **Identifier:** SMC-UC5 • Resource/client owner is required to login (follows SMC-UC2). • Resource/client owner gets the list of services (follows SMC-UC4). • Resource/client owner selects a service to subscribe to. • Client is notified through a subscription request. • Client requests an authorisation to the resource owner for accessing the resource provider and sends a data access request (access manifest). • The use case continues in SMC-UC6 if the data access request is granted.

In this UC, the resource/client owner subscribes to a service which activates the client to request an authorisation to the necessary resources for delivering a service. Notice, the client is passive until it is requested to take action. Furthermore, in order for the client to get the resources, the resource owner has to grant permission based on the access manifest. This way, the resource is aware of the access level and information transferred between the resource provider and the client.

5.1.6 Delegate access

SMC-UC6 - Delegate access
Name: Delegate access **Identifier:** SMC-UC6 • This UC depends on SMC-UC5. • Resource/client owner sets the access level and grants permission to the client to access the resource provider. • System issues a token that contains information about the access level based on the settings of the resource/client owner and smcservice superuser. • System sends the token to the client. • Client requests information from the resource provider using the token to authenticate and authorise itself. • Resource provider sends the requested information to the client.

After the subscription phase where the client sends its data access request (access manifest), the resource/client owner can configure the access level for that particular service. For instance, if the client requests access level on sub-meter level, the resource/client owner can grant the client access on main meter level. In some situations, the client will not be able to provide the service with constrictions, but this is out of scope of the system.

5.2 API Design and Communication

The regular web interface and REST API represent the main entry point for the stakeholders of the system. The UC diagram in Fig. 5.1 identifies these stakeholders. Generally, the system distinguishes between the stakeholders as either service developers or regular users. The system provides the service developers with an API, while regular users should be provided a web interface. At this first stage of the design, the development focused a solution for the service developers in order for service to service interaction.

This chapter will in the following sections present the identified resource list and the service associations as in IR4.1.2.2 of Deliverable 4.1.2. Service associations are presented through a data model that can be used for structuring the database using Object-Relation Mapping (ORM).

5.2.1 Resource List and Associations

The UC diagram illustrated in Fig. 5.1 identifies the primary scenarios for the smcservice. The nouns of the use cases often represent the resources of the system. Based on this, the resources of the smcservice is identified to be:

- `User`
- `Service`
- `ServiceAccessToken`
- `ServiceDetails`

The list of resources determines the relative endpoints of the REST API. `User` is abstraction of the stakeholders for the smcservice. The user can be a resource/client owner or a smcservice superuser. The resource/client owner can be residential user or a service developer. The `Service` and `ServiceDetails` contain all the relevant information about the service. For instance, name, location, description, etc. The `ServiceAccessToken` stored the token for entering the service.

The resource identification supports the design of the data model, which associates the resource together. This is illustrated in Fig. 5.2. It illustrates the smcservice which is composed of the listed resources. Each `User` can have multiple `ServiceAccessToken`s, since he/she can subscribe to multiple services. Each service has zero or multiple `ServiceAccessToken`'s and is description with `ServiceDetails`.

5.2.2 High-level Description of Adaption to OAuth 2.0 Framework

The authentication and authorisation procedure are illustrated in Fig. 5.3, where the protocol phases are indicated by $(\cdot)$. It shows a successful authentication and authorisation

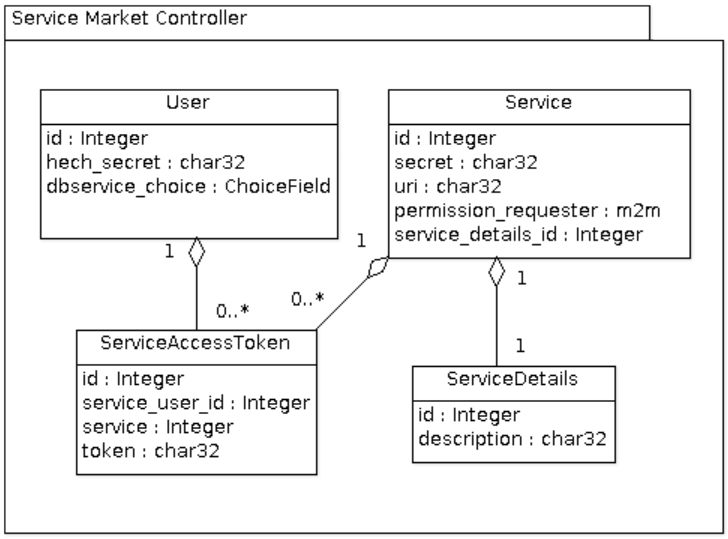

Figure 5.2: Class diagram of data model in the smcservice.

cycle between residential consumer and a Home Intelligent Automation Service (HIAS) (e.g., Energy Bill Reduction (EBR)). It is similar for the DSO and a Grid Intelligent Automation Service (GIAS), but for the sake of brevity, we only present one scenario. The protocol is assumed to be executed over a secure HTTPS connection to prevent man-in-the-middle attacks. Furthermore, it is assumed the HECH and the smcservice have been authenticated, authorised and can exchange data.

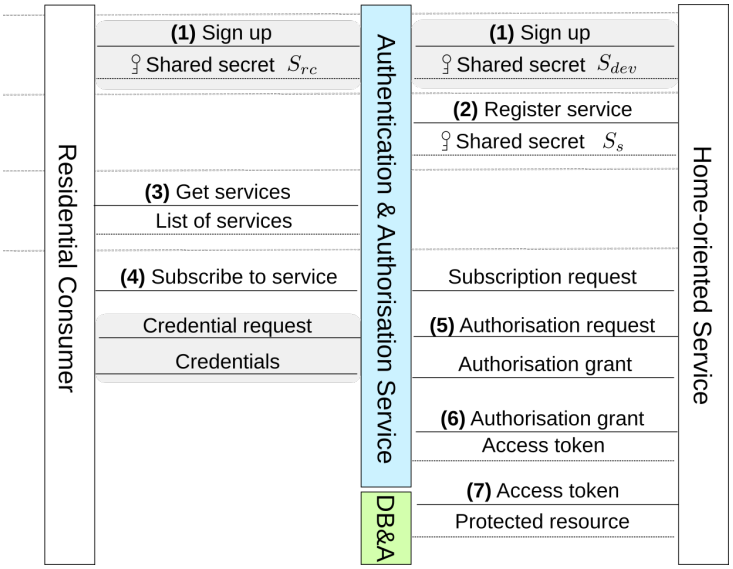

Figure 5.3: Authentication and authorisation process, where the smcservice presented as the authentication & authorisation service.

The phases for an authentication and authorisation cycle between the residential consumer and a HIAS are:

(1) A sign up procedure between the residential user and the smcservice, where the residential user proves his identity by a digital signature scheme (e.g. by a national wide authentication service and/or through a two-factor authentication service) and

exchange a shared secret S_{rc}. Similar sign up procedure is required for the developer wanting to register a service. They also exchange a shared secret S_{dev}.

(2) The resource/client owner registers a service after he is authenticated. It provides the URI, application name, and meta information about the categories it applies to. Furthermore, it provides its admission requirements to other services. The smcservice generates a shared secret S_s for the service that it must append to all messages henceforth. It is used to identify the authenticity of the service.

(3) The residential user requests for a list of services registered at the smcservice.

(4) The residential user subscribes to a service by accepting the admission requirements, and the service gets notified.

(5) For the service to acquire an authorisation grant, it can request the smcservice to request permission from the residential user through an authorisation code. The smcservice will require both authentication of the residential user, but also the acceptance of the access conditions (defining data scope, duration) from the residential user. The residential user never shares his credentials with the service. Furthermore, instead of sending an authorisation grant back to the service (as shown in Fig. 5.3), the smcservice can send the access token directly to the service.

(6) An authorisation grant received by the service can be exchanged with an access token through the smcservice. The access token is opaque for the service and specifies scope and duration of access granted by the residential user. Furthermore, it can be a self-contained access. This can provide the dbservice with all authorisation information which can be verified directly on the dbservice. The access token should be stored securely at the service, in order to reduce the risk of compromising the residential user.

(7) With the access token, the service can request access to the dbservice. The dbservice validates the access token and if it is valid, it sends a representation of the protected resource to the service.

The registration

As shown above the service registration is performed manual by developer in order for others to subscribe to it. In [4], they specify a protocol mechanism for requesting for service metadata in order to register dynamically. However, the mechanisms requires all other services to support this protocol and must obey additional criteria, thus for the sake of brevity this is not included for the smcservice.

5.3 Token Structure

The smcservice is the responsible for the authentication and delegation of access tokens to the dbservice, thus it authenticates and have an account for each user. A service requests permission to the dbservice indirectly through the smcservice. The smcservice generates a JSON Web Token (JWT) that specifies the scope, user id, time, scope that is asymmetric encrypted by the dbservice's public key and signed by the smcservice's private key. This way the service cannot decrypt and modify the content of the data request and the dbservice can be sure it is correctly sent by the smcservice by verifying the signature.

To make this work it has following assumptions:

- The dbservice must trust the smcservice for several things:
  - They should be able to make a trusted connection where information can be exchanged without being intercepted. Furthermore, they should be able to authenticate each other.
  - The smcservice must be allowed to have full access to the dbservice and know its capabilities. The capabilities are defined as e.g., its access control policy and the methods that can be retrieved by the other services.

The dbservice and the smcservice must be able to publish the public key in a secure way, such that no one can manipulate with the content. The requestor should have a default level of request, otherwise it should be verified manually by the smcservice. This way we ensure correct subscription of data sources from the residential home.

5.3.1 JSON Web Token (JWT)

Using JWT is becoming more prevalent data structure in exchanging integrity-protected information between web services. JWT offers multiple advantages compared to normal API keys, e.g., granular access control, decentralised issuance, etc. In the JWT terminology, the information represents a set of *claims* which is signed by the originator. The key feature of JWT, is the server's ability to verify the claims without storing any state about the client. This is necessary in order to follow the REST principles. TheJWT specification [3] defines the structure as:

> ... a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted.

A JWT is composed of three base64 elements where the elements are separated with a period character. The base64 conversion ensures a URL-safe string such that special characters or language specific letters always are among 64 careful chosen letters [5]. This way, the base64 encoding of the data establishes a common alphabet for various environments. The JWT has this general structure when it is digital signed:

`JOSE header . JSON Claims Set . Signature`

The header is called a *Javascript Object Signing and Encryption (JOSE)* header and specifies the algorithm used in the signing/encryption process of the claims set, such that the server can verify the integrity of the token. For a signing process this could be a SHA-256 hash-based authentication function, e.g. a HMAC-SHA256 abbreviated *HS256*. Furthermore, it includes the token type used, e.g. JWT. Without the base64 encoding and digital signature, the header and claims set can be presented as:

```JSON
{//JOSE Header
  "alg": "HS256",
  "typ": "JWT"
}.
{//JSON Claims Set
  "user_id": 1,
  "username": "smik@eng.au.dk",
  "exp": 1441617495
}
```

The set of claims are specified with a claim name (e.g., in the above JWT "exp" is a registered claim name) and claim value. The claim names are either a *registered claim name*, a *public claim name* or a *private claim name*. In the above example, `user_id`, `username` and `scopes` are private claim names specific for the application.

Depending upon whether the JWT should be digital signed or encrypted, the JWT is either a JSON Web Signature (JWS) [6] or a JSON Web Encryption (JWE) [7]. However, choosing either operation is not mutual exclusive, i.e., when applying the JWE format on the header and claim set, it is possible afterwards to apply the JWS.

5.3.1.1 Example of JWT with a Digital Signature

To give an example on a JWT with a digital signature JWS, assume the JOSE header and claims set from previous JSON snippet are used. Furthermore, assume the algorithm for the signing process is based on the HMAC-SHA256 construction. The Hash-based Message Authentication Code (HMAC) is based on symmetric key and a message. The format of the key should follow the JSON Web Key (JWK) [8] and JSON Web Algorithm (JWA) [9].

Python

```python
#!/usr/bin/env python3
header = '{"alg":"HS256","typ":"JWT"}'
claimset = '{"user_id": 1, "username":"smik@eng.au.dk","exp": 144167495}'
jwk_secret = '{"kty": "oct", "k": "bXlfc2VjcmV0X2tleQ"}'

def BASE64URL(string):
    """Encode to base64, translate +/ to -_, replace padding = with ''"""
    import base64
    return base64.b64encode(bytes(string, 'utf-8'), b'-_').replace(b'=', b'')

def SIGN(key_enc, string_enc):
    """Return base64 on HMAC"""
    import hmac, hashlib, base64
    hmac_result = hmac.new(key_enc, string_enc,
                           digestmod=hashlib.sha256).digest()
    return base64.b64encode(hmac_result, b'-_').replace(b'=', b'')

header_enc = BASE64URL(header)
claimset_enc = BASE64URL(claimset)
key_enc = BASE64URL(jwk_secret)

signature = SIGN(key_enc, header_enc + b'.' + claimset_enc)

print('Header:', header_enc.decode('utf-8'))
print('Claims:', claimset_enc.decode('utf-8'))
print('Signature:', signature.decode('utf-8'))

# Header: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
# Claims: eyJ1c2VyX2lkIjogMSwgInVzZXJuYW1lIjoic21pa0BlbmcuYXUuZGsiLCJleHAiOiAxNDQxNjc0OTV9
# Signature: l5cdY_WNrti2tQvnPX_xlxMf0WCjVAQOp5plGOIbYhk
```

As seen in the Python code snippet, the `BASE64URL` is customised to make URL-safe. The encoding ensures the characters `+/` is replaced with `-_` and the padding `=` is removed.

5.3.1.2 Security Considerations

To prevent man-in-the-middle and replay attacks it is critical to use Transport Layer Security (TLS) over HTTP. This will also authenticate the two parties before handing over the JWT. Typically, this will be sufficient to securely exchange the JWT, since TLS will be handled end-to-end. However, in server configurations with a load balancer, it might be necessary to encrypt the payload data to enforce privacy.

A standard JWT should not contain sensitive data. The JWT is only protected against manipulation. The claims should not reveal any secret information about the issuer or signer. For instance including a residential user's address in a JWT will make it very obvious who the JWT belongs to. If any sensitive information has to be stored in a JWT, then it should be encrypted using JWE.

5.3.2 Scopes for the Dbservice

The OAuth 2.0 (OAuth2) framework specifies the authorisation code grant flow when services (a.k.a. clients) request both an access token and a refresh token. Typically, this authorisation flow is used for confidential clients. The client initiates a flow by redirecting the resource owner's user agent to the authorisation endpoint on the authorisation server. In the request, the client specifies the client identifier, local state, redirection, scope and redirect URI to which the authorisation server will redirect the user agent back once granted or denied. The authorisation server authenticates the resource owner and the resource owner decides either to accept the grant proposal or deny access. If the resource owner accepts the grant proposal, the authorisation redirects the user agent back to the client using the URI redirect parameter or specified during client registration. In this redirection URI, the *authorisation code* and the local state are embedded. With the authorisation code and redirect URI, the client can request an access token on the authorisation server. The authorisation server authenticate the client and verify the redirect URI with redirect URI from the resource owner. If this is valid, the authorisation server responds with an access and refresh token to the client.

The OAuth2 facilitates the client to specify the scope of the authorisation grant and the access token. The OAuth2 specification [2] defines only the structure of the scope parameter and not the end-points on resource server. Formally, the scope parameter is defined as[1]:

```
scope       = scope-token *( SP scope-token )
scope-token = 1*( %x21 / %x23-5B / %x5D-7E )
```

The syntax should be interpreted as follows:

- '*' operator indicates repetition
- 'SP' indicate the space character
- scope-token is defined within the character set:

```
!#$%&'()*,+-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[]
^_`abcdefghijklmnopqrstuvwxyz{|}~
```

The design of the authorisation framework utilised this principle of enforcing the scope of data access for smart grid purposes. This ensures the residential consumer is aware of the data access given to a client (a.k.a. service). Furthermore, it uses an JWT assertion profile [10] for the authorisation grant in order to request an access token when the client wished to utilise an existing trust relationship that does not require user approval.

Besides the scope parameter being defined at "authorisation grant time", the agreed scope can also be included in the access token. This has the advantage of enforcing a decentralised issuance of the access, i.e., the authorisation server and resource server do not have to agree after the authorisation grant, the meaning of an access token, since the access token contains the information.

The customised smart grid scope parameter for the dbservice is defined as:

[1] See https://tools.ietf.org/html/rfc6749#section-3.3

```JSON
{
  "permission": ["read", "write"],
  "level": ["feeder", "mainmeter", "appliance_top3", "appliance_all"],
  "max_granularity": ["raw", "5min", "10min", "15min", "30min", "hourly", "daily",
                      "weekly", "monthly", "yearly"]
}
```

In the OAuth2 specification, the scope parameter is structured as a space-separated list of access claims. In this report it is presented as in a JavaScript Object Notation (JSON) format for brevity, however it is possible to define as distinct scopes by a combining them. For instance, reading the feeder with 30 minute granularity would be `read_feeder_30min`. Some combinations will not make sense, e.g., `write_appliance_top3_hourly`.

Chapter 6

Conclusions

In this deliverable we described the third year version of the design of all SmartHG Home Intelligent Automation Services (HIASs), i.e., of the SmartHG services which work on the residential homes side. Prototypes of such services are described in Deliverable D3.3.2. In the following, we present advancements with respect to the first and second year iterations, and we discuss the impacts of the services designed in WP3.

The second year version of Energy Bill Reduction (EBR) consisted in a controller automatically driving selected home appliances (i.e., the Energy Storage System (ESS) and the Plug-in Electric Vehicle (PEV)), in order to minimise the energy bill (either Time of Usage (ToU) or Inclining Block Rate (IBR)) of a residential user. In this third year, we improved the controller capabilities by allowing EBR also to discharge the PEV, whilst the second year version may only charge it. Moreover, in our new third year setting, EBR goal is not to directly minimise the energy bill of a given user, but to reduce costs for the energy retailer/Distribution System Operator (DSO). The communication protocols between home devices and Database Service (DBService) have been re-designed in order to integrate the residential main smart meters. A new protocol has been developed to connect remotely with the Home Energy Controlling Hub (HECH).

The design of open protocols for Intelligent Automation Services (IASs) shows a Use Case (UC) diagram for the SMC Service (smcservice) that embraces the IAS developers, residential users and DSO. It consists of six UCs which handles the sign up procedure, login, service registration, subscription and access delegation. Based on the UCs, a data model for the smcservice is presented together with a high-level description of the adaption to the OAuth 2.0 (OAuth2). For exchanging information using the OAuth2 protocol, a self-containing JSON Web Token (JWT) is used which is elaborated.

6.1 Impact

EBR can be used by both DSOs and retailers to obtain savings on the operation costs. Namely, as for DSO, EBR is able to keep residential users demand inside the power profiles output by Demand Aware Price Policies (DAPP). This allows DSOs to actually and effectively perform load shifting on Electric Distribution Network (EDN) substations, thus lowering down Transmission and Distribution (T&D) investment deferral costs. As for retailers, it allows them to save on energy costs and CO_2 emissions costs, in such a way that the starting hardware and software expenses are fully recovered.

The design of an open protocol for home devices enables seamless communication between smart home appliances and the DBService. Moreover, the design of open protocol

allows the developers to remotely upgrade the software of the HECH. The design of open protocols for the IASs has allowed IASs to integrate using web technologies in the smart grid domain. To enforce data access policies, a SmartHG Service Market Controller has been designed. This ensures user consent from the residential consumer and DSO within the SmartHG ecosystem.

Bibliography

[1] R. Fielding and J. Reschke, "Hypertext transfer protocol (HTTP/1.1): Message syntax and routing," tech. rep., jun 2014.

[2] D. Hardt, "The OAuth 2.0 Authorization Framework," 2012.

[3] M. Jones, J. Bradley, and N. Sakimura, "JSON web token (JWT)," tech. rep., may 2015.

[4] M. Jones, J. Bradley, M. Machulak, and P. Hunt, "OAuth 2.0 dynamic client registration protocol," tech. rep., jul 2015.

[5] "The base16, base32, and base64 data encodings," tech. rep., jul 2003.

[6] M. Jones, J. Bradley, and N. Sakimura, "JSON web signature (JWS)," tech. rep., may 2015.

[7] M. Jones and J. Hildebrand, "JSON web encryption (JWE)," tech. rep., may 2015.

[8] M. Jones, "JSON web key (JWK)," tech. rep., may 2015.

[9] M. Jones, "JSON web algorithms (JWA)," tech. rep., may 2015.

[10] M. Jones, B. Campbell, and C. Mortimore, "JSON web token (JWT) profile for OAuth 2.0 client authentication and authorization grants," tech. rep., may 2015.