FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

Interface between Smart Buildings and Smart Grids

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Technische Informatik

eingereicht von

Stefan Gaida
Matrikelnummer 0728007

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dr. Wolfgang Kastner
Mitwirkung: Dipl.-Ing. Daniel Schachinger

Wien, 03.08.2015

(Unterschrift Verfasser)

(Unterschrift Betreuung)

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.ac.at

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

Interface between Smart Buildings and Smart Grids

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Computer Engineering

by

Stefan Gaida
Registration Number 0728007

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dr. Wolfgang Kastner
Assistance: Dipl.-Ing. Daniel Schachinger

Vienna, 03.08.2015

(Signature of Author) (Signature of Advisor)

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.ac.at

Erklärung zur Verfassung der Arbeit

Stefan Gaida
Oskar Helmer Straße 25/2, 2542 Kottingbrunn

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)

(Unterschrift Verfasser)

Acknowledgements

Foremost, I would like to express my gratitude to my advisor, Wolfgang Kastner, who gave me the opportunity to write my master's thesis as a member of the Automation Systems Group. Likewise, I would like to thank Daniel Schachinger for his steady support. Both of them were always available for questions and discussions. A big thank you also goes to Markus Jung. He introduced me to possible topics for the thesis and gave me great support in the first months.

Of course, I would like to use the chance to thank my parents Sigrid and Georg for their mental, but also financial, support during my time as a student and the time before. My grandparents also deserve the highest gratitude for their support of my studies.

Abstract

Due to the change from fossil fuels to renewable and intermittent energy sources such as wind turbines and solar panels, power grids have to fulfill vastly new demands. Apart from centralized power plants, also electricity that is generated by customers, for example, with their own solar panels has to be considered. The main issue with this evolution is that the extent of energy production varies with weather, time of day, and season. Additionally, it is expected that the shift from less energy efficient sources (e.g. oil, gas) to electricity in sectors like transport will further increase the demand. The future power grid that takes all these developments into account is called smart grid. The decisive difference to the current infrastructure is the intense use of computer networks for communication between service providers, utilities, and customers. This enables all parties to keep the grid stable, save money, and use energy when it is available.

An important part of the smart grid is formed by the interface to smart buildings, i.e. to the customers. Currently, there exists no international standard that specifies rules for this kind of communication. In this thesis, appropriate network protocols and data models are investigated for their ability to fulfill the requirements of such an interface, which are identified in the beginning of the thesis. Afterwards, the OASIS standards Energy Market Information Exchange (EMIX), Energy Interoperation (EI), and Open Automated Demand Response (OpenADR) as well as the IEC Common Information Model are introduced as potential candidates.

In addition to the common interface properties, specific use cases for the communication between smart buildings and smart grids are identified and defined in detail (e.g. propagation of pricing information). This task also includes the description of communication partners and systems. Based on the detailed use case specifications, the information which has to be included in the exchanged messages is analyzed. For each of these messages, standards that are able to define the structure of the signal are determined. Subsequently, the identified possibilities are compared to each other to allow the selection of the best option.

Finally, a proof of concept implementation for the smart grid interface and a corresponding infrastructure, which uses EI, is presented. A model of a smart grid is developed in order to demonstrate applicability of the selected protocol for the specified use cases. The demonstration environment consists of several Raspberry Pis representing customers, or service providers in the grid. The behavior of all these participants is simulated by distinct Java applications. The realization uses the Extensible Messaging and Presence Protocol (XMPP) to exchange messages. Therefore, also an XMPP infrastructure consisting of XMPP servers and additional services like DNS, NTP, or DHCP is defined.

Kurzfassung

Die Energiewende weg von fossilen Brennstoffen hin zu erneuerbaren und intermittierenden Energiequellen wie Windkraftanlagen und Sonnenkollektoren stellt neue Anforderungen an die Stromnetze. Abgesehen von der zentral erzeugten Energie in Kraftwerken muss auch jene, welche zum Beispiel durch Sonnenkollektoren beim Kunden erzeugt wird, berücksichtigt werden. Die größte Herausforderung dabei besteht darin, dass das Ausmaß an produzierter Energie mit dem Wetter, der Tageszeit und der Jahreszeit variiert. Darüber hinaus wird erwartet, dass der aktuelle Trend, Elektrizität an Stelle von weniger effizienten Energiequellen (z.B. Öl, Benzin) in Sektoren wie der Mobilität einzusetzen, zu einer weiteren Erhöhung des Strombedarfs führt. Zukünftige Stromnetze, welche diese Entwicklungen berücksichtigen, werden allgemein als Smart Grids bezeichnet. Der entscheidende Unterschied zur derzeitigen Infrastruktur besteht dabei in der intensiven Nutzungen von Computernetzwerken zur Kommunikation zwischen Dienstleistern, Versorgungsunternehmen und Kunden. Dadurch besteht für alle Teilnehmer die Möglichkeit zur Stabilität des Netzes beizutragen, Geld zu sparen und die Energie dann zu verwenden, wenn sie vorhanden ist.

Ein wesentlicher Teil des Smart Grids wird durch die Schnittstelle zu Smart Buildings und damit den Kunden gebildet. Zurzeit gibt es keinen internationalen Standard, der Regeln für diese Art von Kommunikation festlegt. In der vorliegenden Arbeit werden geeignete Netzwerkprotokolle und Datenmodelle identifiziert und anschließend auf ihre Eignung zur Erfüllung der Anforderungen an eine solche Schnittstelle geprüft. Diese Anforderungen werden zu Beginn der Arbeit definiert. Mögliche Kandidaten sind in diesem Zusammenhang die OASIS Standards Energy Market Information Exchange (EMIX), Energy Interoperation (EI) und Open Automated Demand Response (OpenADR) sowie das IEC Common Information Model (CIM).

Zusätzlich zu den allgemeinen Anforderungen an die Schnittstelle werden spezifische Anwendungsfälle für die Kommunikation zwischen Smart Buildings und Smart Grids identifiziert und im Detail ausgearbeitet (z.B. Verteilung von Preisinformation). Diese Aufgabe inkludiert auch die Beschreibung der Kommunikationspartner und -systeme. Darauf aufbauend wird analysiert, welche Information in den ausgetauschten Nachrichten der Anwendungsfälle enthalten sein muss. Für jede dieser Nachrichten wird bestimmt, welche zuvor identifizierten Standards dazu in der Lage sind, die Struktur und den Inhalt des Signals zu definieren. Die identifizierten Möglichkeiten werden miteinander verglichen und die geeignetste Option wird ausgewählt.

Abschließend wird eine auf EI basierende Proof of Concept Implementierung zur Demonstration der Anwendungsfälle präsentiert. Der Demonstrationsaufbau besteht aus mehreren Rasp-

berry PIs, welche entweder einen Kunden oder einen Dienstleister im Netz repräsentieren. Das Verhalten dieser Teilnehmer wird durch prototypische Java Applikationen simuliert. Zum Austausch der Nachrichten wird das Extensible Messaging and Presence Protocol (XMPP) verwendet. Daher wird auch eine entsprechende XMPP-Infrastruktur bestehend aus XMPP Servern und zusätzlichen Diensten wie DNS, NTP oder DCHP hergestellt.

Contents

CHAPTER 1

Introduction

1.1 Motivation

The way of producing and consuming electrical power is currently changing. In the past, the electrical power generation was a centralized and reliable task which was fully controlled by producers. It was possible to predict the load statistically, and it was sufficient to have only a unidirectional flow of energy, namely from producers to consumers. However, the energy transition away from fossil fuels and, in some countries like Germany, also away from nuclear power towards renewable and intermittent sources, such as wind and solar, provides vastly new demands on the power grid. Additionally, consumers start to produce their own energy by placing solar panels on their buildings. Thus, they turn into so-called *prosumers* as they both produce and consume electricity. If at least a part of the produced energy is fed into the grid, this leads to a bidirectional flow of energy. In order to handle this new role allocation and the variation of produced power over time, the grid needs to be flexible. This flexibility can be achieved by developing intelligent networks, the so-called smart grids [9] [38].

In addition to the decarbonisation, new and intelligent power grids also have to adapt to changing customer needs. It is expected that the demand for electrical power will increase in the next years. In Europe, it is assumed, that the consumption will grow by 1-2% each year until 2020. One reason for this trend is the shift from less energy efficient sources to electricity in sectors like transport, heating, and cooling. Furthermore, the consumers should be able to manage and adjust their use of energy throughout the day by responding to real-time information like the currently available energy or the current price. In summary, the most important goal of smart grids is the use of energy when it is available [9] [38].

1.2 Problem statement

For the realization of such an intelligent power grid, suitable network protocols are needed. An important part of smart grid communication takes place between the grid and the customers.

Although the development of smart grids is forced in many parts of the world like Europe [9], the US [25] or China [4], there is no international standard for this purpose, yet. Thus, leading companies, want to establish a future IEC transportation and communication standard to transport messages between the grid and the customers [29]. First, it is necessary to establish an appropriate network stack, where the Extensible Messaging and Presence Protocol (XMPP) could be an interesting option at the ISO/OSI application layer. It has to be clarified if XMPP fullfills the requirements for this kind of communication. But also the underlying layers have to be specified. Second, the structure of information which is exchanged has to be defined in form of a data model. The latter is the main contribution of this thesis. For this purpose, already existing national standards like Energy Interoperation (EI) or Open Automated Demand Response (OpenADR) must be investigated.

1.3 Aim of the work

On the one hand, the expected outcome of this master thesis is a state-of-the-art survey and a comparison of current developments regarding the communication between smart buildings and smart grids. As part of the comparison also a gap analysis will be performed. Therefore, it will be necessary to define important communication use cases. They should be used to identify standardization gaps and to evaluate the ability of different protocols and data models to support all requirements of these use cases.

Based on the results of the comparison, the second and more practical outcome will be a demonstration setup in form of a proof-of-concept implementation to check if the communication between smart buildings and the agents in the grid (e.g. grid operator or energy retailer) works as expected. Thereby, suitable hardware components for the network infrastructure and the representation of all participating agents have to be found. Afterwards, the most convenient communication protocol will be implemented in accordance with the preceding comparison.

1.4 Methodology

The methodological approach to achieve the expected results consists of the following steps:

Literature review on transport protocols

First, existing and emerging standards for the communication between smart grids and smart buildings will be identified and studied. In addition, XMPP and its possible usage in the field of smart grids will be a crucial topic. Related articles and reports are incorporated to get an overview on these issues.

Identification of use cases

In a second step, important communication use cases will be identified. For each use case, the interaction between the involved agents (grid operator, energy retailer, energy pool manager and customer) will be defined. They are worked out in detail by the use of sequence diagrams showing the messages which have to be exchanged via a computer network.

Signal definition and information model

For all signals, which are needed within the identified communication use cases, suitable message structures have to be found. It will be studied if it is possible to model the signals by means of existing national standards like Energy Interoperation or OpenADR. Moreover, self-defined and vendor specific signals will be taken into account. The resulting structures will be compared to each other. Also the advantages of XMPP in comparison to RESTful web services will be studied. The outcome of this step is a defined message structure for each signal and an analysis about which standard is able to support which signal and use case.

Proof-of-concept implementation and evaluation

The use cases will be demonstrated by the implementation of a possibly XMPP-based solution, which uses the previously identified signals. The aim of this implementation is to check if the protocol works as desired. To do this, it will be necessary to design an experimental setup which represents the involved agents and the communication between them.

1.5 Structure of the work

This section provides a general view on the structure of this thesis. In Chapter 2, the state of the art concerning technologies and concepts for the exchange of data between smart buildings and smart grids is presented. First, the communication interface itself is described in detail. This part also includes the important concepts of demand side management and demand response. Additionally, requirements that have to be fulfilled by protocols which can be used in smart grids in general and for the particular interface are identified. At the end of this chapter, possibly suitable standards for data models and network protocols are described.

Chapter 3 introduces relevant use cases for the communication between electricity customers and the smart grid. It also contains the description of the participating agents.

The investigation whether existing standards are convenient to implement the use cases, is performed in Chapter 4. It starts with the clarification of the question, if XMPP provides a suitable communication infrastructure. Also the underlying network protocols are discussed. Afterwards, suitable data structures for the signals, which have to be exchanged, are searched. If there are multiple options to model all the required signals, they are compared to each other. The chapter concludes with the selection of the best solution.

Consecutively, Chapter 5 deals with the proof-of-concept implementation, which is used to demonstrate the use cases. First, the hardware and software components of the demonstration environment are presented. Subsequently, common configuration parameters and the common software architecture for clients are described in detail. The following subsections deal with functionality, configuration, and program structure of each particular agent. Finally, Chapter 6 completes the thesis with a summary and possible future work.

CHAPTER 2

State of the art

The underlying chapter comprises the theoretical background about technologies and concepts which are of interest for developing an interface between smart buildings and smart grids. The first part of the chapter describes the functionality and the requirements for that interface. As the thesis is concerned with the comparison of current developments regarding the information exchange between smart buildings and smart grids, the main focus of the second part lies on potentially suitable, standardized communication protocols and data models.

2.1 Smart Grid User Interface

In literature, the interface between smart buildings and smart grids is often referred to as Smart Grid User Interface (SGUI). The SGUI is an abstract interface for the communication between the electricity customer domain and one or more external service providers. Another equal term, which is mainly used in the US, is Energy Service Interface (ESI) [27], [25].

2.1.1 Overview

First, it is important to understand the exact meaning of the SGUI. The term user in its denomination refers to an actor that consumes electricity from the power grid. In addition, it can also describe a so called distributed energy resource (DER), which produces local energy (e.g. by solar panels or wind turbines) or stores energy by the use of batteries. In real world, the user will be a residential home, a vehicle, a commercial building, an industrial plant, some single device, or a system in one of these facilities, or even an aggregation of them. The user interface is the means that is necessary for the communication between such a user and a power grid service provider. Therefore, the SGUI serves as a link between the customer domain and the grid domain. Both domains evolved independently from each other over the years. This led to a situation, where different information models and different communication protocols are used in each domain. It is the task of the SGUI to provide a common machine-to-machine interface, which brings these two worlds together [27].

Important data exchanged across the SGUI include: demand response (DR) events like load sheds or load shifts, pricing signals, market transactions, meter data, load parameters (demand, load forecasts or storage availability), emergency signals provided by the grid operators, and other energy related information like power quality [14]. It is not necessarily the case that at each particular interface all kinds of information are exchanged. In some simple scenarios it might be the case that the customer receives only pricing information.

The SGUI can be used either as low-level or high-level interface to influence the behavior of the customers depending on grid-side conditions. On low-level, specific devices or systems in the customer's domain like lighting or HVAC (heating, ventilation and air conditioning) systems are directly controlled by an external actor on the grid side via the SGUI. Thereby, the interface is located on device or system level. Used as a high-level interface, signals like DR events or price information are interpreted by the customer's facility. The customer decides how to react on changing electricity prices or grid-side operational needs. Since the high-level approach promotes more customer choice, scalability, flexibility, and security, the focus of this thesis will be on this case [27].

Until now, there is no final international standard for the communication across the SGUI. But, after the definition of demand side management (DSM), DR, and general requirements on the interface, some existing and emerging national standards, mainly from the US, are presented in this chapter.

2.1.2 Demand side management

A widely accepted definition describes DSM as a task for power companies or utilities to reduce or remove peak load, and thus stretch the capacitive limits of the power grid. Thereby, the need for installation of new capacities and distribution facilities should be prevented. In comparison to increasing power generation and extending the overall grid capacity, DSM is a relatively cheap option to deal with growing demand. Examples for commonly used DSM methods are load shaping and the installation of high-efficiency generation units. A load shape describes the hourly, daily and seasonal demand of electricity. Therefore, the activity of load shaping includes everything that can be done by utilities to influence this demand. Categories of load shaping are: peak clipping, valley filling, load shifting, strategic conservation, strategic load growth, and flexible load shape [10]. All of them are visualized in Figure 2.1. Overall, DSM can be interpreted as a top-down approach where the power industry implements measures on the demand side to increase the efficiency of the energy system. While, for example purchases of less consuming facilities itself are not covered by DSM, a program to promote such investments meets the definition. This shows that there are not just technical mechanisms, but also marketing activities to support the aim of DSM [9] [10].

According to [26], DSM can also be categorized depending on timing and the impact of applied actions on the customer side. These categories are: energy efficiency, time of use, DR and spinning reserves. Figure 2.2 shows the classifications of these categories in terms of timing and impact on the customer. Measures that have a high effect on customer behavior should last only for small fractions of time while others, which have no negative consequences on comfort

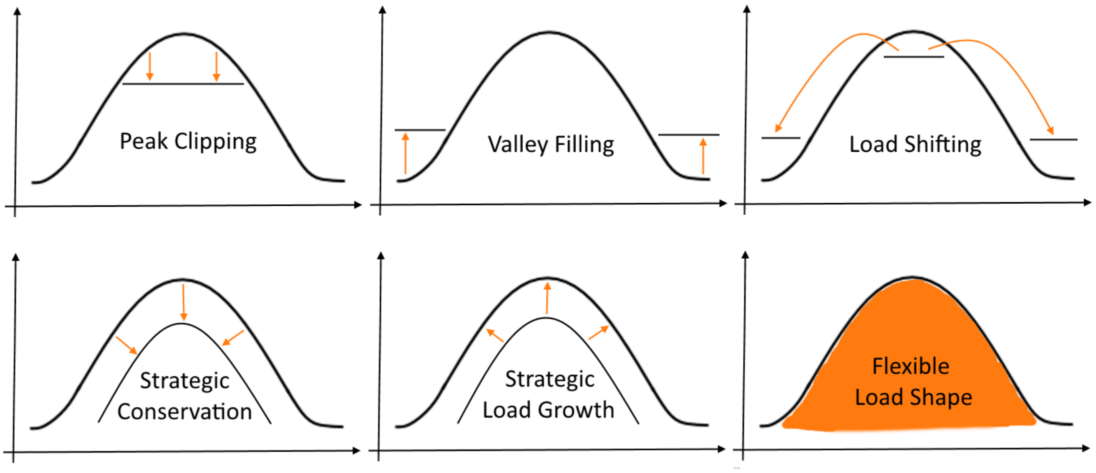

Figure 2.1: Categories of load shaping

or process output at the customer side, can be made permanent. The bottom-left edge of the figure is occupied by energy efficiency measures. They include permanent changes on equipment or buildings like better insulation or replacing facilities. Time of use (TOU) describes pricing models, where the energy price varies with time of usage within a day. Nonetheless, the price schedule is still static and predefined by contracts between supplier and consumer. The much more dynamic mechanisms of DR are described later in the next section. Although DR follows more of a bottom-up approach where the customer becomes the active part, this classification views it as part of DSM. To overcome this contradiction, one could interpret the enrollment of the required infrastructure and programs as top-down activity. The upper end of the spectrum in the diagram is represented by spinning reserves (SR). They cover emergency cases like frequency or voltage band deviations which could lead to blackouts. SR can be interpreted in two different ways. On the one hand, it can be seen as primary and secondary control, which is very traditional. Primary control means that active power output depends directly on frequency or voltage in the grid. The later describes the process of restoring frequency and voltage in the case of steady-state errors introduced by the primary control [39]. On the other hand, loads can be seen as virtual SR. This means, that in case of problems in the grid, the customers have to lower or rise their consumption corresponding to the given conditions [26].

2.1.3 Demand response

As mentioned in previous sections, DR mechanisms are one of the SGUI applications. More precisely, they are key applications. The term DR was defined by the US Federal Energy Regulatory Commission (FERC) in the year 2007 as:

> „An electric demand-response activity is an action taken to reduce electricity demand in response to price, monetary incentives, or utility directives so as to maintain

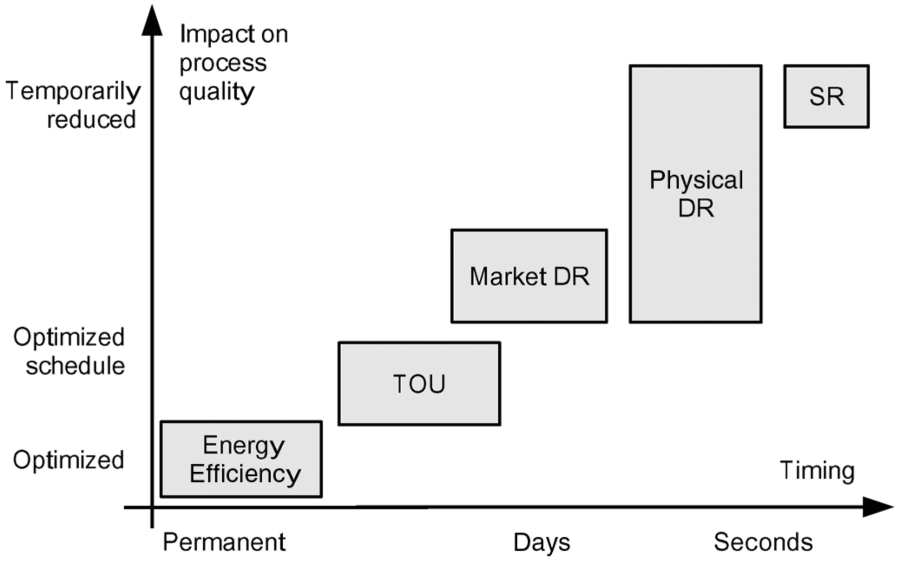

Figure 2.2: Categories of Demand Side Management [26]

reliable electric service or avoid high electricity prices.“ [6, p.5]

One possibility to classify DR actions is the distinction between incentive-based DR and time-based rates DR. Incentive-based DR comprises programs where customers receive payments to reduce their electricity usage in the case of reliability contingencies like imminent blackouts, voltage/frequency band violations, or capacity bottlenecks. A reduction of energy demand can be achieved by shifting load to off-peak periods, curtailing a production process, or also by using on-site electricity generation. On-site generation does not lower the overall energy usage, but reduces the demand on the power grid. Some important kinds of incentive-based DR programs are described in the following [5], [8].

Direct load control (DLC)

Within DLC programs, the system operator or grid operator is authorized to remotely shutdown and restart electrical equipment of the customers in the case of system or local reliability contingencies. In return, the customers get an incentive payment. Prior to the execution of a DLC action, the customers have to be informed about the expected inconvenience. In order to increase customer acceptance, it makes sense to limit the number and duration of equipment shutdowns per year. Nevertheless, the customers have to accept the greatest amount of external determination in this type of DR.

Interruptible or curtailable rates

Customers that take part in DR programs using interruptible or curtailable rates receive a rate discount when they agree to reduce load in the case of impending power grid problems. In contrast to DLC, customers can choose if they want to follow the instructions of the system operator. Additionally, it is up to the customer how the curtailment of load is achieved. But if they do not reduce their electricity consumption, they can be penalized. Also these programs need an upper bound for frequency and duration to increase acceptance.

Emergency demand response programs

Customers participating in emergency DR programs receive requests to change their load in the case of imminent emergencies in the grid from the system operator or grid operator. If they comply with the request, they get an incentive payment. This is the same as in programs using interruptible or curtailable rates. On the contrary, the customer does not get penalized when refusing to obey the instructions. Therefore, it is not necessary to limit the number of such emergency events per year. The lack of penalty payments may lead to better acceptance under the electricity customers.

Capacity market programs

Within capacity market programs, pre-specified load reductions in the case of contingencies are negotiated at conclusion of the contract between the system operator or grid operator and the consumers. If a customer does not curtail when requested, fines must be paid. In return, the customer gets guaranteed payments even if there are no emergency cases. Therefore, capacity market programs can be seen as some kind of insurance for the system operator or grid operator.

Demand bidding or buyback programs

These DR programs encourage customers to provide load reductions at prices for which they are willing to reduce their energy consumption. This means that the customers do not only have the choice to select whether they want to curtail or not, but they can also choose the price. Afterwards, the system or grid operator decides, if the offer to reduce load is accepted. Another but similar possibility is that the customer is asked for the extent of load reduction which is accepted for a given price. Again, the system operator or grid operator chooses if the bid is ordered. Typically, demand bidding or buyback programs include predictions or forecasts of electric demand for one day in advance. To ensure that customers do not offer curtailments without reducing their load when desired, they can be penalized.

Time-based rates DR, on the other hand, refers to energy usage adaptions by the customers as response to rising or falling prices. At times when prices are high, customers will tend to lower their consumption. Normally, prices will be high at peak-load times and low at off-peak times. The following time-based rates DR options are known [8].

Time-of-use (TOU) rates

TOU rates are a pricing concept where the electricity price varies slightly throughout the year (seasonal) and the day. As the prices are static and therefore predefined for long terms, TOU rates are not always classified as DR. For example, Figure 2.2 shows TOU as a distinct category of DSM with longer timing and less impact on the customer. In typical scenarios, the price will be higher for months of a year and hours of a day where the demand is known to be high by historical experience. In contrast, it will be lower at off-peak times.

Critical peak pricing (CPP)
CPP is a pricing model that uses TOU rates, or alternatively classical pricing with fixed rates. Additionally, very high critical peak prices which are triggered by system contingencies are utilized. Typically, the peak prices are predefined by a contract. The number of times when the critical peak price is applied within a year has to be limited.

Real-time pricing (RTP)
RTP is the most dynamic form of pricing. The actual market price is directly forwarded to the customers. Like TOU, this allows multiple price changes throughout the day with the difference that the price is not predetermined. A special form of RTP is day-ahead pricing. Within this concept, the electricity price is provided to the customers for the next day. This has the advantage that the customers can plan ahead their energy usage for each day in a way that costs are minimized. On the other hand, utilities use hourly predictions and experience to control the overall energy consumption throughout the day by low and high prices.

Another possibility to classify DR activities is the distinction between market DR and physical DR. Market DR comprises all mechanisms that rely on market places where prices are formed and products are traded. This includes the presented time-based rates DR options as well as demand bidding or buyback programs. Physical DR in contrast comprises actions where binding requests to reduce or increase load are sent to the customers. This includes most of the described incentive-based DR programs. To ensure an optimal grid operation, a good mixture of market and physical DR will be necessary [26].

2.1.4 Traffic Light Concept

The Traffic Light Concept (TLC) was developed by the German Association of Energy and Water Industries. It allows to describe the system status for a particular point in time and a particular grid area by the use of the three colors green, amber (or yellow) and red. Each color or, in other terms, each state of the traffic light implies the application of certain rules for the interaction of all relevant market roles and the roles regulated by law in the respective area of the grid. The most important role, which is strictly regulated by law, is the grid operator. An example for a market role is given by the suppliers [3].

The green traffic light phase is called market phase and indicates that there exists no critical situation in the grid. Therefore, it is seen as normal operating mode. In this phase, the market operates freely and competitively without any intervention of the grid operator. However, the grid operator is not necessarily prohibited from participating in the market. All smart grid participants try to optimize their own position according to economic factors without taking the grid situation into account [13], [3].

In the amber or yellow phase, the grid operator actively participates in the market to prevent the grid from becoming unstable. The objective of this temporary state is to keep the grid stable by the solely use of market activities. This includes the execution of pre-agreed contracts as well as paying current market prices for appropriate actions that help to prohibit critical situations. Such an action is to lower or increase the energy consumption for some time. The important point of this state is that the customer can decide how to react on requests from the grid operator [13].

If prevention of a critical situation by pure market activities in the amber phase fails, the power system enters the red state. Thereby, the grid operator takes control of the market interactions for the problematic grid area. The actions that can be taken have to be specific, well-defined, and temporary. Depending on contracts and regulations the grid operator can override contracts existing in the market, execute emergency actions, or execute direct controls over generation or demand to an extent that is necessary to stabilize the grid [13].

2.1.5 Requirements

In general, the smart grid has to fulfill several requirements. First, it is necessary to provide a guaranteed level of quality of service (QoS) in the network. Important QoS parameters are latency and bandwidth. As smart grids include real-time actions like online sensing and metering mechanisms, it is important to ensure that latency is low. When the number of connected smart grid participants grows, the infrastructure has to transport more and more messages simultaneously without any impact on the latency. Therefore, the available bandwidth has to grow faster than the demand for it. Just as important is the interoperability of the communication systems between all participating parties. This can be achieved by providing common frameworks and standards. The third critical requirement is scalability. It must be possible to integrate a very large and growing number of devices into the network. As the power grid is a fundamental infrastructure, also security is an important topic. For example, vulnerabilities that allow the intrusion into crucial parts of the smart grid might allow an attacker to alter load conditions which lead to a destabilization of the grid. Apart from deliberate attacks, the security part also covers fragility to user errors, equipment failures and natural disasters [40], [34].

In this thesis, the focus is not on the entire smart grid, but only on the communication via the SGUI. In some sense there are slightly different requirements for that kind of communication. For example, latency is allowed to be higher because real time sensing data that describe the status of the grid are not transmitted from or to customers. The IEC has defined a list of requirements for SGUI communication [38]. In the subsequent chapters, this list will be used to identify appropriate protocols.

- A very large number of customers and resources (more than 1 million) has to be connected to the servers or systems in the grid.

- The communication infrastructure must be optimized for a large number of small data packets.

- There has to be a mechanism that allows the determination of availability of all systems in the network.

- Signals shall reach their destination within seconds without storing them somewhere in between and without polling.

- All communication partners have a unique and trusted identity, which can be used to contact them.

- Encryption and authentication in both directions is required.

- To prevent a single point of failure, the network infrastructure has to be decentralized.

- Server to server communication has to be supported.

- Messages shall be based on XML. Additionally, clients should ignore attributes which they do not understand.

- There should be no need to open firewall ports at the customer sites. This implies that resources at the customer establish only outbound connections.

- Resources must be able to receive messages asynchronously.

- The communication network has to support the addressing of single resources as well as group communications.

- Publish and subscribe has to be supported.

- The protocols have to support service discovery mechanisms.

- Redundancy must be possible.

- Appropriate actions have to be taken to prevent denial of service attacks.

2.2 Standards and protocols

The communication via the interface between smart buildings and smart grids requires appropriate network protocols. On the one hand, a protocol that transports the messages through the network is required [34]. On the other hand, a standard that defines the structure and meaning of the exchanged data has to be found. The following subsections mainly describe standards that could potentially be used to define the message structure. For transportation, the focus will be on the Extensible Messaging and Presence Protocol (XMPP).

2.2.1 Energy Market Information Exchange

The Energy Market Information Exchange (EMIX) specification defines an XML based information model for the exchange of price and product information in power and energy markets. The Organization for the Advancement of Structured Information Standards (OASIS) standard, which is developed by the EMIX technical committee, is available in version 1.0 since 2011. A second release with the same version number (committee specification 2) was published in 2012. The scope of application is not restricted to communication between smart buildings and smart grids. In general, it is not even bound to power or electricity markets. The model is intended to be used in a large variety of market-oriented interactions [23].

Fundamental elements of product information in EMIX are quantity and quality of supply but also optional attributes like energy source or carbon costs. The former elements are called intrinsic, which means that they „belong to a thing by its very nature“ [23]. The latter, which are not an essential part of the product, are called extrinsic and are represented by so-called EMIX Warrants. The specification describes the intrinsic properties as information on the face of an envelope, while the extrinsic properties are the contents of the envelope. One of the most fundamental aspects of EMIX is the use of the OASIS Web services calendar (WS-Calendar) standard to convey time, durations, schedules, and intervals. Therefore, WS-Calender provides the possibility to represent complex price tariffs, where energy costs vary throughout the day. Moreover, quantitative and qualitative data can be modeled over time in this manner. The central element for representing this evolution of data over time is a sequence. If the sequence is populated with product descriptions, the sequence is called a schedule. Sequences consist of intervals while one specific interval is referred to as gluon, which is a special kind of interval. All other intervals inherit the gluon properties, which are not redefined within the particular interval. Intervals have a duration and include a so-called artifact. In EMIX, this artifact is the actual product description that may include the price and quantity of the product [23].

Name	Files	Description
EMIX schema	emix.xsd, emix-terms.xsd, emix-warrents.xsd	Framework and extensibility, agreement types valid for many markets
SI Scale schema	siscale.xsd	Enumerations of decadic units for measurements
Power schema	power.xsd, power-product.xsd, power-quality.xsd	Specific information format for power and energy markets
Resource schema	resource.xsd	Specific capabilities of devices and systems

Table 2.1: EMIX XML schemes

EMIX comprises four different XML schemes, which are listed in Table 2.1. While the core EMIX schema and the SI scale schema are common for all kinds of applications, the power schema and the resource schema are extensions of EMIX especially for power applications [23].

The core of EMIX consists of the four abstract types *EMIX base*, *item base*, *schedule* and *product description*. All these types can be extended for their use in different domains. The most general type is the EMIX base type. It conveys the schedule, which includes product descriptions, and allows extensions to express additional market information. The item base is an abstract base type for units of products. Types that are derived from item base have at least a name, a unit of measure, and a scale factor. Price and quantity, however, are not included in the item base. As previously described, they are part of the product description which is embedded in a schedule. Prices itself are modeled by the low-level abstract type *price base*. Predefined extensions of this type are *price*, *price multiplier* and *price relative*. While price is a simple, absolute amount of money, price multiplier and price relative are related to some baseline (e.g. the last price value). The advantage of this abstract concept of prices is that wherever the specification allows the presence of a price, one can use any concrete type of prices. Additional general and abstract types are interfaces and resource related types. EMIX interfaces describe the location where the ownership of a product changes. In power markets, this location may be a node, a meter, aggregations of them or simply a geographical location. EMIX resource descriptions are used to describe things like generators, load curtailments, or other energy resources [23].

2.2.2 OASIS Energy Interoperation

In November 2009, the National Institute of Standards and Technology (NIST) has started the Smart Grid Interoperability Panel (SGIP). They coordinate the development of smart grid standards and published the NIST Framework and Roadmap for Smart Grid Interoperability Standards [25]. Based on this work, the Energy Interoperation Technical Committee of OASIS has created the Energy Interoperation (EI) specification in version 1.0, which has been finished in May 2014. This specification is not a completely new development. An important part was the standardization of OpenADR, but the specification also includes technologies like EMIX and WS-Calendar to describe products, prices, and schedules [24], [25].

EI is an information and communication model to coordinate energy supply, transmission, distribution, and use between any two stakeholders, which are involved in the operation of a smart grid. Each of these stakeholders belongs to one of the domains shown in the NIST smart grid model (see Figure 2.3). The important part for this thesis is the interaction between customers, located in the bottom-right of the figure, and the remaining parties. In EI, the border between the domain of the customer and the other domains is called Energy Service Interface (ESI). As mentioned in previous sections, ESI is just another term for SGUI [24].

The specification defines an architecture and an XML-based message format to exchange information about price, reliability, and emergency conditions via the communication interfaces. The data structures support real-time interactions, forward projections, and historical reporting. Underlying protocols and technologies, which are used to transport the messages, are not explicitly defined [24]. An obvious and common option for the transmission of XML data is the usage of the Hypertext Transfer Protocol (HTTP) in combination with Web services (WS). But also XMPP could be an interesting alternative.

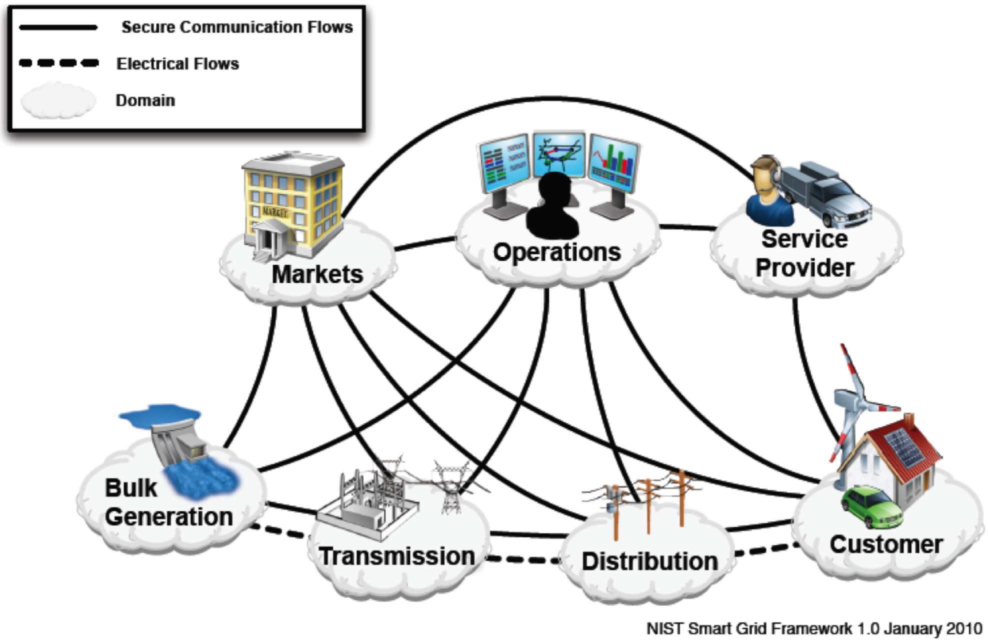

Figure 2.3: Smart Grid Model [25]

In general, EI supports two different kinds of interactions, namely transactive interactions and event interactions. A typical example for the first kind is shown in Figure 2.4, where party B sends a tender (offer) to buy or sell something to party A, and party A sends an acceptance in form of a transaction back to party B. For this type of interaction, the communication partners are always referred to as parties. More precisely, the sender is called party and the receiver is called counter party. The EI standard supports four essential transactive market activities between parties:

Indication of interest can be used to seek parties for a demand response transaction or for an energy sale.

Tender is an offer to buy or sell some service.

Transaction follows from the acceptance of a tender.

Execution is used for the execution or the delivery of the subject of a transaction.

Except for the indication of interest, transactive market activities are binding. Transactions and executions are the result of an accepted tender. The counter party is also allowed to decline the tender by sending a cancel tender message instead. Each communication partner is authorized to initiate the negotiation for a possible transaction at any time. This means that transactive interactions allow a versatile trading of services and energy between arbitrary partners. In contrast, event interactions are used to execute already concluded contracts. The EI specification

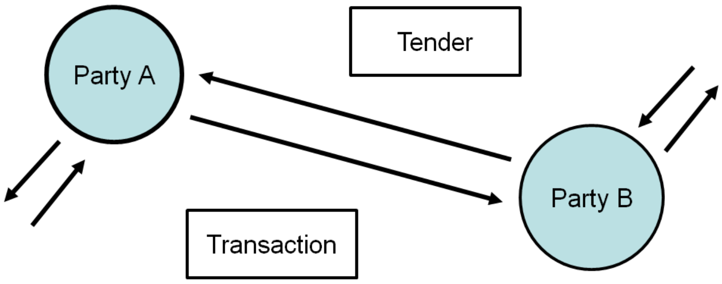

Figure 2.4: Typical transactive interaction between parties [24]

describes the purpose of events as dispatch of generation resources or curtailment resources. Curtailment resources provide reductions in delivery to a customer. Note that in some sense these resources are also generation resources as lowering the load has a similar effect as generating additional energy. Definition of the baseline for the reduction is not within the scope of the standard. An obvious example for the baseline is the current or planned energy consumption. A typical application for events are load sheds in case of imminent shortages or blackouts. Event-based communication always takes place between one designated Virtual Top Node (VTN) and one or more Virtual End Nodes (VEN). Events are always transmitted from a VTN to the VENs. However, this does not mean that messages are always pushed to the VENs. EI also allows pulling events from a VTN. Additionally, the role allocation is not fixed. It is possible that a party acts as VEN for one interaction and as VTN for another one at the same time. This definition allows complex event communication hierarchies like in the example shown in Figure 2.5. Moreover, both roles are allowed to participate in event-based and transactive-based interactions concurrently [24].

All interactions within EI are encapsulated in services. Apart from the two preceding types of interactions, which are part of the transactive and the event services, EI supports services for reporting, enrollment, and event support. Some of them are composed of smaller sub-services. For example, the transactive services are subdivided into registration, pre-transaction, transaction and post-transaction. Reports are suitable to convey sensing or status information back to the requesting party. A special case of reports are so-called projections, which allow the transmission of future reports. All reports are either associated with an preceding event or with a report request. Event support services are used for indicating whether the VEN is available for executing events and for joining or leaving DR programs that use events. Enrollment establishes an identity for a party, and therefore builds the basic relationship for further interactions. Most of the operations within these services follow the concept of a request and a subsequent response. The only exceptions are broadcasts, which exist for tenders, quotes (pricing information) and events. They are realized by an unacknowledged distribute request. There are multiple types of requests, but not all of them are defined for all services, e.g. create, cancel, and update. Each of them has a specific response operation. The concrete structure of the response is dependable

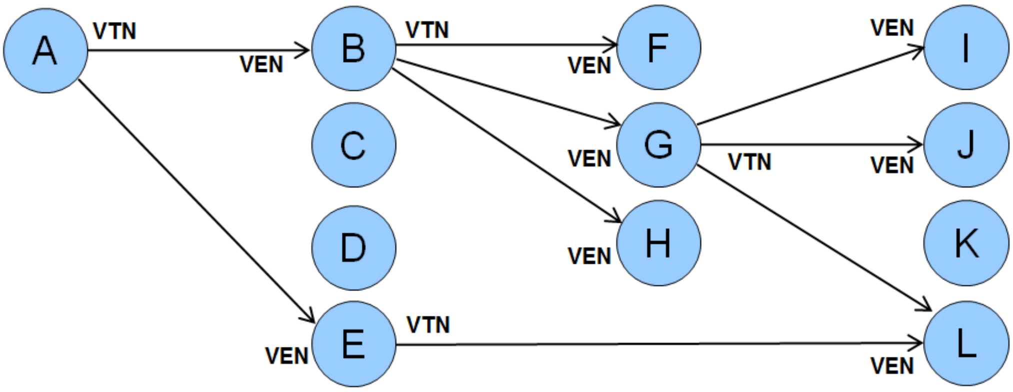

Figure 2.5: Network of multiple VTNs and VENs [24]

on the preceding request. However, all of them include a reference ID and a response code. The code provides information about success or failure, while the reference ID identifies the associated request [24].

Finally, the usage of EI is not restricted to electricity. It is also suitable for the communication infrastructure of other energy forms like natural gas. In summary, the EI specification describes an XML-based information and communication model including all parties participating in the smart grid like customers, service providers, markets, transmission agents, distributors, operators and bulk generation agents [24], [25].

2.2.3 Open Automated Demand Response

Already in 2002, the California Energy Commission Public Interest Energy Research Program funded a research program at Lawrence Berkeley National Laboratory (LBNL) that led to the development of the first version of the Open Automated Demand Response (OpenADR) communication specification. The main objective was to optimize the interaction between electric supply and demand. This includes DR events for load shedding and shifting as well as dynamic pricing to improve the economics and reliability of the power grid. For this purpose, OpenADR defines a non-proprietary and standardized interface for the transmission of required information. The classic version of OpenADR has used a so-called Demand Response Automation Server (DRAS) for the two-way communication between service providers and customers. An example for this concept is shown in Figure 2.6. It can be seen that utilities and customers use Application Programming Interfaces (API) for data exchange with the DRAS [2], [11].

The second version evolved from the OpenADR standardization process as part of the OASIS EI standard. Today, OpenADR 2.0 is available in form of EI profiles. This means that it is a subset of the EI specification. A sketch of the standardization process is presented in Figure 2.7. It shows that the development was forced by contribution from many different sources.

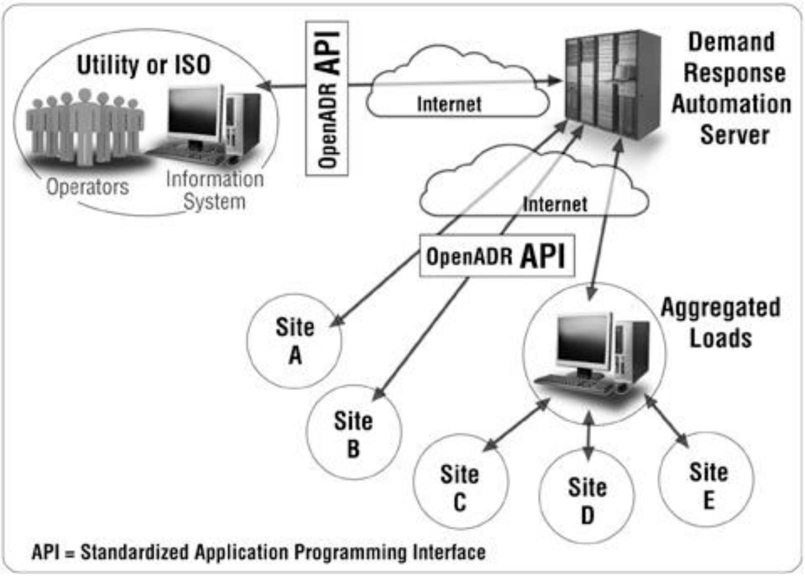

Figure 2.6: OpenADR 1.0 communication architecture [11]

While the primary source was OpenADR 1.0, other entities like the North American Energy Standards Board (NAESB), the ISO/RTO Council (IRC), and the OpenADR Task Force have provided further requirements. Certification of devices for one of the profiles is provided by the OpenADR Alliance. The OpenADR 2.0 profiles differ in the coverage of EI services presented in the preceding section. Currently, the most extensive profile is called 2.0b. While profile A supports only very simple events, profile B additionally implements event services, event support services, enrollment services and report services. Quotes and therefore dynamic prices are not yet part of the specification but are planned for future releases. Also the availability service, which is part of the event support services, is currently not included but scheduled for further releases [2], [11].

As subset of EI, OpenADR defines XML signals which have to be transported via some underlying protocol. In contrast to EI, the profiles describe two possibilities for this protocols, namely HTTP and XMPP. As described before, EI events, and therefore also OpenADR events, are always transmitted from a VTN to one or more VENs. It is mandatory for all VTNs to support the HTTP protocol. When a VTN shall be certified for profile B, it is also required to implement an XMPP interface. For a system or device that is only used as VEN, it is allowed to implement HTTP, XMPP, or both at once. This ensures that each profile B VTN can communicate with all kinds of VENs [2].

The definition of underlying protocols is not the only part where OpenADR is more accurate than EI. For example, also response codes and event signal types are defined in much more detail

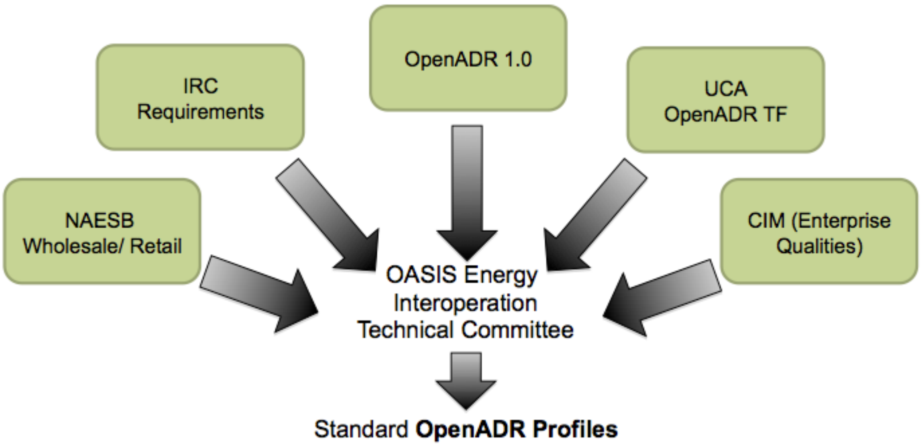
IRC Requirements
OpenADR 1.0
UCA OpenADR TF
NAESB Wholesale/ Retail
CIM (Enterprise Qualities)
OASIS Energy Interoperation Technical Committee
Standard OpenADR Profiles

Figure 2.7: OpenADR 2.0 standardization process [11]

in the profile specifications. While EI includes at least common classes for response codes, it does not contain any types for event signals to describe the format and meaning of included data. In OpenADR, the simplest type is called *SIMPLE*. It defines the levels 0 to 3 and therefore does not need any unit. An example for the usage of these levels is to change the operation mode of some devices from remote. Profile A event signals are restricted to this simple kind of event signals. The reason for that restriction is that OpenADR 2.0a is meant to be used for devices with very limited resources. Profile B defines more event signal types ranging from multiple kinds of electricity prices to load control [1], [2].

2.2.4 Common Information Model

The Common Information Model (CIM) is a series of IEC standards that was originally developed by the Electric Power Research Institute (EPRI) in the US. The CIM core specification (IEC 61970-301 [36]) specifies a language-independent UML model that defines classes and attributes representing the components of a power system. Additionally, the relationships (inheritance, association, and aggregation) between these components are part of the model. IEC 61970-301 covers the modeling of electrical networks from the perspective of the transmission system operator (TSO). Such a TSO is the organization that provides the national or regional electricity transportation infrastructure. Subsequently, IEC 61968-11 [35] was developed to extend the scope of CIM into the distribution network domain, including low and medium voltage networks. This part contains UML modeling for asset management, work management, customer management, maintenance and construction, or network planning. It also comprises modeling of data exchanged between systems within the distribution utility. For application of CIM in the area of deregulated energy markets, the CIM for Market Extensions was developed and published as IEC 62325-301 [37]. In contrast to the two previously described specifications, this part of CIM does not describe a model of the market structure itself. Instead, it specifies the data, which are exchanged between market participants. Another characteristic of IEC 62325-301 is that there are different extensions for European-style markets and for US-style markets. All these basic parts and several additional documents describing profiles and implementations of the CIM are maintained under the responsibility of IEC TC 57 within several workgroups. For example, WG13 works on IEC 61970-301, WG14 on 61968-11 and WG16 on the CIM for Market Extensions [17].

An important concept of CIM are profiles. There are more than 600 CIM classes and thousands of attributes and relations. Yet all of these attributes and relations are optional. This means, that, on the one hand, the overall model is quite complex and impractical to implement, and on the other hand, the data structures and therefore interfaces between applications are not explicitly defined. The solution to this problem is provided by the definition of profiles. Such a profile is a restricted subset of CIM classes, which brings the CIM in some context. Thereby, the information model has been used to derive a contextual model. It also defines the cardinality of relations and specifies which attributes are mandatory. There are already IEC standard documents that specify pre-defined groups of profiles. An example is the Common Power Systems Model (IEC 61970-452) which defines profiles for equipment, topology, state variables, or analog and digital measurements for the transmission environment. The deviation of self-defined profiles is supported by tools such as the open source CIMTool[1] by Langdale Consultants. Finally, the contextual model is used to derive an implementation model. This is either an XML schema, an RDF schema, or a database schema. The creation of XML and RDF schema files is also supported by the CIMTool [17].

Although the CIM model does not define an official common super-class which is extended by all other classes, most of them are derived from *IdentifiedObject*. The purpose of this class is to provide attributes for a unique identification of objects. The attribute that serves as identifier is called *Master Resource Identifier* (MRID). Additionally, any number of human readable names can be defined for an object derived from IdentifiedObject [17]. EMIX and therefore also EI and OpenADR, are using semantic elements from the CIM model. Also the naming conventions for requests (e.g. create, cancel) and responses (e.g. created, canceled) in EI and OpenADR are consistent with the CIM [24], [23].

2.2.5 Extensible Messaging and Presence Protocol

The task of XMPP is to transmit relatively small pieces of XML data, called *stanzas*, from one client to another nearly in real time. It is an open technology that is maintained and published by the Internet Engineering Task Force (IETF) within their Request for Comments (RFC) series as RFC 3920 [30] and RFC 3921 [31]. Additionally, many extension specifications are provided by the XMPP Standards Foundation. These extensions are called XMPP Extension Protocols (XEP). Some important services, which are defined in these RFCs and XEPs are one-to-one (unicast) messaging, multi-party (multicast) messaging by the use of multi user chats (MUC), service discovery, presence, contact lists, channel encryption, and authentication. Most of these

[1] http://wiki.cimtool.org/index.html

exemplary services are already defined in the RFCs while multi-party messaging is an example for an extension (XEP 0045). Nowadays, there exists a large variety of applications which use XMPP for data exchange. Well-known XMPP-based products are Google Talk IM and Google Voice over Internet Procotol (VoIP) services. But also other global companies like Apple, Cisco, IBM, and Sun use XMPP in their products [33].

XMPP uses a decentralized client-server architecture that also allows server-to-server communication. Figure 2.8 shows an XMPP network with three servers and six clients. Such a network is quite robust because it does not have a single point of failure. Further improvements in reliability can be achieved by server clustering. Additionally, most of the complexity can be concentrated on the servers. For example, user authentication is performed on the server infrastructure. Each client establishes a connection to a distinct XMPP server, where its user account has been created. In the following, this server is called the local server. In contrast to traditional technologies like HTTP, there is no need to open a new TCP connection for each transaction. Instead, a session with the local XMPP server is started by opening a long-lived TCP connection and negotiating XML streams in both directions (client-to-server and server-to-client). Such a stream allows an asynchronous data transfer, where data is pushed to the receiver. If a client wants to send some information, it is always sent to the local XMPP server by the use of such a stream. Afterwards, the transfer to the target client is performed by the server. If the information shall be transmitted to a client that is registered at another server, the local server performs a DNS lookup and forwards the data directly to the corresponding XMPP server. This means, that there is no multi-hop routing between servers. Communication from one server to another is always performed in one step (one hop). Thus, XMPP follows a so-called direct federation model. Subsequently, the remote server pushes the message to the desired target client. Finding the correct server is supported by the structure of the client addresses, called JabberIDs (JID). In combination with the password, the JID is also used for authentication. A JID, or more precisely a bare JID, looks like an email address, i.e. *user@domain.tld*. Therefore, it is case-insensitive and consists of a local and a global part. The local part is the actual username within a particular domain while the global or domain part is used to find the correct server. This allows the usage of the existing DNS infrastructure. In addition to the bare JID, there is the full JID, which adds the concept of resources and looks like *user@domain.tld/resource*. A resource is a string that is added to the end of the JID. In contrast to the local and global part, the resource part is case-sensitive. It allows the identification of a particular connection of a device or service to the server. In this context, it is possible to establish multiple connections with a single user account [33].

There are three fundamental types of XML stanzas, namely message, info/query (IQ) and presence. Each of them is represented by an XML stanza element with attributes and optional child XML elements. The XMPP message stanza is the basic method to push data to another client. It is used for “fire and forget” communication, which means that no acknowledge is required. Typical applications for messages are instant messages, group chats, alerts, and notifications. The XMPP specification defines five types of messages [33].

- *normal*: This type is used for a simple, single message to one receiver, which is similar to

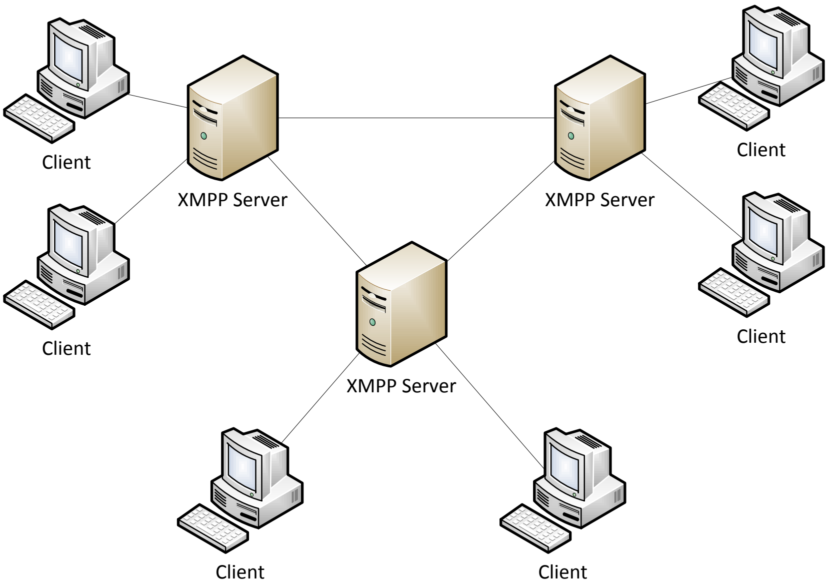

Figure 2.8: XMPP client-server architecture

an email message.

- *chat*: Messages of this type are exchanged in chat sessions, which correspond to instant messaging.
- *groupchat*: This type is used for messaging in chatrooms with multiple users.
- *headline*: This type is intended for transmission of alerts and notifications.
- *error*: The entity that detects an error for some communication, returns a message with type error.

In addition to the type, messages contain *from* and *to* attributes and an optional *id*. While the receiver has to be defined by the sending client, the *from* attribute is assigned by the server. This ensures, that the identity of the sender cannot be falsified. Inside the message element, the XMPP core specification defines the XML elements *body* and *subject*, which can occur multiple times, e.g. for texts in multiple languages. Usually, the *body* element contains the actual data to transmit. Apart from that, it is possible to define custom payloads, which can be used instead or together with the predefined elements.

The next type of XML stanzas, the IQs, are intended for request-response communication. This means, that the receiver always has to send a reply back to the sender. Similar to messages, IQs have *from*, *to*, *id*, and *type* attributes. However, the id is no longer optional, but is required to link request and response together. The *type* attribute allows the distinction of the following IQs.

- *get*: This type is used to ask the receiver for some information.
- *set*: IQs of this type are used to provide some information for the receiver or to make a request, which has to be processed.
- *result*: The result type is used by the receiver of a *get* or *set* IQ to return the requested information or simply as acknowledge.
- *error*: If the responding entity or an intermediate entity, such as an XMPP server, is unable to process the get or set request, it uses the *error* type to inform the requesting entity.

A typical interaction with get, set and result IQs is shown in Figure 2.9. In contrast to messages, the specification allows only one payload element within the IQ stanza element. The payload element defines the request to be processed or the action to be taken by the receiver. Additionally, it must be assigned to an XML namespace by the use of the *xmlns* attribute.

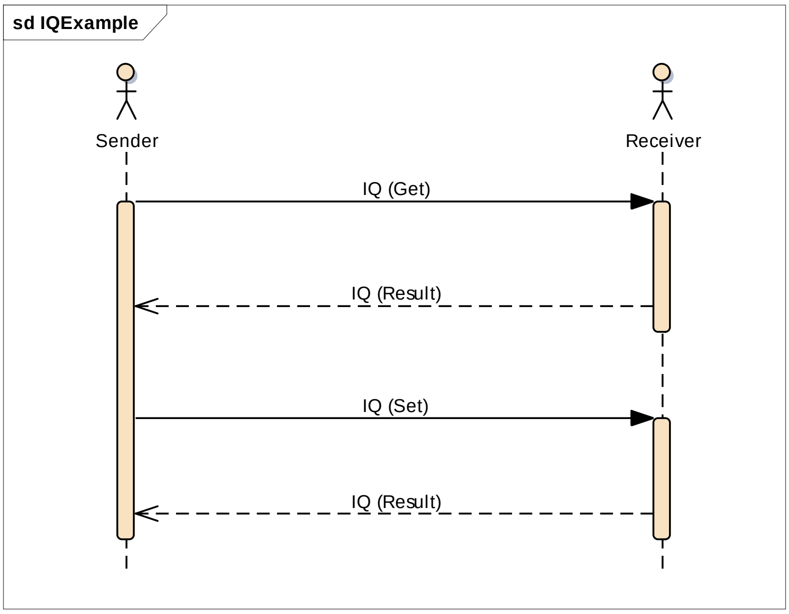

Figure 2.9: Example for IQ interaction

The third type of stanzas, the presence stanza, indicates whether other entities are online and available for communication. Also a personalized status text can be included. The notification of clients about the presence status of other participants is done by the XMPP server. To prevent unwanted or foreign users from gaining the presence information, XMPP uses a publish and subscribe method [33].

CHAPTER 3

Use Cases

This chapter contains relevant use cases for the communication between electricity customers and the smart grid. The main focus is on functional use cases which are referred to as primary use cases. They are used to model the actual information exchange along the interface between a smart building and the smart grid. In a first step, an introduction to the involved communication partners or agents is given. Then follows the detailed description of the use cases. The chapter concludes with some approaches and thoughts according to configuration tasks.

3.1 Agents

There are five important types of agents that appear in the identified use cases. More specifically, each of them is a system, which is driven by some operator. The important operators within this thesis are grid operators, energy retailers, energy pool managers, and customers. The assignment of the systems to the operators is shown in Figure 3.1. Additionally, the figure shows two examples for supporting services together with their corresponding operators. These are a weather forecast system and a system which supports the configuration process of devices at the customer. For future applications also other supporting communication partners like open data sources are possible. As many high consuming facilities like the heating, ventilation, and air conditioning (HVAC) system or lighting depend on weather conditions, weather forecasts will play a major role for optimizing the energy consumption. But even though such supporting services have a high impact on energy usage, they are not part of this thesis.

3.1.1 Customer Energy Management System

The customer energy management system (CEMS) is an internal automation facility at the customer's site, which optimizes and plans the energy usage throughout the day. The planning is influenced by internal as well as external information. Internal data come from the building automation system while external information describes data from the smart grid or weather

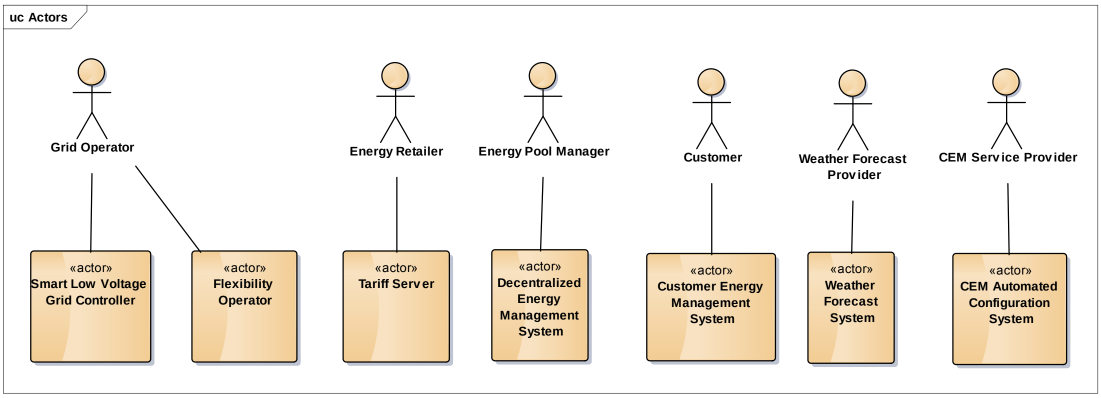
uc Actors
Grid Operator
Energy Retailer
Energy Pool Manager
Customer
Weather Forecast Provider
CEM Service Provider
«actor» Smart Low Voltage Grid Controller
«actor» Flexibility Operator
«actor» Tariff Server
«actor» Decentralized Energy Management System
«actor» Customer Energy Management System
«actor» Weather Forecast System
«actor» CEM Automated Configuration System

Figure 3.1: Communication agents together with supporting services

forecasts. Besides these information, it is important to take preferences of the customer into account for the planning activities. These preferences can be given in the form of upper and lower bounds for operating parameters. For reasons of simplicity, these bounds can be defined by modes like "*comfort*" or "*economic*". However, the CEMS does not only receive information from the grid. It also provides data for other agents. Examples for information which is provided towards the grid are current and future energy consumption or flexibility offers. Flexibility offers are bids to shed or shift loads. Since all communication between smart buildings and smart grids takes place between the CEMS and the other smart grid agents, the CEMS is the central agent in this work [12] [29].

3.1.2 Tariff Server

The tariff server provides price information for customers. It is operated in the sphere of an energy retailer. The server sends pricing signals including the energy price for some hours in advance to all registered CEMSs. As response it receives energy balance forecasts from the CEMSs. These forecasts contain the scheduled energy consumption for the same period of time as the pricing signal and can be used by the energy retailer to manage purchases on the global energy market. All the following use cases will use day-ahead pricing. This means that the signal includes the price for the next 24 hours and it is sent exactly once a day [29].

3.1.3 Flexibility Operator

The flexibility operator is localized in the sphere of a grid operator. It receives energy balance forecasts from the CEMSs and uses them to identify future bottlenecks in the grid. If such a situation occurs, the flexibility operator negotiates trades of flexibilities with the CEMSs to eliminate the problem. The term "*flexibility*" is defined as sheds or shifts of load or production [29].

3.1.4 Decentralized Energy Management System

The decentralized energy management system (DEMS) is operated by the energy pool manager and tries to buy flexibilities from the CEMSs, as well. In contrast to the flexibility operator, it does not use them to prevent bad situations in the grid, but to trade flexibilities on the (global) energy market [29].

3.1.5 Smart Low Voltage Grid Controller

The smart low voltage grid controller (SLVGC) is part of the grid operator domain. It monitors the current grid quality parameters of the power grid, e.g. voltage levels and power. If a problem is recognized, appropriate instructions are sent to the CEMSs [29].

3.2 Primary use cases

In Figure 3.2, an overview of the primary use cases is shown in combination with their interactions and dependencies. Additionally, the diagram shows a security use case, which is included by all other use cases. This use case is responsible for a secure communication between the involved agents and therefore contains encryption and authentication. Nevertheless, details concerning the security use case are omitted in this thesis and should be addressed in future work.

3.2.1 Energy balance forecast

Energy balance forecasts are used by the CEMSs to provide information about future energy consumption. The forecasts are partitioned in time slots of a fixed length (e.g. 1 hour). Each slot contains the projected energy usage for the particular time interval in kilowatt hours (kWh). The message itself is part of a unicast communication which has to be acknowledged. If this acknowledge does not arrive within a predefined interval, the sending of the energy balance forecast has to be repeated. Indeed, the number of retries has to be limited to prevent the CEMSs from flooding the network with lost signals. A visualization of the use case can be found in Figure 3.3.

3.2.2 Day-ahead pricing

Once a day (e.g. at midnight) each CEMS receives a day-ahead pricing signal containing the price per kilowatt hour for 24 hours in advance. An important difference between such a pricing signal and the pricing situation commonly used nowadays is that the day-ahead pricing signal is not constant throughout the day. It is a so called time-of-use (TOU) tariff. Here, the day is divided into time slots with potentially different electricity prices. If the price is higher in peak time slots and lower in non-peak time slots, customers will try to shift their energy usage towards non-peak time slots. Therefore, up to a certain extent energy retailers can use the pricing signal to control their customer's energy usage throughout the day [18].

Based on the day-ahead pricing signal and the planned energy consuming tasks within the building, the CEMS calculates a cost-effective schedule for the use of energy. It is beneficial to

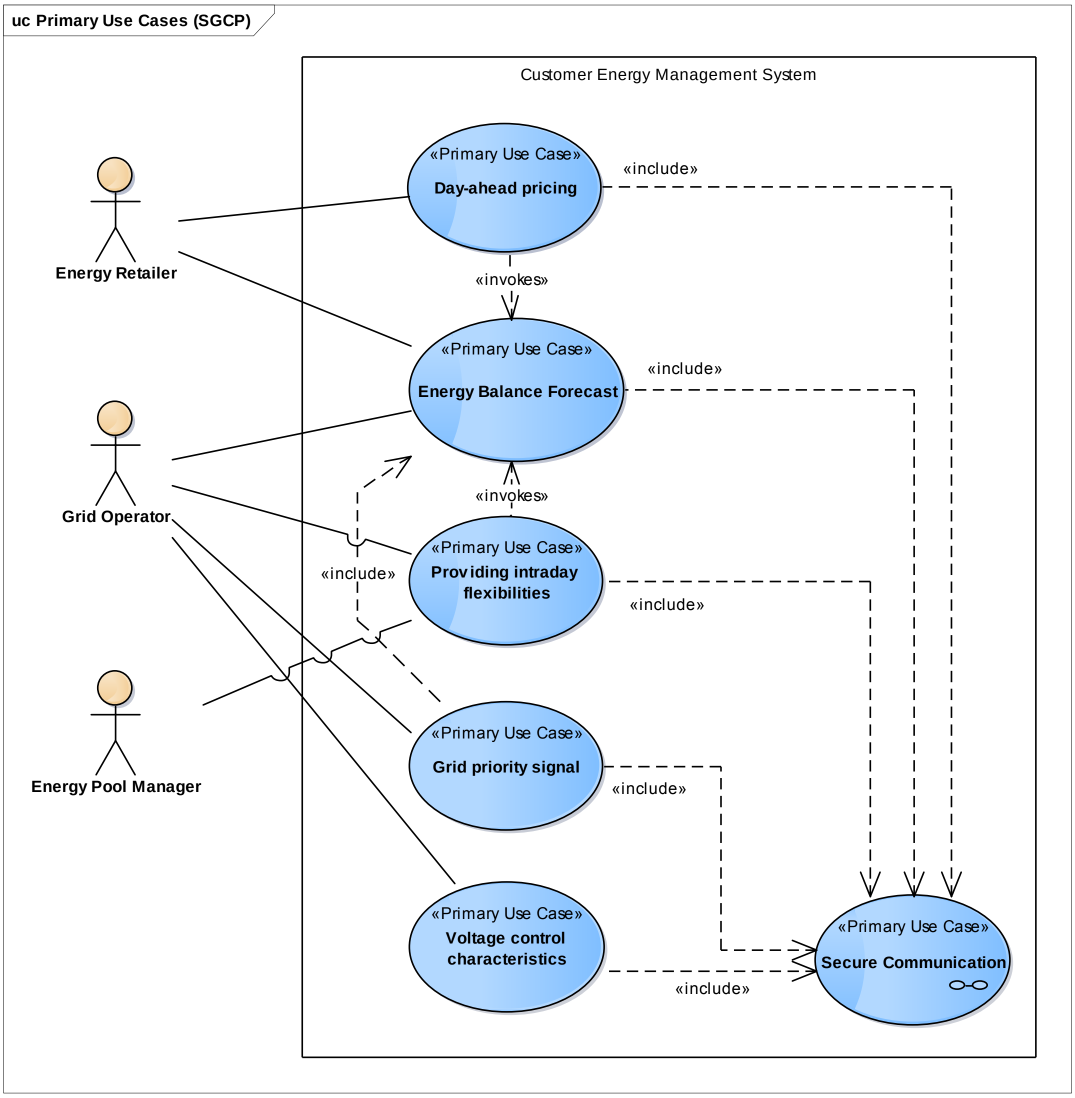

Figure 3.2: Primary Use Cases Overview

send the pricing signal with some time gap between sending time and the first time slot within the signal. This ensures that the CEMSs have enough time to plan their energy schedule before the new prices are applied. Based on this schedule, the energy balance forecast for the next 24 hours is calculated. The forecast is sent back to the tariff server. Additionally, the forecast is transmitted to the flexibility operator, that needs the information to recognize bottlenecks in the power grid. Both get the forecast immediately after finishing the calculation of the schedule. The whole process is shown within the use case diagram in Figure 3.4. The pricing signal is sent to a potentially large number of receivers. Therefore, the day-ahead pricing signal is transmitted

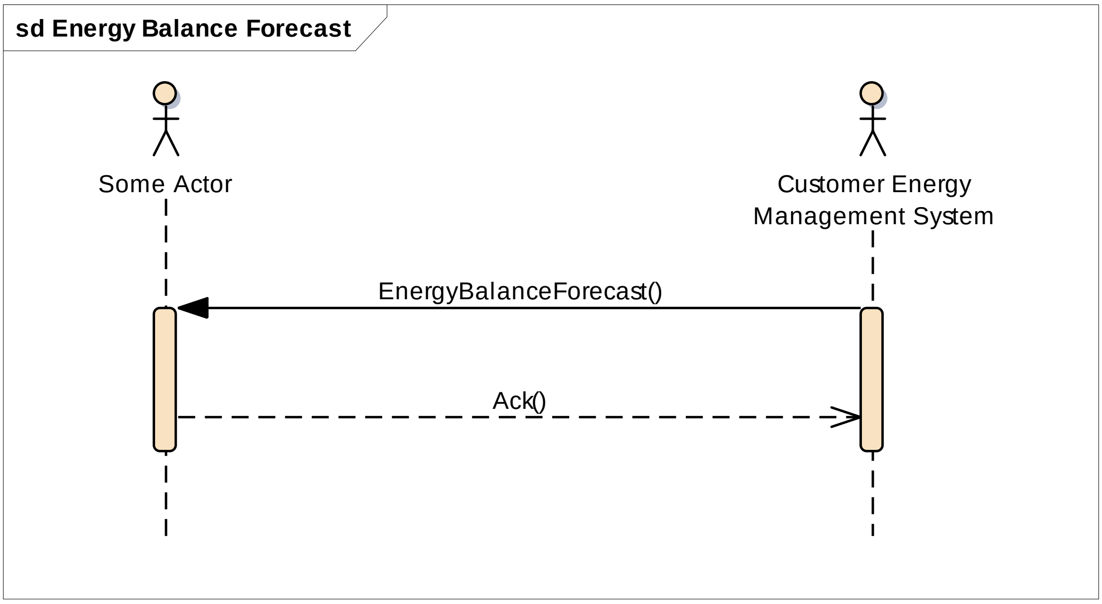

Figure 3.3: Energy balance forecast use case diagram

as unacknowledged multicast or broadcast message. Details about the energy balance forecast can be found in the previous section.

3.2.3 Providing intraday flexibilities

If the flexibility operator detects bottlenecks in the grid for one or more time intervals, the CEMSs are asked to provide flexibility offers for these specific time frames. The identification of bottlenecks relies on previously received energy balance forecasts. As the flexibility request is sent to multiple CEMSs, there are two possibilities to transmit this signal. First, it can be sent as an unacknowledged multicast or broadcast message. This alternative has the disadvantage that it is relatively elaborate to reach only a variable subset of the CEMSs. But, if a bottleneck for a certain part of the grid is recognized, it is unnecessary to transmit the request to CEMSs which are localized outside that problem area. For the actual achievement of the objective to remove the existing bottleneck it makes no difference whether all or only the desired targets get the signal. The flexibility operator can simply decline all irrelevant flexibility offers. But, for reasons of efficiency (utilization of the CEMSs, network capacity) it is better to restrict the request to problematic areas of the grid. Therefore, the second and better option is to use multiple unicasts. Subsequently, the CEMSs check if it is possible to comply with the flexibility request. It is necessary to ensure that flexibilities are only offered to such an extent that all tasks of the customer can still be executed within given thresholds. If this is possible, the CEMS sends one or more flexibility offers back to the suppliant. Otherwise, the CEMS has to inform the flexibility operator that no flexibilities can be offered. Each offer contains a schedule which describes how the energy usage (or generation) would change compared to the previously transmitted energy

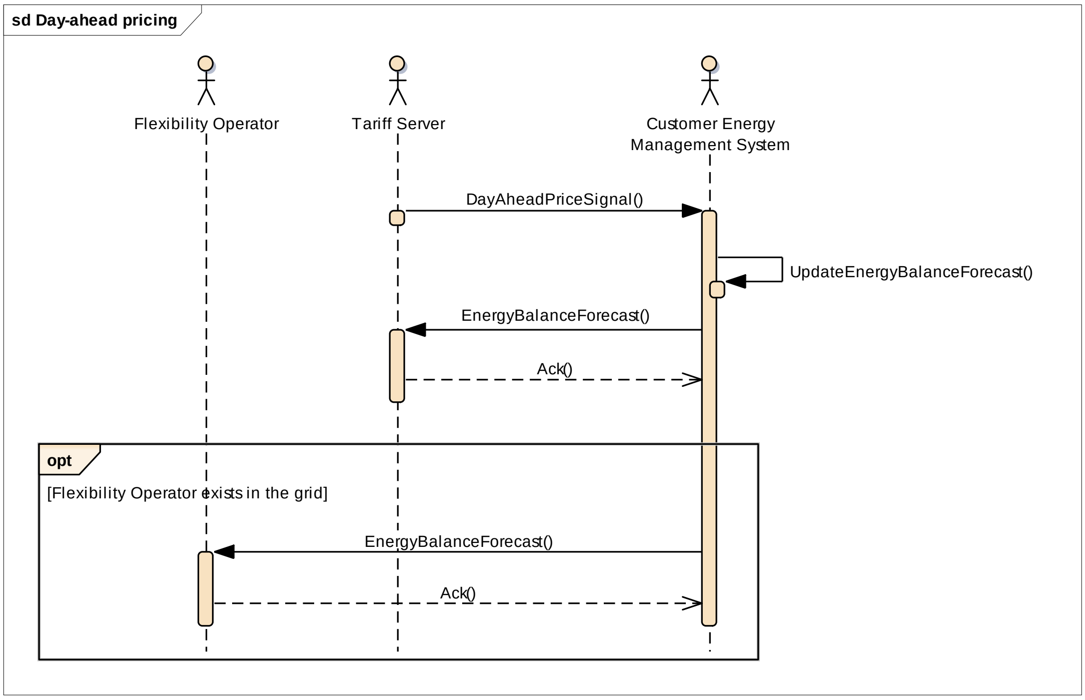

Figure 3.4: Day-ahead pricing use case diagram

balance forecast. Additionally, every offer has a price to which the flexibility is sold. In general, this price needs to be higher than the cost for saved energy in the corresponding time window. In the next step, the flexibility operator checks the received offers. Even though each CEMS is allowed to send more than one offer for a particular flexibility request, the flexibility operator is only allowed to order one offer per CEMS. Based on previously received energy balance forecasts and the incoming offers, it is possible to calculate how the overall energy consumption in the grid would look like for all valid combinations of accepted offers. If there is a suitable ensemble of offers, corresponding flexibility orders are sent back to the affected CEMSs. All offers, which are not ordered, have to be declined. Sometimes, it might be the case that no CEMS is able to offer flexibilities or that all possible combinations do not help to remove the bottlenecks. Then the use case ends at this stage. Received bookings are stored at the CEMS. Depending on a flag in the initial flexibility offer, the CEMS applies the schedule of the ordered flexibility immediately or after an explicit release of the order sent by the flexibility operator. All messages have to be acknowledged. If an acknowledge does not arrive within some time frame, the signal has to be sent again. After a defined number of retries, the procedure has to stop. Figure 3.5 shows the UML sequence diagram for this use case. For the sake of clarity, the special case, where the CEMS has no flexibilities to offer, is not shown in the diagram.

A slightly changed version of the use case is shown in Figure 3.6. The communication regarding the negotiation of flexibilities is the same as before, but the flexibility request comes from a DEMS. This operator does not use flexibilities to prevent problems in the grid. Instead, the DEMS trades them on the global market. Thus, no energy balance forecasts are required in advance. Moreover, the selection of offers does not depend on grid conditions. The DEMS only buys offers that are economically attractive. Apart from this, ordering flexibilities works exactly as before. It is only allowed to book at most one offer per CEMS and all other offers have to be declined.

3.2.4 Grid priority signal

The smart low voltage grid controller (SLVGC) monitors the current voltage and power levels in the grid. In case of emergency, like voltage band violations, so-called grid priority functions are used to transmit instructions to the CEMSs. For example, a sudden change in load on the grid may lead to excess voltage. The exchanged messages do not belong to market demand response mechanisms anymore. They are a classical instance of physical demand response activities. If and how each CEMS reacts to the instructions depends on the contracts between customers and grid operator. In general, physical demand response requests are binding. As described in Chapter 2, this kind of demand response should only be used if market mechanisms are not sufficient to prevent problems. The impact on processes of the customer is relatively high, but still better than a power outage.

If a problem is recognized, e.g. with voltage or power levels, a grid priority signal containing instructions to reduce or increase the load is sent to the CEMSs in the affected area of the grid. The degree of required load change is provided relatively to the current load (e.g. reduce load by 50%). If the message is received correctly, an acknowledgment is sent back to the SLVGC. When it is safe to return to normal operation, the SLVGC sends a signal indicating the end of the emergency case. This message has to be acknowledged by the CEMSs as well. All signals are sent as unicasts and have to be repeated if the acknowledge does not arrive within a specific time window. The corresponding use case diagram is provided in Figure 3.7.

3.2.5 Voltage control characteristics

In some sense, voltage control characteristics are a more conventional method of grid priority functions. The CEMSs receive P(U) and Q(U) curves from the SLVGC, where P stands for the effective power, Q for the reactive power, and U for voltage. These curves are used as instructions how to change the power usage when the voltage levels deviate from the reference voltage. In both diagrams, the power values are provided in percentage of the current power usage. In general P(U) and Q(U) will be transmitted in a single message, which has to be acknowledged by the CEMS. When necessary, it is also conceivable to send only one of the curves. In this case, the signal should have the same structure, but one is discarded. This particular use case (see Figure 3.8) is focused on the problem of voltage band violations. However, it would be possible to use the same mechanism for the transmission of P(F) and Q(F) data with frequency F in order to deal with frequency band violations.

3.2.6 Attendance or weather changes

A less technical but also very important use case is the change in conditions in and around the customers' buildings, which affect the calculated energy balance forecast. A change in weather conditions or in the number of people in a building can have a significant impact on the energy consumption of home and building automation systems. For example, the ventilation or air conditioning system needs to run on higher intensity when the attendance is higher or when it is hot outside. Therefore, such changes will cause a recalculation of the energy balance forecast, which have to be retransmitted to the external agents. Figure 3.9 shows that this communication is fully covered by the energy balance forecast use case.

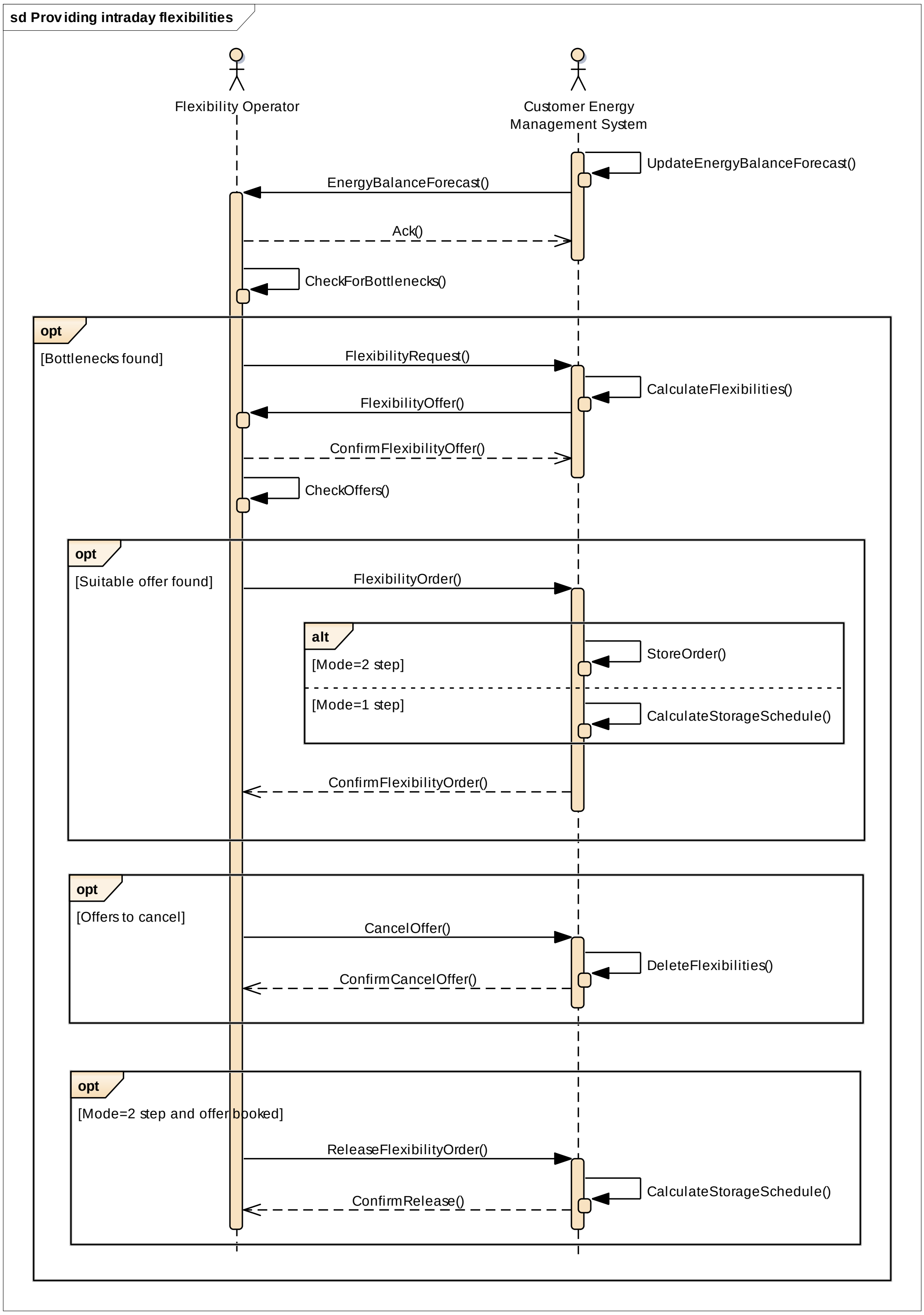

Figure 3.5: Providing intraday flexibilities (flexibility operator) use case diagram

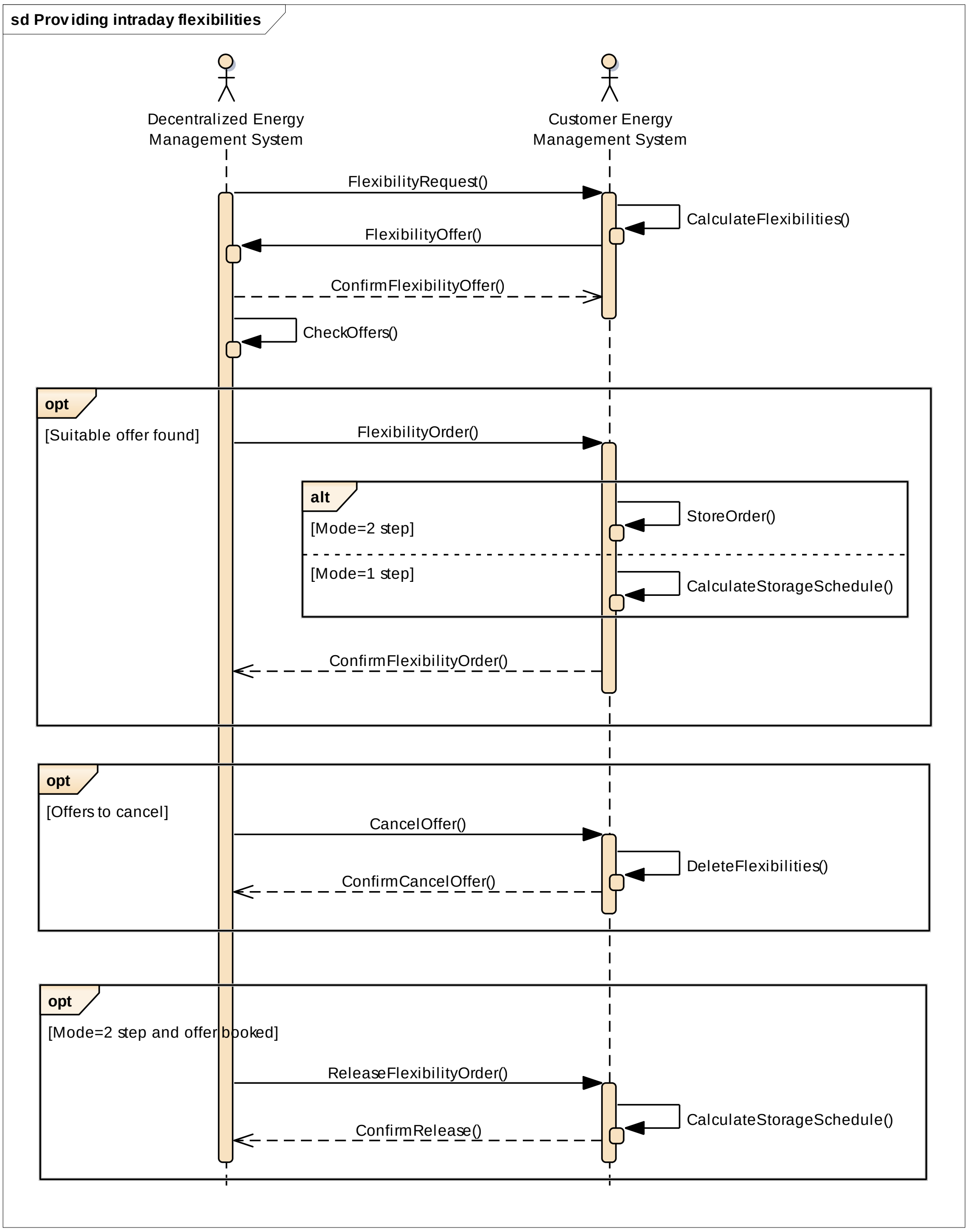
sd Providing intraday flexibilities
Decentralized Energy Management System
Customer Energy Management System
FlexibilityRequest()
CalculateFlexibilities()
FlexibilityOffer()
ConfirmFlexibilityOffer()
CheckOffers()
opt
[Suitable offer found]
FlexibilityOrder()
alt
[Mode=2 step]
StoreOrder()
[Mode=1 step]
CalculateStorageSchedule()
ConfirmFlexibilityOrder()
opt
[Offers to cancel]
CancelOffer()
DeleteFlexibilities()
ConfirmCancelOffer()
opt
[Mode=2 step and offer booked]
ReleaseFlexibilityOrder()
CalculateStorageSchedule()
ConfirmRelease()

Figure 3.6: Providing intraday flexibilities (DEMS) use case diagram

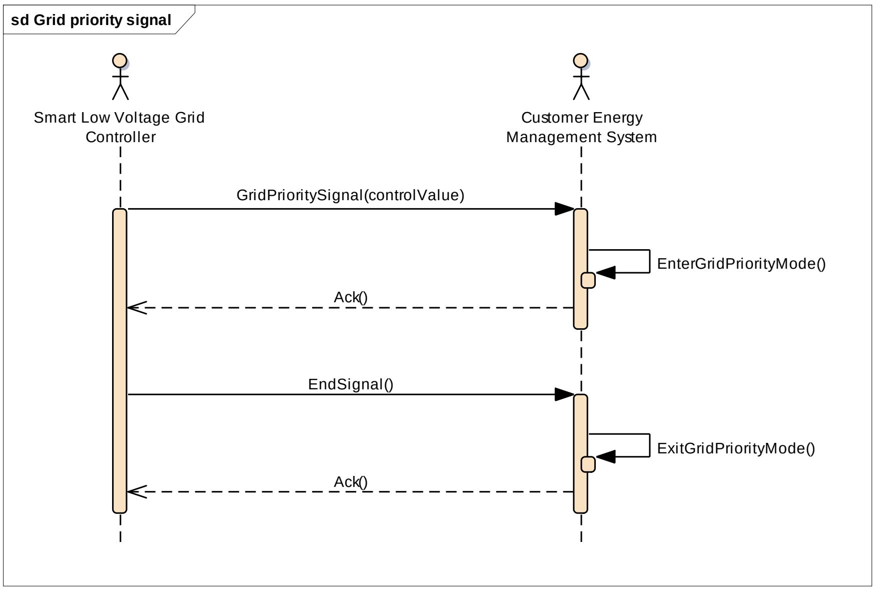
sd Grid priority signal
Smart Low Voltage Grid Controller
Customer Energy Management System
GridPrioritySignal(controlValue)
EnterGridPriorityMode()
Ack()
EndSignal()
ExitGridPriorityMode()
Ack()

Figure 3.7: Grid priority signal use case diagram

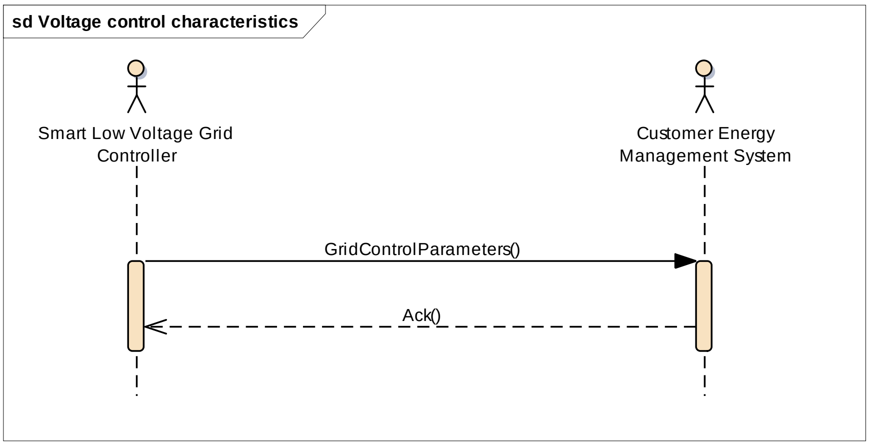
sd Voltage control characteristics
Smart Low Voltage Grid Controller
Customer Energy Management System
GridControlParameters()
Ack()

Figure 3.8: Voltage control characteristics use case diagram

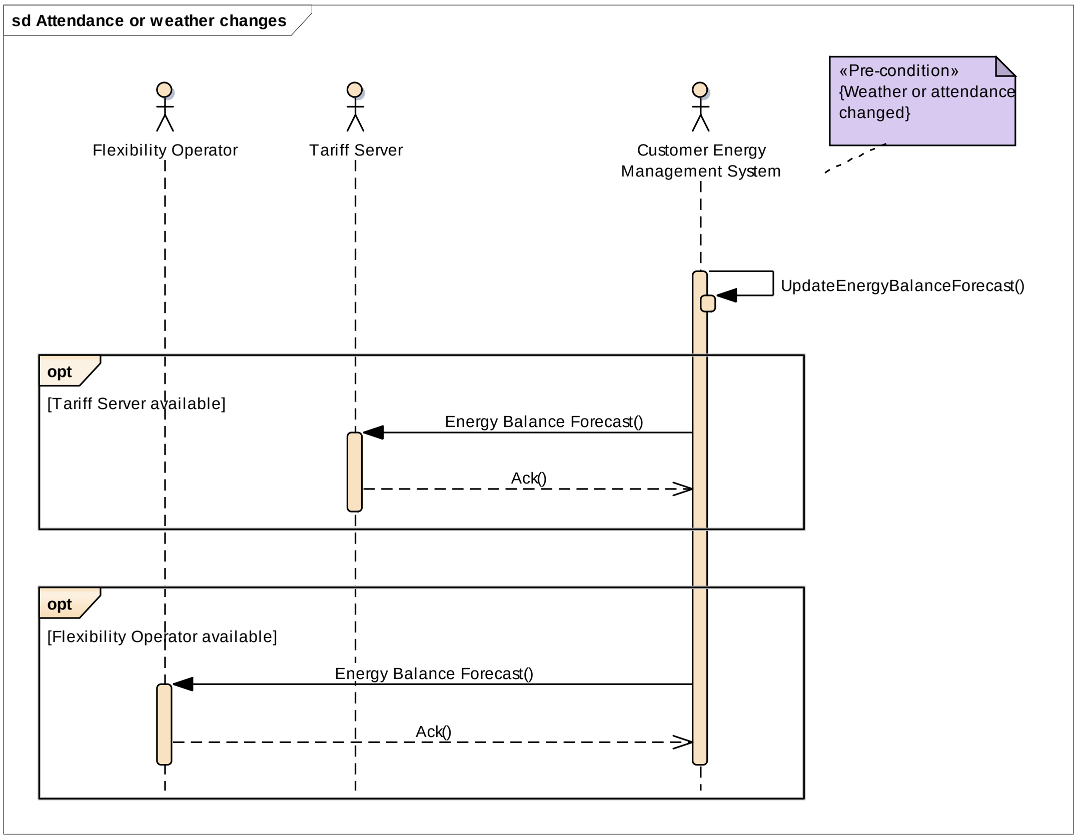
sd Attendance or weather changes
«Pre-condition»
{Weather or attendance changed}
Flexibility Operator
Tariff Server
Customer Energy Management System
UpdateEnergyBalanceForecast()
opt
[Tariff Server available]
Energy Balance Forecast()
Ack()
opt
[Flexibility Operator available]
Energy Balance Forecast()
Ack()

Figure 3.9: Attendance or weather changes use case diagram

3.3 Configuration

Since this work focuses on the communication of the CEMSs with all the other agents, only a possible configuration process of the CEMSs is described in this section. The first step for integrating a new CEMS is the registration process. As part of that task, a unique identifier for the CEMS has to be established. The actual form of that identifier will depend on the used network protocols. In general, there is a fixed relationship between the grid operator, that provides the infrastructure for electricity transmission, and the electricity customers. Therefore, it makes sense to establish the unique identifier at the conclusion of a contract with the grid operator. This also means, that the registration process does not need to be a fully automated task. Afterwards, a connection to some server in the sphere of the grid operator has to be configured at the client. The address of that server is identical for all customers sharing the same grid operator. This is also the case for the address of the smart low voltage grid controller and the flexibility operator. There are multiple possibilities of how these addresses can be made available to the CEMSs. An example is to provide them, together with the unique identification and required certificates, on a smart card or something comparable like a USB drive. Another possibility is, that the grid operator is responsible for providing pre-configured CEMS to the customers. Also a combination of these two, where only the identity of the customer is stored on a smart card, is an option.

Furthermore, the CEMSs need to know their communication partners that are not in a fixed relationship with the grid operator. These are the DEMS and the tariff server. Addresses and authentication information is first known at the conclusion of contract with an energy retailer or an energy pool manager. Therefore the use of smart cards, which store the mentioned information, is a suitable option, as well.

In Table 3.1, the configuration parameters for one possible solution, that uses smart cards and pre-configuration by the grid operator, are summarized. Note, that the actual form of the contact data for SLVGC, flexibility operator, tariff server, and DEMS depends again on the used network protocols.

Parameter	**Type**	**Storage**
Grid operator's server	IP address and port	Fixed
Unique identification	ID and Certificate	Fixed (Smart Card)
SLVGC	Address and/or ID	Fixed
FlexOp (0..1)	Address and/or ID	Fixed
Tariff Server	Address and/or ID	Subject to change (Smart Card)
DEMS (0..n)	Address and/or ID	Subject to change (Smart Card)

Table 3.1: CEMS configuration

CHAPTER 4

Signal definition and data model

The signals appearing in the previously defined use cases have to be transmitted through a communication network to reach their intended receivers. For this purpose appropriate network protocols and a data model have to be found. With regard to network protocols the focus is on the upper three layers of the ISO Open Systems Interconnection (OSI) reference model. These are session, presentation, and application layer. In the following, they are merged and referred to as application layer. Underlying layers are chosen by the needs of eligible application layer protocols and additionally by the degree of compliance with common requirements for SGUI communication, which are defined in Chapter 2. Afterwards, a data model that defines the exact format of the exchanged signals is specified. An important objective of this task is to make use of already existing standards. As one of the requirements states that all messages shall be based on XML, the specification process is restricted to technologies that support this data format.

4.1 Network protocols

In general, it is useful to utilize existing information and communication technology (ICT) infrastructure as far as possible. This approach keeps costs and efforts for development, enrollment, and maintenance lower than they would be for completely new developments and mechanisms. For example, the Internet infrastructure and its protocols are well suited for the exchange of data between electricity customers and service providers on the grid [19], [34]. Thus, only established and widely used technologies are taken into account.

4.1.1 Application layer

A popular technology which can be used for the exchange of XML-based data is HTTP in combination with Web services. However, the use of HTTP for communication between Smart Grids and Smart Buildings has some disadvantages. If an agent on the grid side transmits data to a CEMS, the customer has to accept inbound connections. According to the requirements defined in Chapter 2, this is not wanted. Additionally, standard HTTP does not support publish and subscribe methods where new information is pushed to the clients. On the other hand, XMPP represents a communication protocol which fulfills all requirements for data exchange via the SGUI. The push model of information transfer within an XMPP network supports good scalability, as there is no server which has to be continuously polled. This model also ensures that all resources are able to receive messages asynchronously. Long-living TCP connections are perfectly suitable for a large number of small data packets. Thereby, it is not necessary to establish a new connection for each data packet which should be sent. Furthermore, XMPP was initially invented for instant messaging, which also requires many small messages. Presence stanzas allow the determination of availability of systems participating in the network. They ensure that participants, who need to communicate with each other, get information about their current status. A unique and trusted identity for all systems is formed by the JIDs that are used as addressing scheme and for authentication at the XMPP servers. Secure authentication is provided via the Simple Authentication and Security Layer (SASL) defined in RFC 4422 [21]. Additionally, channel encryption is supported by the optional profile of Transport Layer Security (TLS). Note that channel encryption does not include content encryption. This means that confidential information should be encrypted before it is embedded into an XMPP stanza. Otherwise, the operators of the servers are able to read the messages. The decentralized architecture of XMPP consisting of several servers and the possibility of clustering avoid single points of failure and allow server-to-server communication. Also, the requirement of redundancy is covered by decentralization and clustering. As all messages are XML-based, it is easily possible to ignore information which is not understandable by a client. The XML parser just has to skip the attributes which are not known. Another advantage of XMPP is that all connections established by the clients are outbound. Hence, there is no need to open any inbound firewall ports. Only operators of XMPP servers have to open the corresponding ports. The communication network supports the addressing of single resources as well as group communications by the utilization of MUCs. XMPP provides two mechanisms of publish and subscribe, and they also fulfill the requirement of service discovery. On the one hand, there is the presence service, which is a specialized form of publish/subscribe. On the other hand, there is an extension known as *PubSub*, which allows the publication of offered services and the subsequent subscription to them. PubSub is defined in the XMPP extension XEP-0060 [22].

These characteristics of the XMPP protocol cover all requirements for SGUI communication except the prevention of denial of service attacks. This threat is discussed in an informational specification known as XEP-0205 [32] which provides potential solutions and recommendations for the protection of XMPP servers against such attacks. Examples for these recommendations are the limitation of bandwidth which can be used by a client within a given time interval, the restriction of the number of stanzas which can be sent in a particular interval of time, the limitation of the number of connection attempts, or constraints for the number of simultaneous connections from one IP address. Thus, XMPP has the potential to provide a suitable application layer in the communication network for a future smart grid. This and the fact that it is a popular open standard, which is used in many applications, lead to the decision to use XMPP for the transportation of messages.

4.1.2 Underlying network stack

Apart from the possibility to transport XMPP over HTTP, the native transport layer protocol (layer 4) for XMPP is the Transmission Control Protocol (TCP) [33]. The advantage of TCP compared to the User Datagram Protocol (UDP) is that it is connection-orientated and fully reliable [34]. One level below, at the network layer (layer 3), the Internet Protocol (IP) has been identified as the best choice. It is independent of underlying network technologies and has been evolved to a de facto standard for transmitting data over the Internet. IP provides a scalable architecture combined with appropriate QoS measures for latency, jitter, delay, and other parameters. Additionally, it allows easy network reconfiguration and supports unicast as well as multicast and broadcast services. As the underlying physical medium does not matter, any device that communicates via IP can be integrated into the smart grid infrastructure. Also the usage of different technologies in different network segments is supported. This means, that interoperability is one of the big strengths of IP. The upcoming Internet Protocol version 6 (IPv6) has the potential to further increase the advantages of the protocol. For example, IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) based on IEEE 802.15.4 can be used for short distances [20], [34].

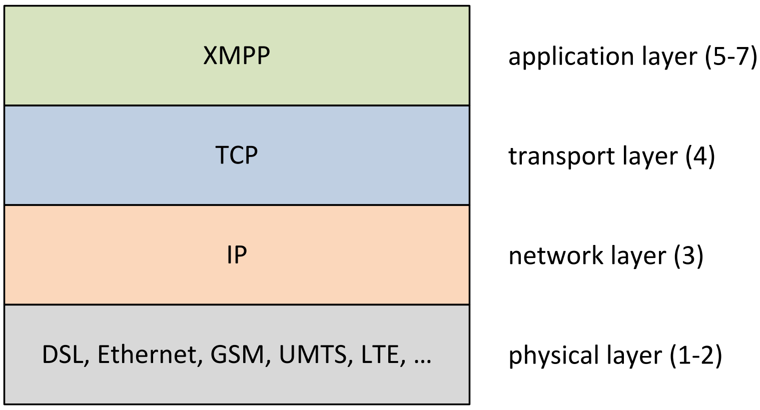

Figure 4.1: Network protocol stack

Layer 1 and 2 of the OSI reference model are merged into the physical layer. As previously mentioned, IP can be transported via any physical medium. The available media can be grouped into wired and wireless systems. Important wired technologies are Ethernet, powerline communcations (PLC), and Digital Subscriber Line (DSL). Especially the first two provide high bandwidth and robustness. In the wireless domain, popular options are IEEE 802.11 in all its versions, IEEE 802.15.4/ZigBee, and various types of cellular networks, e.g. Global System for Mobile Communications (GSM), Universal Mobile Telecommunications System (UMTS) or Long-Term Evolution (LTE). Depending on the geographical location and the area of application, each physical medium has its advantages and disadvantages. A possibility is to use DSL or cellular networks for wide range data transfer. Communication over short distances can also be realized with Ethernet, PLC, or IEEE 802.11 [15], [34]. The overall network protocol stack used in this work is shown in Figure 4.1.

4.2 Data model

After selecting XMPP as application layer protocol, appropriate XML schemes have to be found to define the structure of the signals, which will be embedded into the XMPP messages. The first step for this task is to identify the required information, which has to be carried by the various messages. For this purpose, all attributes needed by the intended receiver are described by an abstract name, which can differ from the actual data model, a short description, a rough data type, and the ordinality. The ordinality indicates if the information is required for one or more times, or if it is optional. A reason for optionality is that the receiver needs knowledge of the information, but there is a possibility to gain it from other sources. An example is that the information is already predefined in a business contract between the communication partners. In addition, it is investigated whether the standardized data models described in Chapter 2 are suitable to define the signals. In particular, the investigated options are EI, OpenADR, and CIM. In a second step, one of these standards has to be selected for implementation. If it is not feasible to use an existing data model for a signal, it has to be self-defined. It must be taken into account that most of the messages require a subsequent acknowledgment, which has to be definable by the same standard as the signal itself. Another crucial criterion for the final selection of the data model is the possibility to define all signals by the same standard.

Name	**Ordinality**	**Description**	**Type**
ID	Optional	The ID serves as unique identifier for the signal. The optionality comes from the fact that an ID identifying the message is also included in the XMPP message which is used to transport the signal.	String
Created	Optional	Represents the time at which the signal has been created by the sender.	Date and Time
Sender	Optional	Contains the unique identifier of the sender. This field is optional because the sender is also part of the XMPP message which is used to transport the signal.	String
Receiver	Optional	Contains an identifier for the receiver or a group of receivers. This field is optional because the receivers are already part of the XMPP message which is used to transport the signal.	String

Table 4.1: Common information required by the receivers

4.2.1 Common information

There are some attributes which are common to all signals containing a unique signal identifier, the identification of the sender, and the identification of the receiver or a group of receivers. All of them are important for the receiver, but they can also be part of the enclosing XMPP message header. Therefore, Table 4.1 defines the inclusion into the signal itself as optional. Additionally, a creation time can be included in the signals. Signal specific information is described in the following sections while the described common attributes are left out in these sections.

4.2.2 Day-ahead pricing signal

The day-ahead pricing signal has to convey pricing information over time for 24 hours in advance. As it should be sent to a potentially large number of clients (all customers using a particular electricity tariff), it is important to use a data structure which is appropriate for multicasting or broadcasting. Table 4.2 summarizes the information that is needed to propagate day-ahead prices. The prices itself are organized in time slots, which can occur arbitrarily often, but at least once. This allows the usage of the same data structure for pricing models with time slots of another length (e.g. 15 minutes). Also signals, which convey prices for less or more than 24 hours, can be created in this way.

Name	Ordinality	Description	Type
Start time	Required	This is the time at which the first price included in the signal gets effective.	Date and Time
Slot duration	Required	Specifies the duration of a price slot. A common option to describe duration is provided by RFC 5545 [7].	String
Prices	Required [1..n]	The prices define the actual price value for each time slot.	Decimal
Currency	Optional	Defines the currency of the contained prices. A way to represent currency is the usage of ISO 4217 currency codes. The knowledge of the currency is mandatory, but when it is predefined in a contract it is not necessary to send it with each day-ahead pricing signal.	String
Unit	Optional	Specifies the accounting unit, e.g. kWh. Like the currency, also knowledge of the accounting unit is mandatory. But also this unit can be predefined in a contract.	String

Table 4.2: Information needed in the pricing signal

In general, all standardized data models presented in Chapter 2 are suitable for the representation of this information. The intended service within EI to provide market prices to customers is called *EiQuote*. Note that quotes are only indications of the price, they are not actionable. The service supports the distribute operation, which allows broadcasting or multicasting of pricing information to several customers without acknowledgments. Apart from the *EiQuote* service there are two additional services within the EI specification which support the definition of the signal. First, the *EiTender* service could be used in the same way as quotes. The structure of a tender is similar to that of a quote and also the distribution operation is defined for tenders. However, the intention of tenders is to provide binding offers and not to propagate information. Second, as the announcement of new prices is a form of DR event, also the *EiEvent* services is suitable to transfer the prices to the customers. In this case, the tariff server would act as VTN while the role of the VENs is taken by the affected CEMSs. Also events can be sent as broadcast or multicast via the distribute operation. As the *EiEvent* service is part of the OpenADR profiles A and B, the day-ahead pricing signal can also be modeled with OpenADR. However, the structure of dynamic price information is too complex for profile A. Therefore, only the more advanced profile B is suitable for this kind of data. In addition to these options within EI and OpenADR, there are several ways to define CIM-based signals for day-ahead pricing. An option, which is also covered by the CIM European market model exchange profile (IEC 62325-351 [16]), is to embed the prices into a *TimeSeries* which is included in a *MarketDocument*. Note that IEC 62325-351 was developed for business-to-business market information exchange and not for communication with customers. Another possibility is to use a scheduling class, e.g. *RegularIntervalSchedule*. The temporal position and length of the intervals are defined by a single start time and a single slot duration. The duration applies to all intervals, which only need a sequence number to determine the ordering. Also units and a creation time can be included in this schedule. Furthermore, the *TariffProfile* class can be used. In principle, this class also defines a schedule, but it includes more redundancy, e.g. the units are contained in each interval.

4.2.3 Energy balance forecast signal

Energy balance forecasts inform a communication partner at the grid side about planned energy consumption for a particular range of time. The signal must be transmitted as an acknowledged unicast. The central component of the forecast is a list of time slots containing minimal and maximal consumption values. As there are no gaps between these slots, it is sufficient to send the start time of the first slot and the duration of a slot. All information needed by the receiving entity (e.g. energy retailer or grid operator) is described in Table 4.3.

An obvious option to exchange energy balance forecasts with EI is to use *EiReport* services in form of forward projections. The structure of these reports is ideal to represent the projected energy consumption for the next day. However, the problem with them is that they cannot be sent in the way it is required within the previously defined use cases. While sending of an energy balance forecast has to be triggered by a CEMS after it has finished the calculation of a new consumption schedule, EI reports have to be requested by the intended receiver before they can be transmitted. This means that the transmission of forecasts requires a service where the communication can be initiated by the CEMS. The most eligible option within the EI specification fulfilling this requirement is provided by the *EiTender* service. All other EI services are not convenient for the exchange of binding predictions of energy consumption. As the energy bal-

Name	Ordinality	Description	Type
Start time	Required	This is the time, at which the first time slot starts.	Date and Time
Slot duration	Required	Specifies the duration of a time slot.	Duration
Consumption	Required [1..n]	Each consumption entry consists of a minimal and maximal decimal value.	Decimals
Unit	Optional	Specifies the unit of the consumption values, e.g. kWh. The knowledge of this unit is mandatory, but it can be predefined in a contract.	String

Table 4.3: Information needed in an energy balance forecast

ance forecast is sent as an acknowledged unicast, the *eiCreateTender* operation has to be used. OpenADR does not provide any mechanisms for sending energy balance forecasts in the way it is required for the specified use cases. The only way of communicating data about planned or also current and historic consumption is the usage of reports. But this option would require a different definition of the use cases, where information gets requested by the agents on the grid side, as mentioned before. CIM allows the definition of energy balance forecast messages in different ways. In principle, the options are the same as for the day-ahead pricing signal. On the one hand, it is possible to use a *MarketDocument* which includes a *TimeSeries* populated with quantity data. On the other hand, the *RegularIntervalSchedule* class can be used to present the consumption values.

4.2.4 Grid priority signal

Grid priority signals convey instructions to reduce or increase the energy consumption in case of critical situations in parts of the grid. Such a command needs a point in time when the load adaptation should take place and a numeric value that specifies the relative extent of load change. The content of the grid priority signal is shown in Table 4.4.

Name	Ordinality	Description	Type
Start time	Required	This is the time, at which the customer has to adapt the load.	Date and Time
Load change	Required	This is the requested relative load change in percent. The baseline is the planned energy consumption, e.g. in kWh.	Decimal

Table 4.4: Information needed in the grid priority signal

Grid priority situations are typical applications for the *EiEvent* service. As this service is also part of the OpenADR 2.0 profiles, EI and OpenADR are capable of modeling this kind of signals. The intended EI operation to send an event with an awaiting acknowledge is called *eiCreateEvent*. The OpenADR profile specification differs slightly from the mechanisms defined in EI. There is a single operation called *oadrDistributeEvent*, which is used for unicasts as well as broadcasts and multicasts. An additional attribute defines whether an acknowledge is required or not. Although the data contained in the grid priority signal is relatively simple, the more advanced profile B of OpenADR is required for the exchange of these events. An option based on the CIM is to take a *MarketDocument* which contains a *TimeSeries* with just one time point that includes the desired load change.

4.2.5 Grid priority cancel signal

Some time after a grid priority signal has been sent to a CEMS, a grid priority cancel signal has to be transmitted to indicate the end of the critical situation. This allows the CEMS to go back to normal operation mode. In essence, the content of the message consists of an identifier and the reference to the original signal that should be canceled. Table 4.5 summarizes the information, which should be included in the signal.

Name	Ordinality	Description	Type
Grid Priority	Required	This is the reference to the identifier of the grid priority signal, which should be canceled.	String

Table 4.5: Information needed in the grid priority cancel signal

Like the grid priority signal itself, the corresponding cancel signal can be modeled by the *EiEvent* service, as well. Therefore, EI and OpenADR are appropriate standards for the definition of the message. Within EI, the *eiCancelEvent* operation provides the possibility to terminate an event. Another option is to use the *eiChangeEvent* operation in order to alter the load change of the existing event to zero. As the baseline for the change of load is defined as the planned energy consumption, this approach also allows the CEMS to go back to normal operation mode. OpenADR supports neither of the two operations. It uses the *oadrDistributeEvent* operation to send a new event with the same identification number as the original event and an increased modification number. This allows the alteration of an existing event and therefore can be used in the same way as the method with *eiChangeEvent* described before. Similar to the grid priority signal, the cancel signal can be modeled with a CIM *MarketDocument*. This document only needs a reference to the corresponding signal which should be canceled. The information indicating the end of a critical grid situation can be included in the type of the document.

4.2.6 Flexibility request signal

The flexibility request is intended to ask a CEMS for flexibilities. As described in Chapter 3, flexibilities are defined as sheds or shifts of load. This means that the CEMS needs to know for which time slots it should offer load curtailments while an extent of desired load change is not part of the signal. The customer decides how much energy can be saved for the requested time. Therefore, the most important part of the flexibility request is a list of time slots. Each slot can be described by a start time and a duration. In addition, an expiration time is required. This is the latest time when a possibly following flexibility offer has to arrive at the sender of the flexibility request. A summary of information that is required by the receiver of the signal is shown in Table 4.6.

Name	**Ordinality**	**Description**	**Type**
Expiration	Required	This is the time at which the request expires. After expiration, no more flexibility offers are accepted. This means that customers have to send their offers before this point in time.	Date and Time
Intervals	Required [1..n]	The intervals define the actual time slots for which the flexibility operator or DEMS wants to buy flexibilities. As the customers are asked for an arbitrary amount of load curtailments, it is sufficient to include a start time in each interval.	Decimal
Duration	Required	Specifies the duration of the time slots (intervals). A common option to describe duration is provided by RFC 5545 [7].	String

Table 4.6: Information needed in the flexibility request

The *EiTender* service can be utilized to define the data structure of the flexibility request. EI provides the possibility to mark a tender as a so-called indication of interest. In particular, the flexibility request is an indication of interest to buy flexibilities. The signals have to be acknowledged, and they are sent to the CEMSs as unicasts. This procedure is supported by the *eiCreateTender* operation. As OpenADR does not support any transactive services, it is not suitable for the transportation of this signal type. Basically, the flexibility request is just a time schedule and therefore can also be modeled with a CIM-based signal. For example, a *MarketDocument* containing a *TimeSeries*, which consists of TimePoints with start and end time, can be used. The problem with CIM is that there is no attribute representing the expiration date.

4.2.7 Reject flexibility request signal

If a CEMS cannot offer any flexibilities for the requested time slots, it must reject the preceded request. In principle, the message only has to convey the identifier of the corresponding flexibility request. All information which should be accessible by the receiver is shown in Table 4.7.

As described before, the only possibility to define the flexibility request in EI is the usage of the *EiTender* service. Therefore, the rejection has to cancel a tender. The appropriate operation is called *eiCancelTender*. Everything that has to do with tenders is part of transactive services

Name	Ordinality	Description	Type
Flexibility Request	Required	This is the reference to the flexibility request, which should be rejected.	String

Table 4.7: Information needed in the reject flexibility request signal

and therefore cannot be realized with OpenADR. A CIM-conform signal could consist of a *MarketDocument* which contains a reference to the MarketDocument representing the flexibility request. Such a document also provides a type attribute, which can be used to identify the signal as reject signal.

4.2.8 Flexibility offer signal

The flexibility offer signal is used by the CEMSs to provide one or more offers to adapt their energy consumption in accordance with a precedent flexibility request. The required information for a flexibility offer signal is shown in Table 4.8. The data which is required to define a particular offer in this signal is summarized in Table 4.9. The central element of the offers is a list of intervals which contain the flexibilities. The intervals are defined by a start time and a duration. The included flexibilities consist of quantity and price. Quantity specifies the offered change in consumption in absolute terms, and price is the monetary equivalent which has to be paid by the flexibility operator or DEMS if the offers is accepted. Each offer also needs an expiration time, which states the latest time when the offer can be ordered.

Name	Ordinality	Description	Type
Offers	Required [1..n]	The actual flexibility offers, which are included in the signal.	See Table 4.9

Table 4.8: Information needed in the flexibility offer signal

The suitable service within EI is again the *EiTender* service. However, now it is no tender to buy something, but a tender to sell flexibilities. The flexibility offer signal is sent to a single receiver and has to be acknowledged. Therefore, the *eiCreateTender* operation can be used. OpenADR is not capable of modeling this kind of message, because it does not support transactive services. When using CIM, one or more *MarketDocument* objects could represent the offers. Each document contains a *TimeSeries* object that consists of time points with quantity values and an associated price. A problem is again the absence of an attribute for the expiration date when using a CIM-based signal.

4.2.9 Reject flexibility offer signal

To tell a CEMS that one or more of the offers will not be accepted, a message including references to the corresponding offers is required. The signal is quite similar to the reject flexibility request signal. When comparing Table 4.10 to Table 4.7, one can see that the only difference is that it includes identifiers for offers instead of the reference to the flexibility request.

Name	Ordinality	Description	Type
ID	Required	This is the unique identifier of the offer.	String
Expiration	Required	The expiration defines the latest time, at which the offer can be ordered by the flexibility operator or the DEMS.	Date and Time
Created	Optional	Represents the time at which the offer has been created.	Date and Time
Intervals	Required [1..n]	Intervals define time slots and contain the offered flexibilities. The timing of the slots is described by a start time and a duration. Apart from timing information, an interval consists of price and quantity. Quantity describes the absolute change of energy consumption for the corresponding time slot. Price defines the amount of money which has to be paid by the flexibility operator or DEMS when the offer is ordered.	
Duration	Required	Specifies the duration of the time slots (intervals). A common option to describe duration is provided by RFC 5545 [7].	String
Currency	Optional	Defines the currency of the price to which the flexibility is offered. A way to represent currency is the usage of ISO 4217 currency codes. The knowledge of the currency is mandatory, but when it is predefined in a contract it is not necessary to include it in each flexibility offer.	String
Unit	Optional	Specifies the unit of the offered energy savings, e.g. kWh. Like the currency, knowledge of a unit is mandatory. But this unit can be predefined in a contract, as well.	String

Table 4.9: Information needed in a particular flexibility offer

By the use of EI, the offers which shall be rejected are defined by tenders. This action is supported by the *eiCancelTender* operation defined within the *EiTender* services. This service is not available in OpenADR. By the utilization of CIM, the signal can be modeled in the same way as the reject flexibility request message.

4.2.10 Flexibility order signal

The information that must be included in the flexibility order is nearly the same as in the rejection described before. To order an offer, the receiving CEMS needs to know the identifier of the

Name	Ordinality	Description	Type
Offers	Required [1..n]	These are the references to the flexibility offers, which shall be rejected. As a flexibility offer signal can contain more than one offer, the reference has to point to the identifier of the particular offer and not to the identifier of the entire signal.	String

Table 4.10: Information needed in the reject flexibility offer signal

desired offer. The only difference concerning the content of the message is that the order contains information about the necessity of a subsequent release signal. Using this information, the CEMS knows whether it should apply the schedule of the ordered flexibility immediately or after an explicit release. A summary of the data which is needed by the CEMS to relate the order to an offer is shown in Table 4.11.

Name	Ordinality	Description	Type
Offer	Required	This is the reference to the flexibility offer, which should be ordered. As a flexibility offer signal can contain more than one offer, the reference has to point to the identifier of the particular offer and not to the identifier of the entire signal.	String
Release required	Required	The release type describes if the order needs a subsequent release signal.	Boolean

Table 4.11: Information needed in the flexibility order

EI supports the acceptation of a tender in form of transactions which are also part of transactive services. The concrete service is called *EiTransaction*. Sending the order in form of a transaction can be done with the *eiCreateTransaction* operation. Orders with required release can be distinguished by the so-called transactive state. The state *transaction* can be used for orders with required release and the state *execution* for the ones without required release. OpenADR does not support the acceptance of offers. When using CIM the signal could look like one of the previously presented reject signals. It consists of a *MarketDocument* with a reference to the desired offer. The indication whether a subsequent release is required has to be included in the document type.

4.2.11 Flexibility order release signal

If an order needs a subsequent release, the flexibility order release signal is required. In essence, the information which has to be included in the signal is the same as in the flexibililty order. The only difference is, that the indication whether a release signal is required is not needed.

It is also possible to set the *Release required* attribute to *false*. This method uses a reference to the concerned offer. Another option is to address the order which should be released. For functionality it does not matter whether the offer or the order is used as reference.

The first method, where the signal refers to an offer, is supported by EI. As the message includes the same information as the flexibility order, it can also be sent via the *eiCreateTransaction* operation. But the transactive state is permanently set to *execution*. The second method, where the signal refers to the corresponding order, cannot be implemented with EI, because there is no operation that can use a transaction as reference. It is also not allowed to alter an existing transaction. Therefore, it is not possible to change the transactive state of the existing order from *transaction* to *execution*. OpenADR does not provide any mechanisms to release an order. A CIM-conform signal can be modeled by the same structure as the flexibility order, just with another document type.

4.2.12 Grid control parameters signal

The grid control parameters are used to provide information about how to adapt the load dependent on the current voltage levels in the grid. The central elements of the signal are the two curves that describe the desired extent of load change for particular voltage levels. One curve must specify how the effective power has to be changed, and the other one how the reactive power has to be changed. Both curves are represented by multiple interpolation points, which consist of a voltage level and a relative load change. The baseline for the load adaption is the current load. Additional information required by the receiver of the signal includes the nominal voltage level and the affected section of the grid. Table 4.12 shows a summary of the data which is needed by the CEMSs to react on voltage deviations without gaining explicit instructions from the SLVGC.

Name	Ordinality	Description	Type
Nominal voltage	Optional	The nominal voltage specifies the desired voltage level in the grid.	Decimal
Grid section	Optional	Specifies the section of the grid, where the included information is valid.	String
Effective power	Required	Describes a P(U) curve.	List of points
Reactive power	Required	Describes a Q(U) curve.	List of points

Table 4.12: Information needed in the grid control parameters signal

Neither EI nor OpenADR provide a service or an operation which can be utilized to exchange grid control parameters. CIM supports the definition of curves with x and y values. It allows the specification of units and multipliers for both axes. Also multiple y values in one diagram are possible.

4.2.13 Conclusion

In summary, EI supports the definition of all signals except of the grid control parameters. The currently most powerful OpenADR 2.0 profile is only suitable for a small subset of the required messages. Even though OpenADR supports reporting services for forecasting planned energy consumption and provides the possibility to publish tariff information by the utilization of event services, only the signals which are related to grid priority functions can be presented exactly in the way needed within the defined use cases. As mentioned before, future profiles of OpenADR may include quote services for distributing complex price information. If this development takes place, OpenADR will also be a good option for propagating day-ahead price signals. Because of the previously defined aim to define as many signals as possible by the same data model, EI is the preferred option of these two within this work. Nonetheless, as OpenADR specifies profiles of EI, it is no problem to replace EI messages by OpenADR messages. Except for the prefix of some XML elements they have the same format and meaning.

Basically, CIM-conform messages can be defined for all signals. Only expiration dates which are needed in various signals cannot be modeled. Hence, CIM-based signals must be extended by non standardized elements to include all required information. It has to be noted that CIM contains a very large number of classes, relations, and attributes. Therefore, the statement that some data cannot be presented has to be treated with caution. Another disadvantage of using CIM for defining the required signals is that all currently standardized profiles are intended for business-to-business communication and not for exchanging data with customers. Problems with using CIM without a standardized profile are that it is not defined, which elements, relations, or attributes are mandatory and that there are no specified values for attributes like message or document types. But even profiles like IEC 62325-351 for market information exchange between utilities include only enumerations with exemplary character. This means that concrete types for *MarketDocuments* or currency codes have to be maintained outside the standard. Also future extensions of messages with informative data like carbon footprints or the origin of energy is not that easy as with EI. The suitability of EI, OpenADR, and CIM to model the defined signals is summarized in Table 4.13.

The lack of a CIM profile for communication between parties at the grid side and customers including demand response actions, and the fact that EI is already a stable OASIS standard are the main reasons for selecting EI as the best option for defining the signals. Only the structure of grid control parameters has to be self-defined. This means that this is the only standardization gap for the defined use cases. All other use cases and signals are covered by the OASIS standard. The disadvantage of relatively large EI messages in comparison to CIM messages can be solved by using compressed formats like Efficient XML Interchange (EXI) [28]. Another benefit of EI is the existence of its well-accepted OpenADR profiles. To round up the selection of the data model, Table 4.14 summarizes the advantages and disadvantages of utilizing EI and CIM for the signal definition.

Figures 4.2 to 4.10 visualize the structure of the finally selected signals. The corresponding acknowledgments can be seen in Figures 4.11 to 4.15. The vertical and underlined labels specify the name of complex XML elements which include further XML elements. The horizontal

Signal	EI	OpenADR	CIM
Day-ahead pricing signal	+	~(+)	+
Energy balance forecast signal	+	-	+
Grid priority signal	+	+	+
Grid priority cancel signal	+	+	+
Flexibility request signal	+	-	~
Reject flexibility request signal	+	-	+
Flexibility offer signal	+	-	~
Reject flexibility offer signal	+	-	+
Flexibility order signal	+	-	+
Flexibility order release signal	+	-	+
Grid control parameters signal	-	-	+

Table 4.13: Suitability of data models

EI	CIM
+ OASIS standard	- Enumerations (values) not standardized
+ Supports nearly all signals	+/- Supports all signals (partially)
- Relatively large messages	+ Small messages
+ Mighty protocol (easy extension of signals)	- Some extensions not possible
+ OpenADR parts well accepted	
+ CIM conformance in small parts	
- Grid control parameters not supported	

Table 4.14: Advantages and disadvantages of EI and CIM

labels, on the other hand, represent simple XML elements including some value like a string. Except for grid control parameters, all signals are fully covered by the EI standard. The format of the grid control parameters message is self-defined.

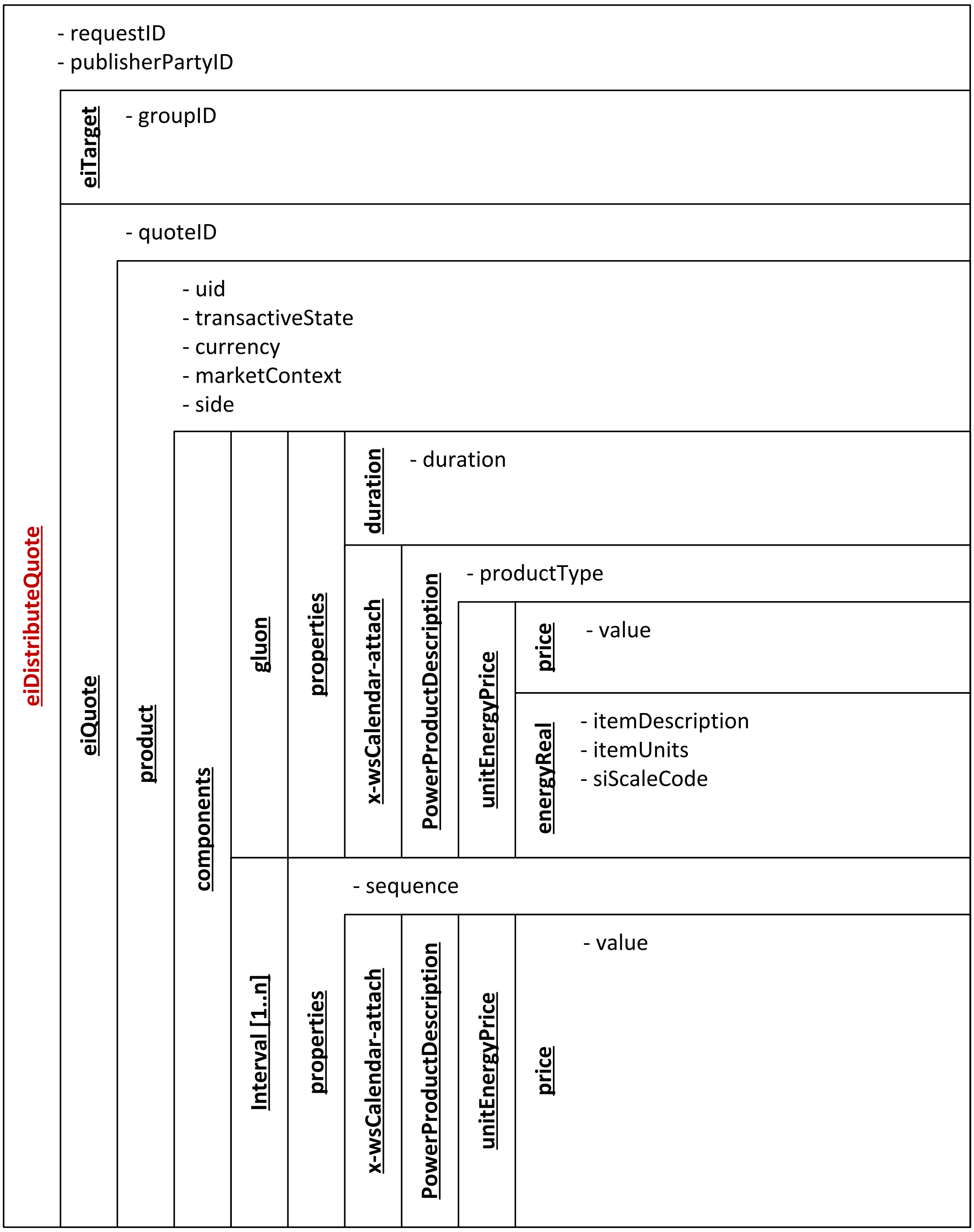

Figure 4.2: Day-ahead pricing signal structure

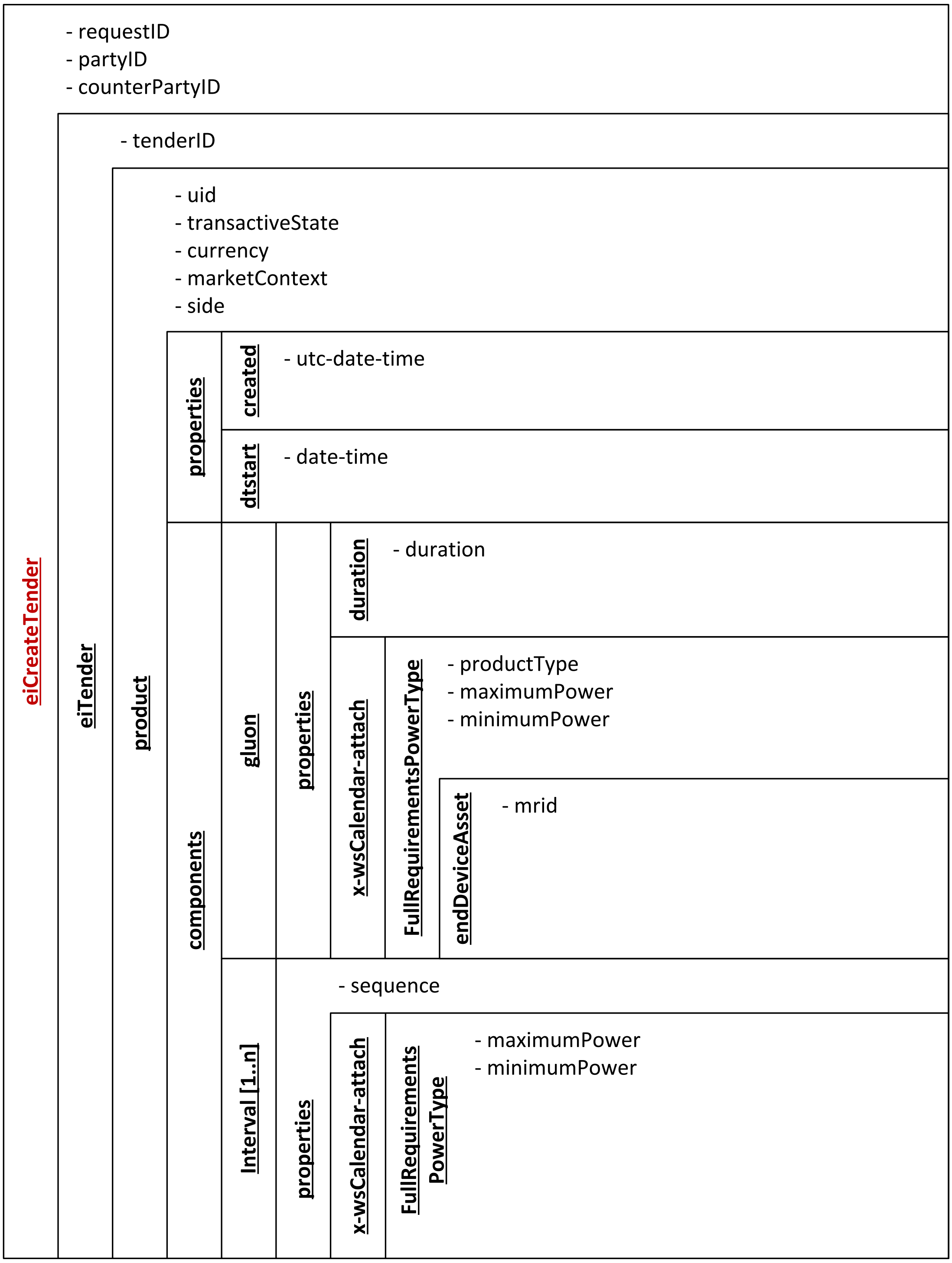

Figure 4.3: Energy balance forecast signal structure

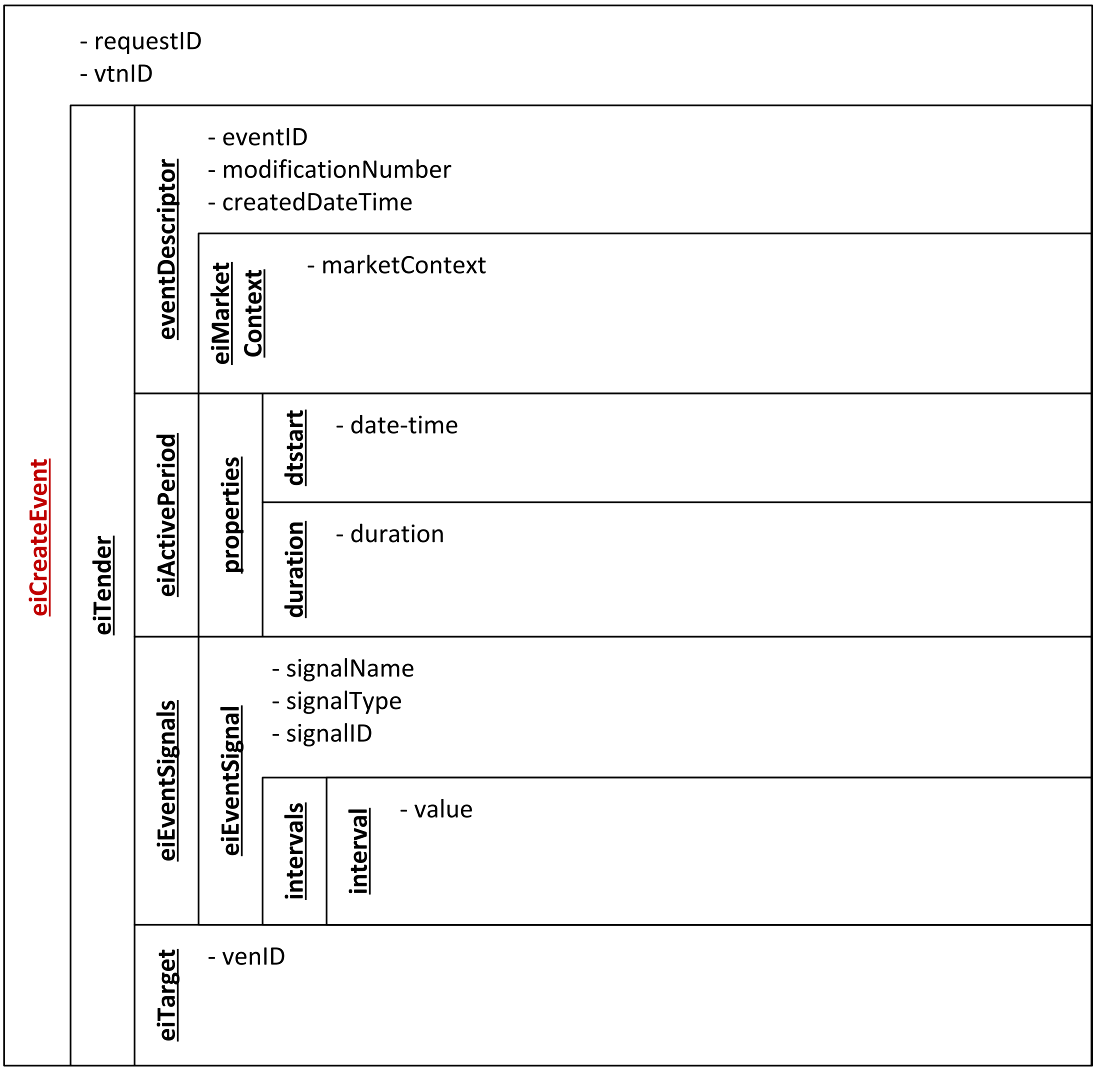

Figure 4.4: Grid priority signal structure

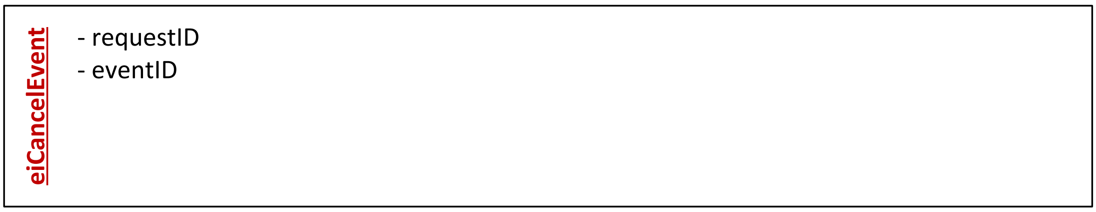

Figure 4.5: Grid priority cancel signal structure

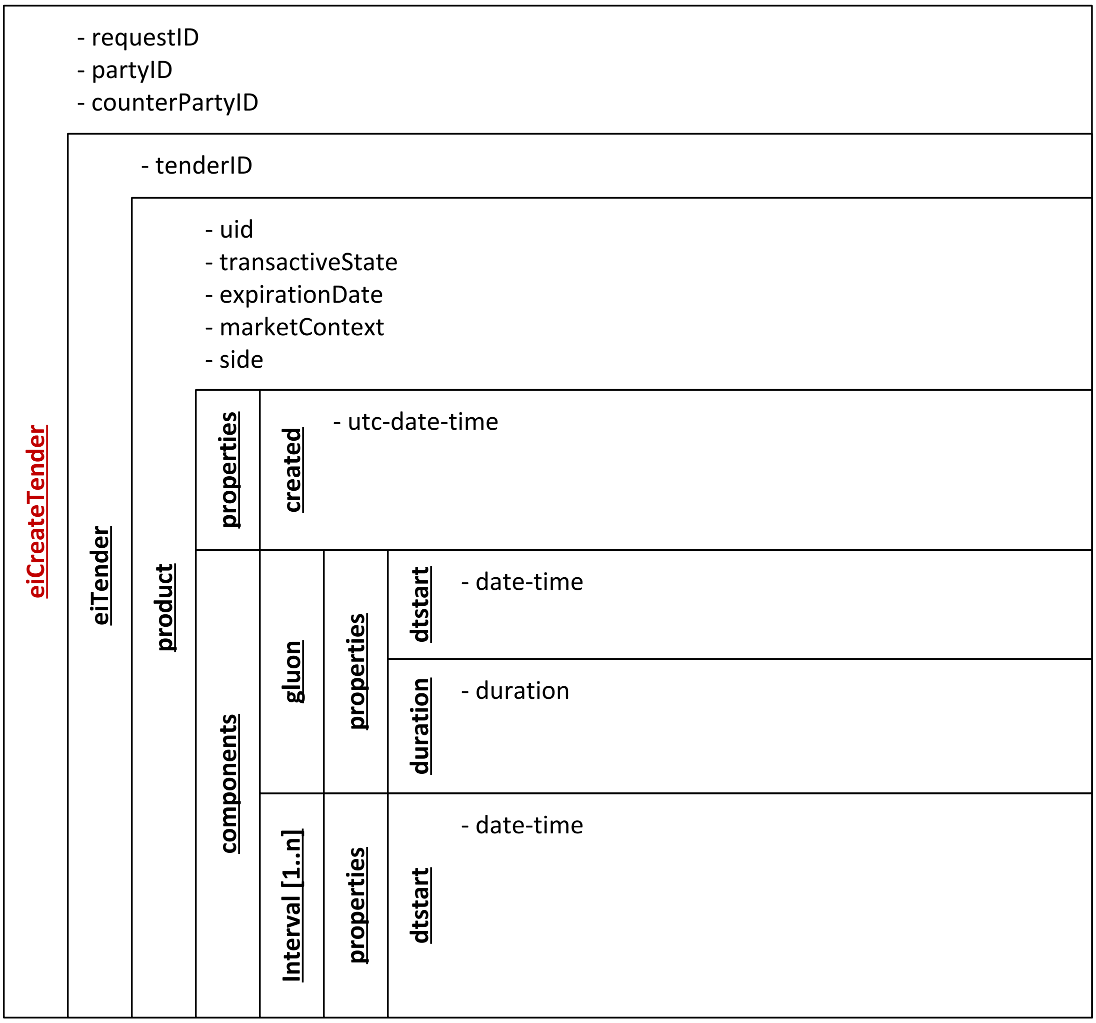

Figure 4.6: Flexibility request signal structure

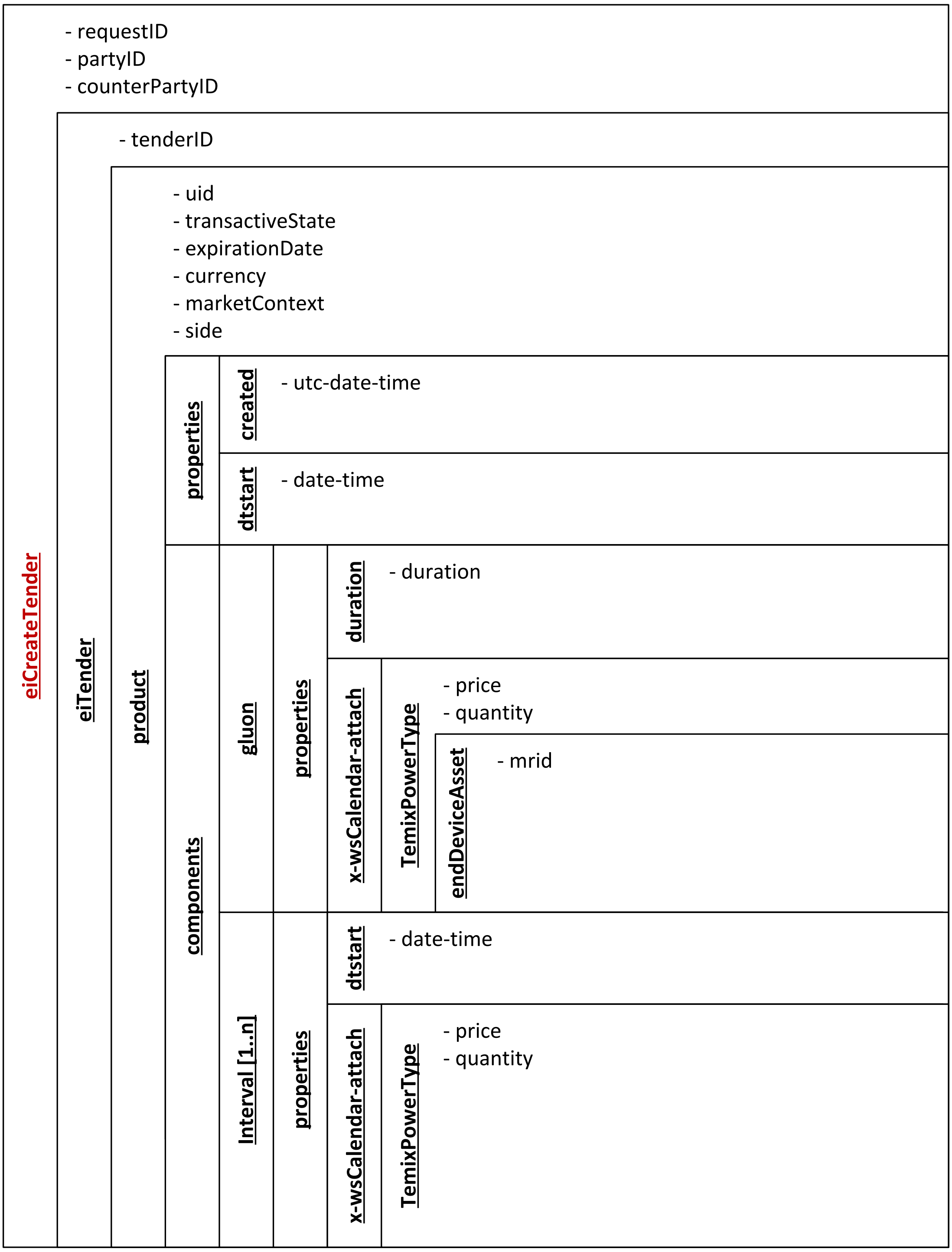

Figure 4.7: Flexibility offer signal structure

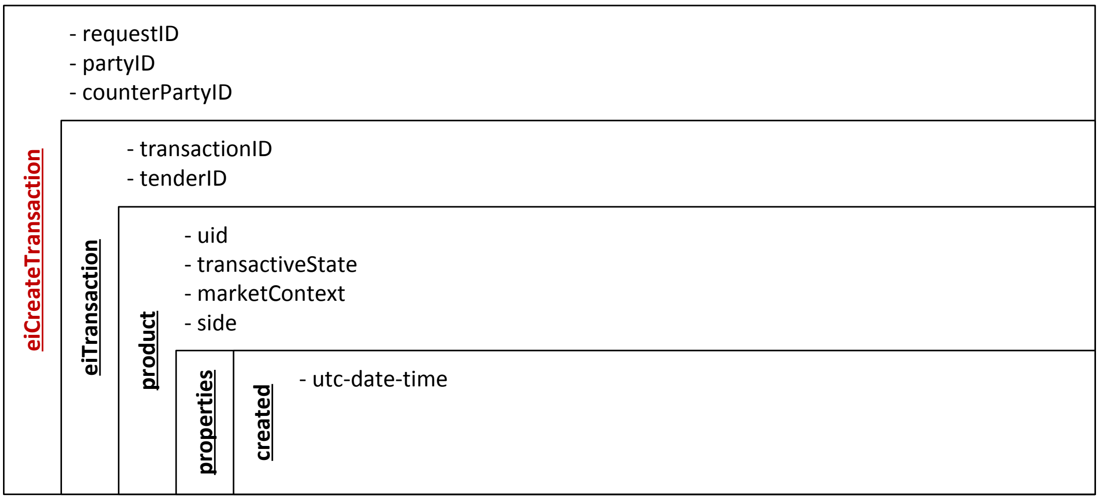

Figure 4.8: Flexibility order or release signal structure

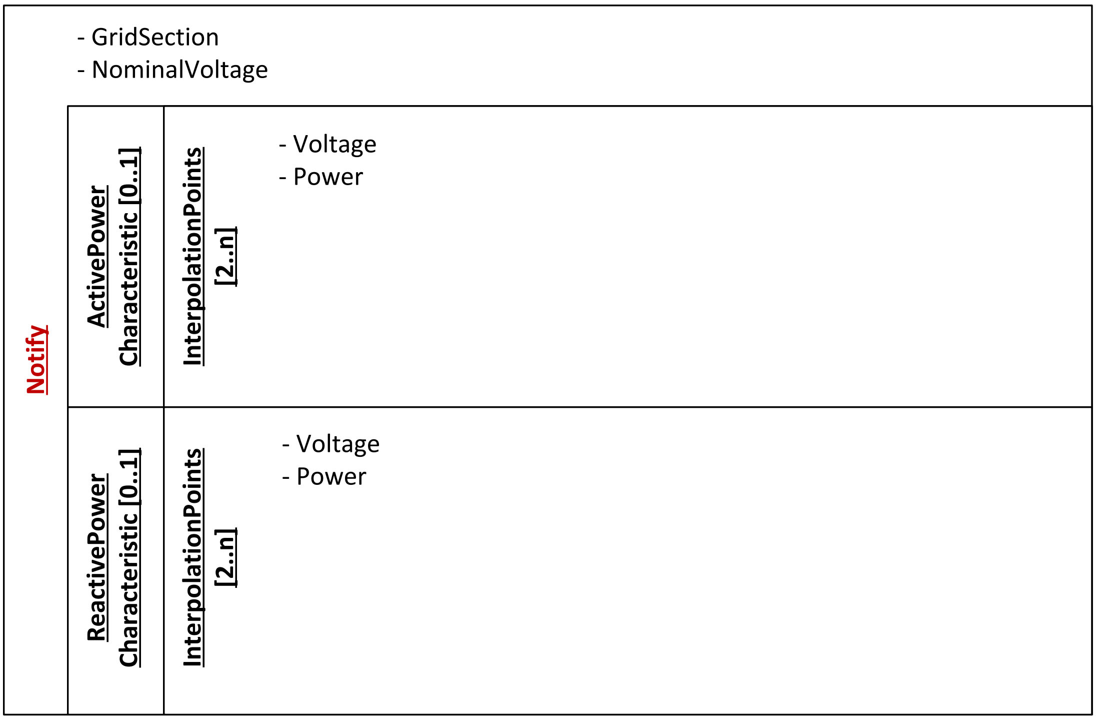
Notify
- GridSection
- NominalVoltage
ActivePower Characteristic [0..1]
InterpolationPoints [2..n]
- Voltage
- Power
ReactivePower Characteristic [0..1]
InterpolationPoints [2..n]
- Voltage
- Power

Figure 4.9: Grid control parameters signal structure

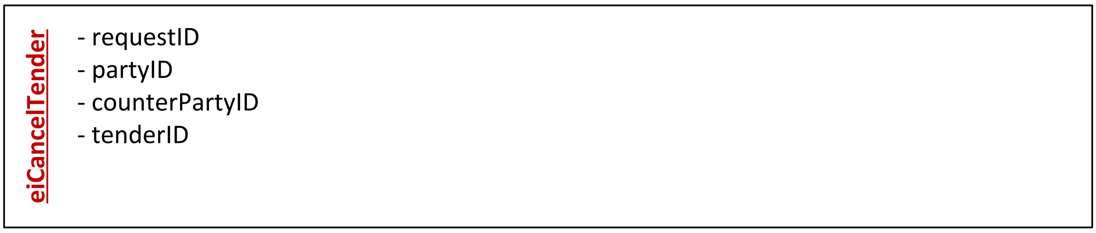

Figure 4.10: Cancel tender signal structure

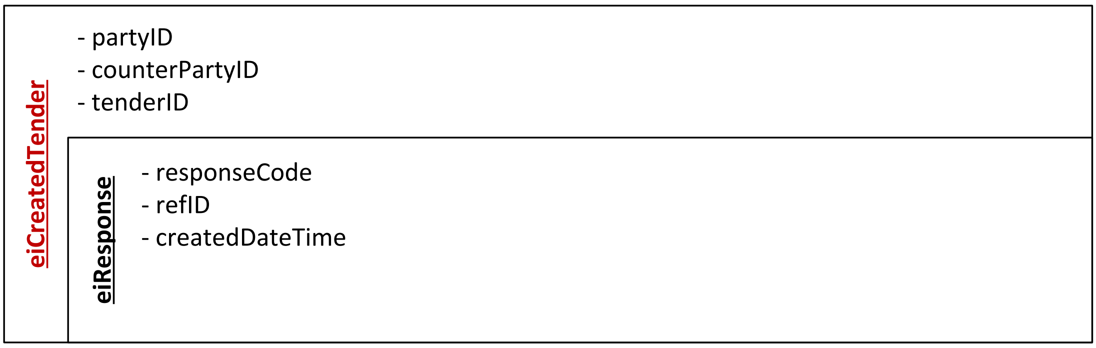

Figure 4.11: CreatedTender signal structure (acknowledge)

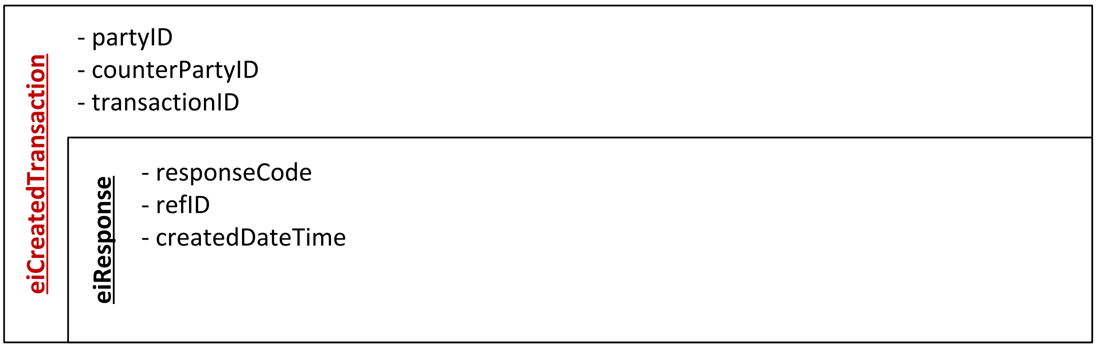

Figure 4.12: CreatedTransaction signal structure (acknowledge)

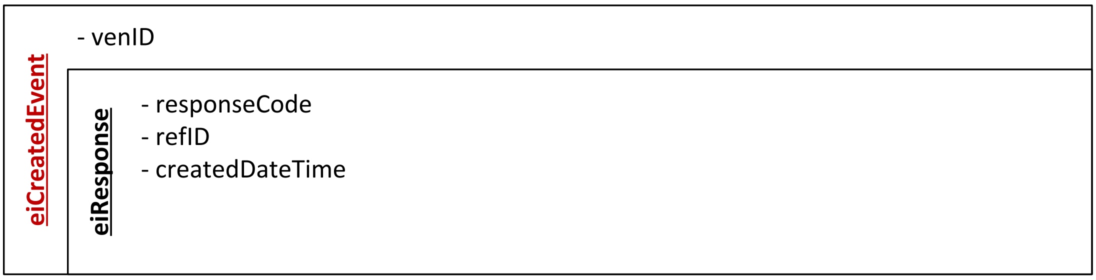

Figure 4.13: CreatedEvent signal structure (acknowledge)

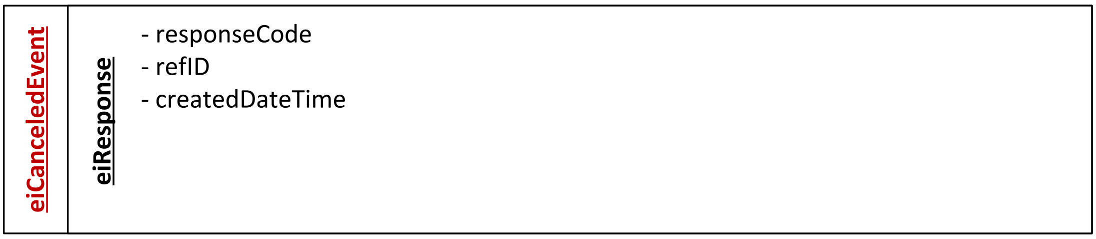

Figure 4.14: CanceledEvent signal structure (acknowledge)

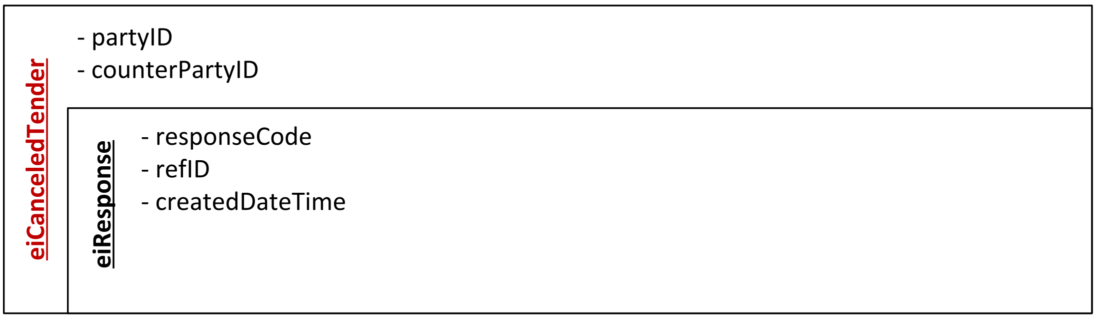

Figure 4.15: Canceled tender signal structure (acknowledge)

CHAPTER 5

Proof of concept implementation

All the previously introduced use cases have been implemented as part of a demonstration environment. The installation consists of several Raspberry Pis of the model B+, which represent either an agent participating in the smart grid or an XMPP server. An additional Raspberry Pi is used for the domain name service (DNS), a dynamic host configuration protocol (DHCP) server, and a network time protocol (NTP) server. In the following sections, the overall setup and its subsystems are described in detail. In this context, agents and XMPP servers are also referred to as systems.

5.1 Demonstration setup

Figure 5.1 shows a setup that is used to visualize a small demonstrative smart grid. Each agent or XMPP server is in the scope of an operational domain represented by circles in the figure. The domain specifies in whose area of responsibility the particular system is operated. All Raspberry Pis belonging to the same operator, are grouped together in a common hardware case. This cases are placed in the rectangles within the circles representing the domains. Although CEMSs are operated in the sphere of a customer, or in some cases of a prosumer, they are assigned to the domain of a grid operator. This is because there is a fixed relationship between a CEMS and the grid operator. The grid operator can not be changed by the customer and it provides the communication infrastructure for the CEMSs. It is not unlikely that the hardware and software, which is needed for the CEMSs must be made available by the grid operator. Figure 5.1 also contains two different kinds of communication links. The solid, bold lines represent the active links, which are part of this work. These links are covered by the interface between the smart building and the smart grid. However, not all links have to be actually used. For example, when none of the CEMSs in the domain of grid operator 2 is associated with the DEMS in the domain of the energy pool manager, the link between grid operator 2 and energy pool manager will not be in use. The dashed lines represent links, which can be used for communication between two agents on the grid side. They are supported by the XMPP infrastructure and might be used in

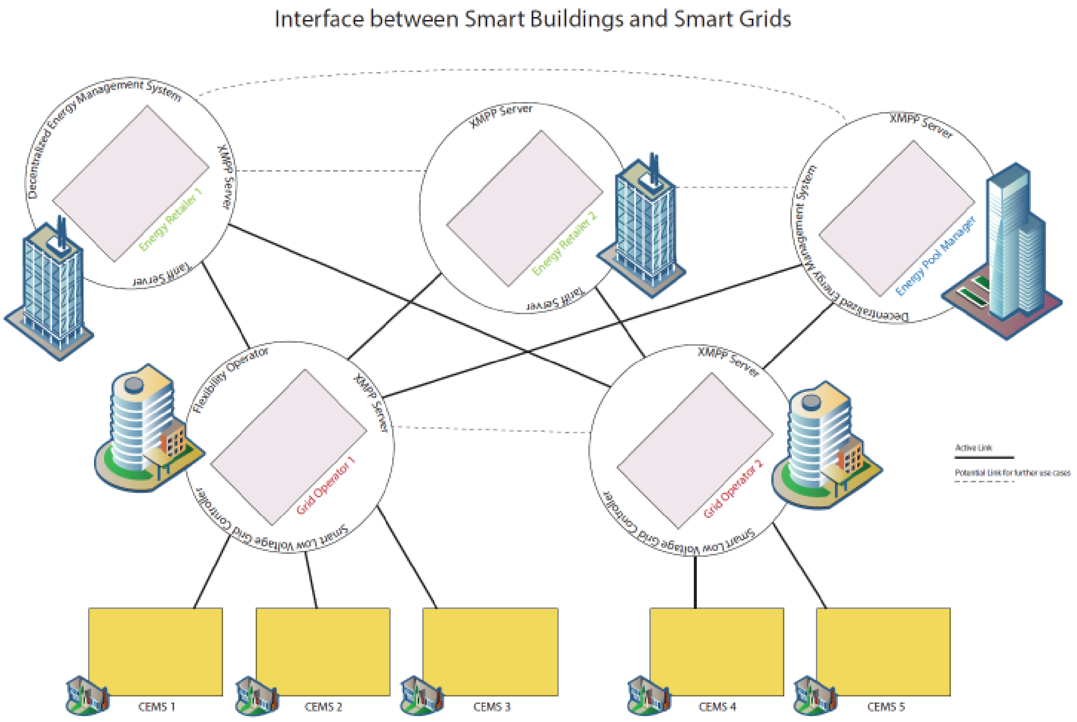
Interface between Smart Buildings and Smart Grids
Decentralized Energy Management System
XMPP Server
Tariff Server
Energy Retailer 1
XMPP Server
Energy Retailer 2
Tariff Server
XMPP Server
Decentralized Energy Management System
Energy Pool Manager
Flexibility Operator
XMPP Server
Grid Operator 1
Smart Low Voltage Grid Controller
XMPP Server
Grid Operator 2
Smart Low Voltage Grid Controller
Active Link
Potential Link for further use cases
CEMS 1
CEMS 2
CEMS 3
CEMS 4
CEMS 5

Figure 5.1: Demonstration environment

future smart grids. Nonetheless, they are not part of the actual SGUI and therefore not used in this work.

All Raspberry Pis are connected via a Wi-Fi router, which provides Internet access to request the current time from an external NTP server. If no Internet connection is available, time synchronization is achieved via an internal NTP server. Actually, there is always a time window of some minutes or even hours to react on incoming signals at each agent. Therefore, time is not a very critical issue. But the system time is lost every time Raspberry Pis are disconnected from power as they do not include a battery and hardware clock. Moreover, time has to be approximately equal on all clients. These are the reasons why an NTP server is indispensable. An internal DNS server is especially needed for SRV resource records, which are used by the XMPP servers to request available network services. More information about NTP and DNS configuration can be found later in this section. As operating system the Debian based Raspbian has been chosen due to the large number of software packages available for Debian distributions. Additionally drivers for many Wi-Fi adapters are included out of the box. In order to provide the ability of displaying status information directly on each Raspberry Pi, the hardware extension BerryClip is used. This extension board consists of six LEDs, a switch, and a buzzer. Connection of the board to the Raspberry Pi is realized via the GPIO ports. The Java library Pi4J project enables the control of these GPIO ports.

In summary, the demonstration smart grid consists of overall five domains. While Figure 5.1 shows them just with their shorthand name, the fully qualified names of the domains are listed in the following together with their associated agents.

gridoperator1.xmpp.local : SLVGC 1, Flexibility Operator, CEMS 1, CEMS 2, CEMS 3

gridoperator2.xmpp.local : SLVGC 2, CEMS 4, CEMS 5

energyretailer1.xmpp.local : Tariff Server 1, DEMS 1

energyretailer2.xmpp.local : Tariff Server 2

energypoolmanager.xmpp.local : DEMS 2

Each domain name is coincident with the name of an XMPP server. Therefore, the agents are identified by JIDs in the form of *hostname@operator*.xmpp.local. All agents as well as important system and network settings are listed in Table 5.1.

Agent	**Host name**	**JID**	**IP address**
CEMS 1	cems1	cems1@gridoperator1.xmpp.local	10.0.0.11
CEMS 2	cems2	cems2@gridoperator1.xmpp.local	10.0.0.12
CEMS 3	cems3	cems3@gridoperator1.xmpp.local	10.0.0.13
CEMS 4	cems4	cems4@gridoperator2.xmpp.local	10.0.0.14
CEMS 5	cems5	cems5@gridoperator2.xmpp.local	10.0.0.15
Tariff Server 1	tariffserver1	tariffserver1@energyretailer1.xmpp.local	10.0.0.21
Tariff Server 2	tariffserver2	tariffserver2@energyretailer2.xmpp.local	10.0.0.22
DEMS 1	dems1	dems1@energyretailer1.xmpp.local	10.0.0.31
DEMS 2	dems2	dems2@energypoolmanager.xmpp.local	10.0.0.32
SLVGC 1	slvgc1	slvgc1@gridoperator1.xmpp.local	10.0.0.41
SLVGC 2	slvgc2	slvgc2@gridoperator2.xmpp.local	10.0.0.42
Flexibility Operator	flexop	flexop@gridoperator1.xmpp.local	10.0.0.51

Table 5.1: System configuration for all agents

According to the architecture of XMPP, there is no direct communication between two agents. All agents are connected to an XMPP server within their particular local domain, and they are identified with their unique JID. If the receiver of a message is located in the same domain as the sender, the server forwards the message directly to the sink. Otherwise, the message is forwarded to a remote server. Finding the right remote server is done by using the domain part of the JID and requesting the corresponding TCP port and IP address from the DNS server.

5.1.1 XMPP servers

The XMPP infrastructure is realized by five Openfire servers in version 3.9.3. Openfire is a real time collaboration server published under the Open Source Apache License and therefore free to use. It can be easily configured with a Web interface and supports many extensions. Moreover the creation of multi user chatrooms (MUCs) for multicast messages is possible. As Openfire is written in Java, it requires the Java Runtime Environment. The five server instances and some important parameters are shown in Table 5.2. Web interfaces for configuration and monitoring are available via http://*name*:9090 or with enabled transport layer security via https://*name*:9091. When there is no DNS server running, the IP address has to be used instead of the name.

Name	**IP address**	**Username**	**Password**
energyretailer1.xmpp.local	10.0.0.2	admin	S3rv3r_XMPP
energyretailer2.xmpp.local	10.0.0.3	admin	S3rv3r_XMPP
energypoolmanager.xmpp.local	10.0.0.4	admin	S3rv3r_XMPP
gridoperator1.xmpp.local	10.0.0.5	admin	S3rv3r_XMPP
gridoperator2.xmpp.local	10.0.0.6	admin	S3rv3r_XMPP

Table 5.2: XMPP server configuration parameters

On each server, users for the local agents of the domain have to be created. For the demonstration setup, it is sufficient to configure a username and the corresponding password. In a real world scenario, it may be useful to classify users into groups. Also additional attributes like the name of the owner or an email address can be attached to the users. There are two important services which have to be enabled and configured on each server. First, remote servers have to be allowed to establish a connection to the server. For security issues, it is feasible to define a white list of servers, which are allowed to connect. Second, the group chat service has to be configured. In the demonstration environment, the Openfire standard service *conference* is used. The service is addressed by its full name conference.*hostname*.xmpp.local (e.g. conference.energyretailer2.xmpp.local). In order to prevent unauthorized agents from creating a chat room, a list of users (or groups) defines who is allowed to create a chat. Apart from security issues, this list excludes CEMSs from creating chat rooms when they try to join a room that does not yet exist. Table 5.3 shows who is allowed to create MUCs on the servers.

Server	**Allowed to create MUC**
energyretailer1	Tariff Server 1
energyretailer2	Tariff Server 2

Table 5.3: MUC creation permissions

5.1.2 Supporting services

The supporting services include DNS, NTP, and DHCP. As mentioned before, all of them are hosted on a common Raspberry Pi. DNS and DHCP servers are realized with Dnsmasq, a lightweight software package included in most Linux distributions. Despite of DNS and DHCP, Dnsmasq is also intended to provide the following services for small networks: router advertisement, and network boot. The demonstration implementation makes use of the DNS and DHCP modules only. The main part of the configuration can be found in the file */etc/dnsmasq.conf*.

The DHCP service needs a range of assignable IP addresses, the standard gateway (IP address of the router) and for this particular project a list of reserved IP addresses. This list assigns an IP address to a MAC address and ensures that each Raspberry Pi is always reachable by the same IP address. Another possibility is to assign static addresses to all Raspberry Pis. DNS for static IPs uses the hosts file on the server. As the DHCP service assigns fixed IP addresses to each client, static IPs can be used for DNS records. Additionally, the domain name *xmpp.local* is specified in the main configuration file. Also the following SRV resource records for XMPP server services and XMPP conference services are defined (see Listing 5.1).

```
srv-host=_xmpp-server._tcp.energyretailer2.xmpp.local,
   energyretailer2.xmpp.local,5269
srv-host=_xmpp-server._tcp.gridoperator1.xmpp.local,
   gridoperator1.xmpp.local,5269
srv-host=_xmpp-server._tcp.gridoperator2.xmpp.local,
   gridoperator2.xmpp.local,5269
srv-host=_xmpp-server._tcp.energypoolmanager.xmpp.local,
   energypoolmanager.xmpp.local,5269
srv-host=_xmpp-server._tcp.energyretailer1.xmpp.local,
   energyretailer1.xmpp.local,5269
srv-host=_jabber._tcp.energyretailer2.xmpp.local,
   energyretailer2.xmpp.local,5269
srv-host=_jabber._tcp.gridoperator1.xmpp.local,
   gridoperator1.xmpp.local,5269
srv-host=_jabber._tcp.gridoperator2.xmpp.local,
   gridoperator2.xmpp.local,5269
srv-host=_jabber._tcp.energypoolmanager.xmpp.local,
   energypoolmanager.xmpp.local,5269
srv-host=_jabber._tcp.energyretailer1.xmpp.local,
   energyretailer1.xmpp.local,5269
srv-host=_jabber._tcp.conference.energyretailer1.xmpp.local,
   energyretailer1.xmpp.local,5269
srv-host=_jabber._tcp.conference.energyretailer2.xmpp.local,
   energyretailer2.xmpp.local,5269
srv-host=_xmpp-server._tcp.conference.energyretailer1.xmpp.local
   ,energyretailer1.xmpp.local,5269
srv-host=_xmpp-server._tcp.conference.energyretailer2.xmpp.local
   ,energyretailer2.xmpp.local,5269
```

Listing 5.1: SRV Resource Records

An NTP server is already pre-installed on Raspbian. In principle, it is not necessary to do any further configuration, but things like other time servers or access control can be configured in */etc/ntp.conf*. If there is no Internet connection and therefore the local time of the server shall be used in the network, the local clock has to be defined as NTP server. The local clock is reachable via 127.127.1.1. On the clients it is necessary to set the correct NTP server (10.0.0.100) and to ensure that time synchronization is also allowed when the clock drift is large.

5.2 Demonstration use cases

Figure 5.2 shows an overview on the realized demonstration use cases along with the primary use cases and their interactions and dependencies. As security is not part of this work, the security use case which was part of the overview diagram (see Figure 3.2) has been removed. Also the use case *attendance or weather changes* is not included in the implementation because it does not include any additional signals.

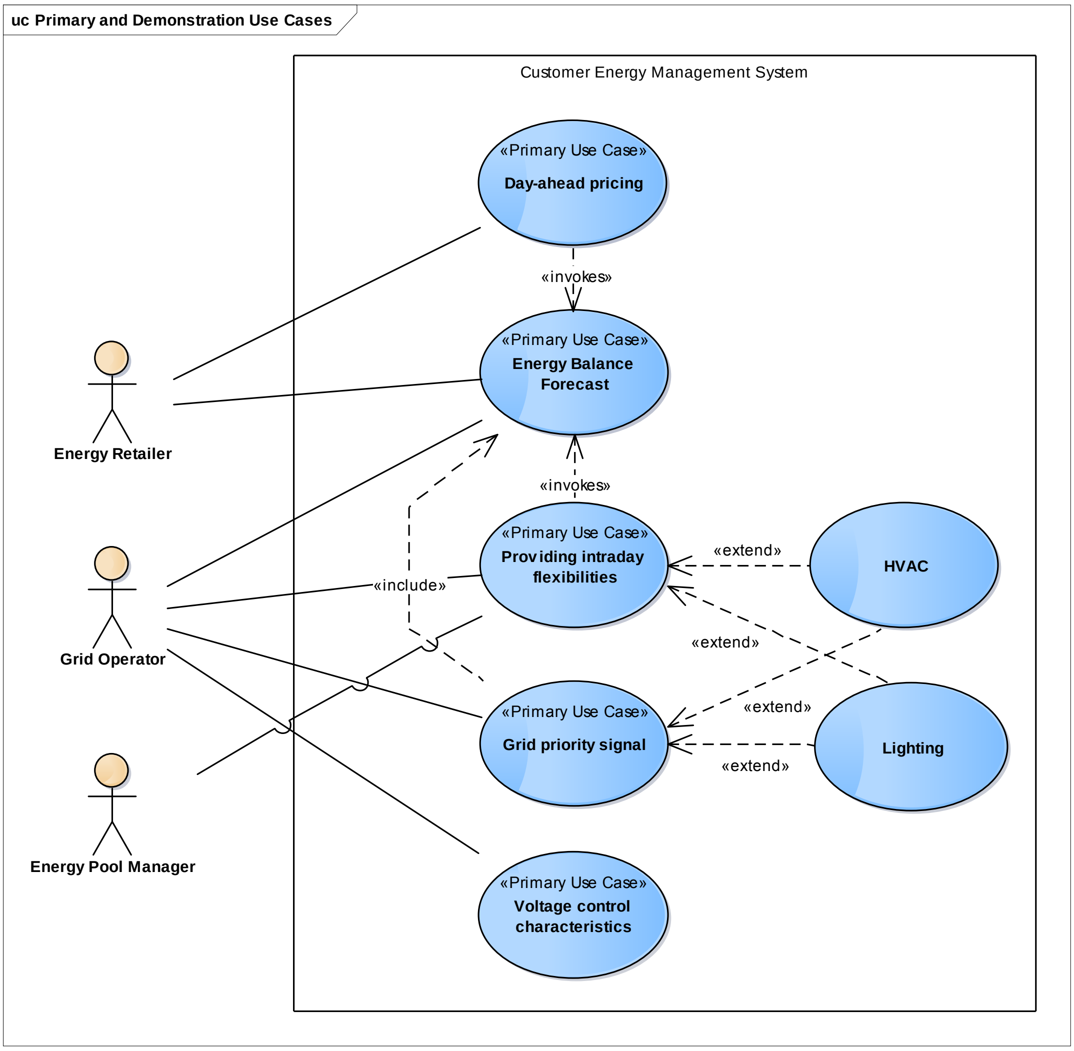

Figure 5.2: Demonstration use cases

5.2.1 Heating, Ventilation and Air Conditioning (HVAC)

The HVAC system is responsible for a large amount of the energy costs. Generally, people prefer a particular temperature in their buildings. The amount of used energy to reach and hold this temperature depends on many factors, like the number of people in the building or the weather, but also simple things like an open window. In order to create reasonably accurate energy balance forecasts, the CEMS needs information about all these influencing factors. Although the acquisition of this information is not part of this work, it is important to observe that all of this information might change throughout the day. Additionally to the desired target temperature, it is possible to specify a range for temperature variations. This can be done by the definition of operating modes, e.g. "*economy*" and "*comfort*". It is possible to use this margin to provide flexibility offers to external agents like the flexibility operator. The mechanism of flexibility negotiation was described in the primary use case *providing intra-day flexibilities* in detail (see Section 3.2.3). When the CEMS receives the daily price signal from the tariff server (see primary use case *day-ahead pricing* in Section 3.2.2), the HVAC schedule is planned. The same procedure takes place when a flexibility order is released by the flexibility operator, with the difference that the planning of the schedule is not just based on the price, but also on the sold flexibilities. If a release of the flexibility order is not required, the schedule is planned directly after the flexibility order is received. The HVAC system can also be used to lower or raise the energy consumption in the case of a grid priority situation.

5.2.2 Lighting

The second facility to demonstrate the primary use cases is a lighting system. Unlike the HVAC system, changing the energy usage of the lighting has an immediate impact on the comfort inside a building. Again, different lighting scenes can be defined, e.g. "*economy*" and "*comfort*". In particular, when a power failure is imminent, it is still better to lower the light intensity than having a complete blackout. Thus in real world this will be a use case for grid priority signals only, but in the demonstration implementation it is also used for *day-ahead pricing* and *intra-day flexibilities*.

5.3 Common Architecture and Configuration

Each CEMS, tariff server, flexibility operator, SLVGC, or DEMS runs on its own Raspberry Pi. The programs are written in Java, which allows an easy transfer of the code to other platforms. For execution, the latest virtual machine JRE 1.8 is used. Since the internal behavior of the agents is not part of the interface between smart buildings and smart grids, the implementation is focused on functions, which are needed to show that the communication protocol works correctly. Nevertheless, it is necessary to calculate valid and reasonably realistic signals. The following sections describe in detail, how this was achieved for each type of agent. Prior to that, implementation details which are common to all programs are presented.

All programs have the same basic structure. Besides the main class, this basic framework consists of a configuration class and the class *EnergyAdmin* which handles the calculation as well as sending and receiving of IQs and messages. The configuration class handles both reading the settings from the file system and providing these settings for usage in all parts of the program. The configuration parameters which are common to all agents are summarized in Table 5.4. All programs have in common that the configuration is read directly after program start up. Afterwards, the connection to the XMPP server is established and potentially needed MUCs are created.

Parameter	Description
mRID	Unique identification
name	Human readable name of the system
xmpp-server	Address of the XMPP server (IP or fully qualified name)
xmpp-port	TCP/IP port of the XMPP server
xmpp-user	XMPP user name as JID without resource
xmpp-password	Password for log in on XMPP server
xmpp-resource	Name of the XMPP resource for login
dayLength	Virtual length of a day, e.g. 24 minutes
referenceDate	Reference date for calculating a virtual date

Table 5.4: Common configuration

The project *SguiXMPPLib* provides a library with classes which are used by two or more of the other projects. This library consists of five packages. One package includes all messages, and another one contains the IQ packets which are sent via XMPP. For both, there is an additional package that contains the parsers to interpret incoming IQ packets and messages. All parsers return an object that can be used to access data from the received signal. Even though sending and receiving IQ packets works differently from handling messages, the principle of transmission is the same. Before sending, the signal has to be converted to XML, and after the receiving, the XML code has to be converted back to a Java object. The conversion to XML is directly included in the classes which represent the signal while the reconversion is done via the parsers. The fifth package is the *common* package that contains all other classes, interfaces, and enumerations which are used by multiple agents. One of the most fundamental classes, the *XMPPAdmin*, which bundles most of the XMPP functionality, can be found in this package. It handles the connection to the XMPP servers including automatic reconnects in case of connection loss and the sending and reception of signals. The class manages the required MUCs to send or receive multicasts. Like the general connection to the server, also these chats are rejoined automatically when the connection was temporarily lost. Another important functionality relating to the MUCs is the possibility to send invitations to join a chat room. For the realization of that XMPP communication interface, the Smack API in version 4.0.3 is used. Smack is a powerful open source XMPP client library for Java. For receiving messages, IQ packets, invitations or more generally XMPP stanzas, Smack provides different kinds of packet listener interfaces. Most of these listeners are implemented directly in the projects of the individual agents, but the registration of the listeners is also done in the *XMPPAdmin* class. Registering the listener enables its execution when a corresponding event, e.g. reception of an IQ, is recognized by the

Smack API. Besides the *XMPPAdmin*, the *common* package contains the following classes:

- CEMAdmin: This is the management class for registered CEMSs at tariff server, flexibility operator, SLVGC, or DEMS. The CEMSs are read from a configuration file.
- ContactData: Stores important data about an agent including mRID, JID, type of the agent and multi user chats to join.
- HelpFunctions: Mainly provides functions to parse or write XML elements.
- MinMaxEnergy: A simple pair of double values which represent minimal and maximal energy consumption which is used in energy balance forecasts.
- PriceQuantity: A simple pair of double values which relates some price to a quantity which is used within pricing signals.
- RegisteredCEM: Represents the contact data for a single CEMS.
- ReliableIQSender: This class is used to send an IQ packet with acknowledge. If the acknowledge does not arrive within a specified time the packet is sent again. An upper bound for retries ensures that the XMPP network does not get flooded with undeliverable packets.
- ReliableMultiSender: It is used to send a message to a MUC. The message has to be read back from the chat by the sender. Otherwise the sending is repeated until reading back was successful or up to a number of maximal retries.
- TenderCombination: Stores a combination of Energy Interoperation tenders.
- VirtualDateConverter: Converts real date and time into a virtual date and time and vice versa.

Additionally, the following enumerations and interfaces are defined in the package:

- Enumerations
  - AgentTypes: CEMS, FlexibilityOperator, SLVGC, TariffServer, DEMS
  - InvitationReason: Reason for invitation to a MUC (tariff)
  - ResponseCodes: Ok, Expired
  - SignalTypes: PricingSignal
  - TransactiveStates: exercise, transaction, tender, indicationOfInterest
- Interfaces
  - IQResultSetter: Is used for the communication between ReliableIQSenders and the class which starts the sender.

- ResultSetter: Is used for the communication between ReliableMultiSender and the class which starts the sender.

For demonstration purposes, it is not practicable to observe real days and signal intervals of 15 minutes or even an hour. Therefore, a solution for scaling a day down to some minutes had to be found. This is achieved by introducing a virtual date and time. The virtual time depends on a reference date, which serves as initial time, and the length of a virtual day in real minutes. For example, if the virtual day lasts for 24 minutes, the virtual date gets incremented by 1 day every 24 minutes. In that case, the virtual time runs 60 times faster than the real time. Since the signals are sent with much smaller time frames in between, this scaling has some influence on the performance of the network. For the demonstration setup with Wi-Fi IEEE802.11n restricted to a very small area with just 5 CEMSs, this is not a problem. However, if one likes to perform some scalability tests, it is feasible to use the real time with a day length of 24 hours.

5.4 Tariff Server

The tariff server is the system, that provides pricing information in the form of day-ahead pricing signals to the customers. To simulate the internal behavior of changing prices more than one price profile has to be defined. Each profile specifies the price per kilowatt hour for a whole day divided in equal time intervals. An example of a very simple profile, which consists of four different prices within a day can be seen in Listing 5.2. Based on such a profile, the actual day-ahead pricing signal is constructed. Every day, at a predefined point in time, the program generates a new signal and sends it to the registered CEMSs. For this purpose, a MUC on the local XMPP server is created. That action takes place at program start up, directly after all configurations have been read and the connection to the XMPP server has been established. If the creation is successful, an invitation to join the chat room is sent to all CEMSs which should receive the signal. In the following, the pricing signals are written to this MUC and are consumed by the CEMSs. The tariff server also receives energy balance forecasts from registered CEMSs, which consume the outgoing pricing signals. In real world scenarios, the information about future energy consumption will be used to optimize energy purchases on the global market. However, the implemented tariff server just outputs the received signal to the console in order to demonstrate the successful reception of the signal.

5.4.1 Configuration

Apart from general settings like server connection parameters, name, and day length, the main configuration file contains specific options to control the transmission of pricing signals. The user can select whether the profiles are chosen at random or a predefined sequence should be used to generate future signals. In addition, the time, when the profile change should be performed, can be defined. The time when the pricing signal should be sent can be configured in 3 different ways. First, the time of day can be defined directly in the configuration. That option is closest to reality, but for demonstration purposes it may be impractical to wait for this time. Even if a day is scaled down to 24 minutes, it can take nearly the full 24 minutes until the next signal is sent. Therefore, it is also possible to send the next pricing signal at a time which is specified at program start up or at the full hour after next. Another important parameter is the length of an interval within the day-ahead pricing signal. The interval length is the smallest amount of time, in which the price can not be changed. All configuration options, which are not common to all types of agents, are listed in Table 5.5. In addition, the configuration includes the aforementioned price profiles.

```
<PriceProfile id="2" active="true">
  <PricePoint time="12:00:00">0.228</PricePoint>
  <PricePoint time="18:00:00">0.200</PricePoint>
  <PricePoint time="00:00:00">0.198</PricePoint>
  <PricePoint time="06:00:00">0.210</PricePoint>
</PriceProfile>
```

Listing 5.2: Sample price profile with four time points

Parameter	**Description**
tariff-chatroom	Name of the chat room for multicasts of pricing signals, e.g. TS2tariffs@conference.energyretailer2.xmpp.local
sendTimeMode	Way to determine sending time of pricing signals. The following options are available: 1 ... Use send time from configuration 2 ... Full hour after the next 3 ... Read time at start up
sendTime	The time for sending day-ahead pricing signals; is only used when sendTimeMode is set to 1.
SendOffset	Time in minutes between the send time of the pricing signal and the start of the first interval within the signal.
PriceInterval	Length of a pricing interval in minutes.
ProfileSelectionType	Determines how the next price profile should be selected. Following options are available: 1 ... Random selection 2 ... Specified order If there is just one profile defined, the result of both options is the same.
profileReloadTime	Time when the next profile should be loaded

Table 5.5: Tariff server configuration

5.4.2 Architecture

Figure 5.3 shows the class diagram of the tariff server. The interesting part, which differs from the other agents, is the *TimeManagement* class. It is responsible for the timing of sending the pricing signal and changing the price profile. Calling the corresponding function within the *EnergyAdmin* is done via the *FunctionCaller* interface. The diagram also shows that there might be multiple price profiles in the configuration, but just one profile is active in the *EnergyAdmin* class. All other classes are already described in the previous sections.

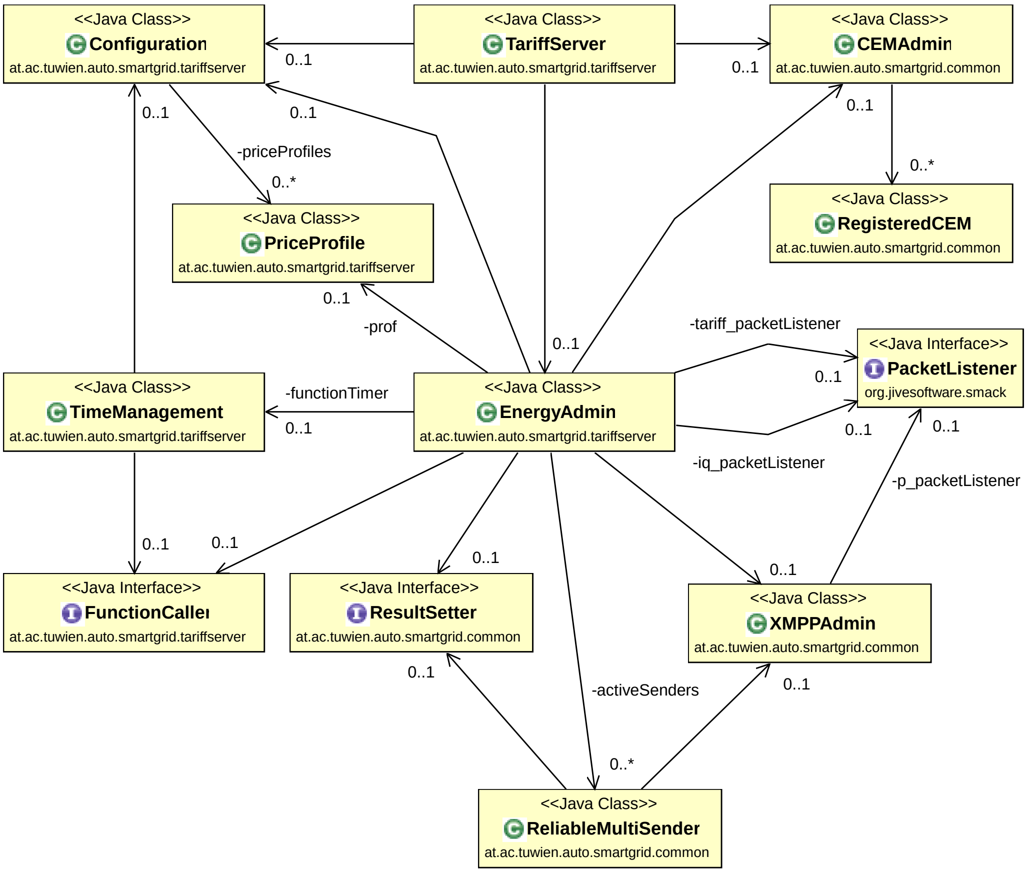

Figure 5.3: Simplified tariff server class diagram

5.5 Flexibility Operator

As already known, the flexibility operator recognizes bottlenecks in the grid and tries to negotiate intra-day flexibilities to solve them. Potential capacity problems are identified by aggregating all energy balance forecasts from the CEMSs. When such a forecast arrives, the flexibility operator merges it with historical forecasts from this CEMS and starts a time window which is intended to receive further energy balance forecasts from other CEMSs. The maximal length of this window is definable in the configuration file. It specifies the maximal waiting time because the time frame is closed earlier when projections from all CEMSs have been received. Each forecast is stored until the latest time interval within the signal is reached. Subsequently, all energy balance forecasts are combined to a total forecast which describes the planned grid capacity utilization. This total forecast is compared to a predefined schedule of available energy to identify the bottlenecks. In the case of recognized problems, a flexibility request for the corresponding time slots is sent to the registered CEMSs. Now, a time window for receiving flexibility offers is started. The end of this window is again user definable and is additionally used as expiration date of the flexibility request. When the end time is reached or offers or rejections from all CEMSs have been received, the best combination of offers is selected for ordering. For the demonstration setup, the optimal selection is defined as the cheapest option which solves the capacity problems. In real world, it may be useful to accept offers in a way such that there are safety margins between the planned utilization and the capacity limit of the grid. Afterwards, the orders are sent to the CEMSs, and other offers are canceled. This means that each CEMS gets at least one message from the flexibility operator, either an offer, a rejection, or both. If 2-step ordering mode is enabled, a timer thread is started to trigger the sending of the release signal after some definable time.

5.5.1 Configuration

Essentially, the additional configuration parameters of the flexibility operator are some time constraints, the ordering mode, and the available energy schedule. The schedule consists of pairs of time and kilowatt hours. The amount of energy is valid until the start time of the next pair. For the sake of simplicity the defined energy is available for each interval of the energy balance forecasts. An example with four pairs is shown in Listing 5.3.

```
<availableEnergy>
  <EnergyPoint time="12:00:00">280</EnergyPoint>
  <EnergyPoint time="18:00:00">200</EnergyPoint>
  <EnergyPoint time="00:00:00">180</EnergyPoint>
  <EnergyPoint time="06:00:00">200</EnergyPoint>
</availableEnergy>
```

Listing 5.3: Available energy schedule with four time points

When we assume that the interval length is set to 60 minutes, this schedule allows the usage of 280 kWh in each hour between 12:00 and 18:00. The same applies to the rest of the time slots. All other configuration parameters are summarized in Table 5.6.

Parameter	Description
order-mode	Select whether a release has to be sent after the flexibility order. 1-step ... No release required 2-step ... Release required
releaseInterval	Time interval in virtual minutes between order and release.
offerWaitTime	Maximal length of the time window for receiving offers in virtual minutes.
maxEBFWaitTime	Maximal length of the time windows for receiving energy balance forecasts in virtual minutes.
interval-granularity	Length of intervals within all types of signals in virtual minutes.

Table 5.6: Flexibility operator configuration

5.5.2 Architecture

Apart from the CEMS, the flexibility operator is the most complex agent in this implementation. However, the class diagram (see Figure 5.4) is relatively compact. The logic for the identification of bottlenecks and the negotiation of intra-day flexibilities is embedded in the *EnergyAdmin* class. The other classes are known from the previous sections.

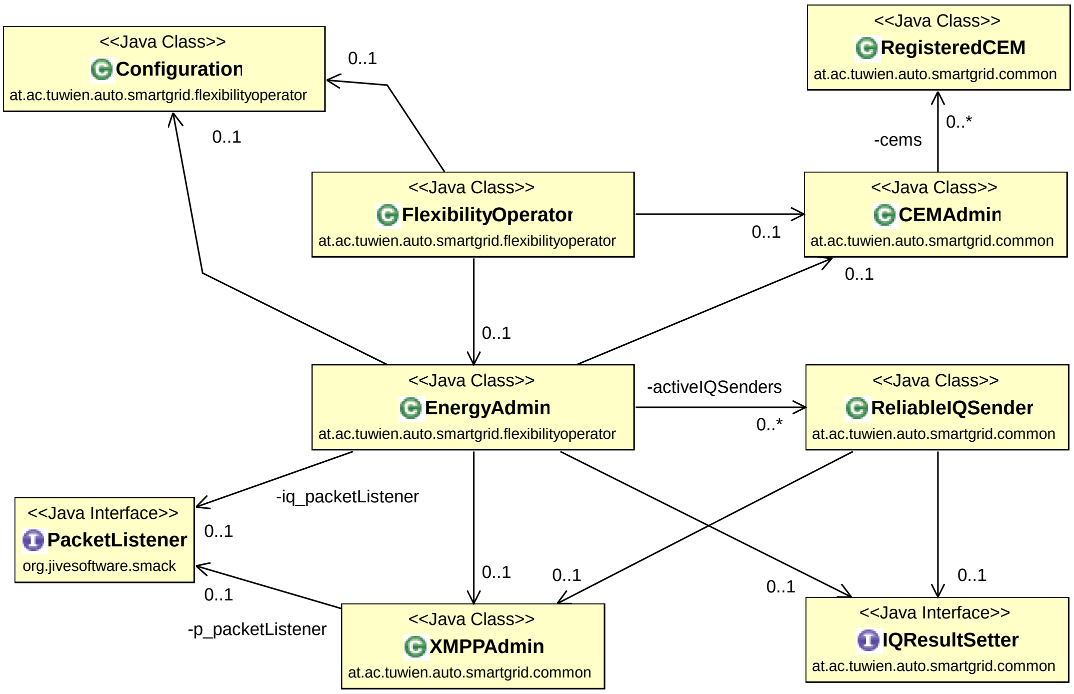

Figure 5.4: Simplified flexibility operator class diagram

5.6 Decentralized Energy Management System

Regarding the communication between smart buildings and smart grids, the DEMS implements a subset of the flexibility operator. Everything to do with computing bottlenecks within the grid can be omitted. In principle, requesting and ordering flexibilities works in the same way. The difference lies in the procedure which starts the negotiation of flexibilities, and how the offers are selected. Sending a flexibility request is triggered by the *sendRequest* command. Moreover, the time interval, for which the DEMS wants to buy flexibilities, is defined in a much easier manner. It just adds two hours to the current time and uses this time truncated to a full hour. The length of the interval is definable in the configuration file. Later in the procedure, the offer which should be ordered is randomly selected. In contrast to the flexibility operator, the DEMS does not receive any energy balance forecasts. All other parts, ranging from releasing orders via maintaining time windows to wait for offers are adopted from the flexibility operator.

5.6.1 Configuration

The close relationship to the flexibility operator is also seen in the settings. Actually, the only difference is the absence of the available energy data and the time interval which defines the waiting time for energy balance forecasts.

5.6.2 Architecture

The class diagram for the DEMS can be found in Figure 5.5. Comparing the diagram to Figure 5.4, one can see the identical structure of DEMS and flexibility operator once again.

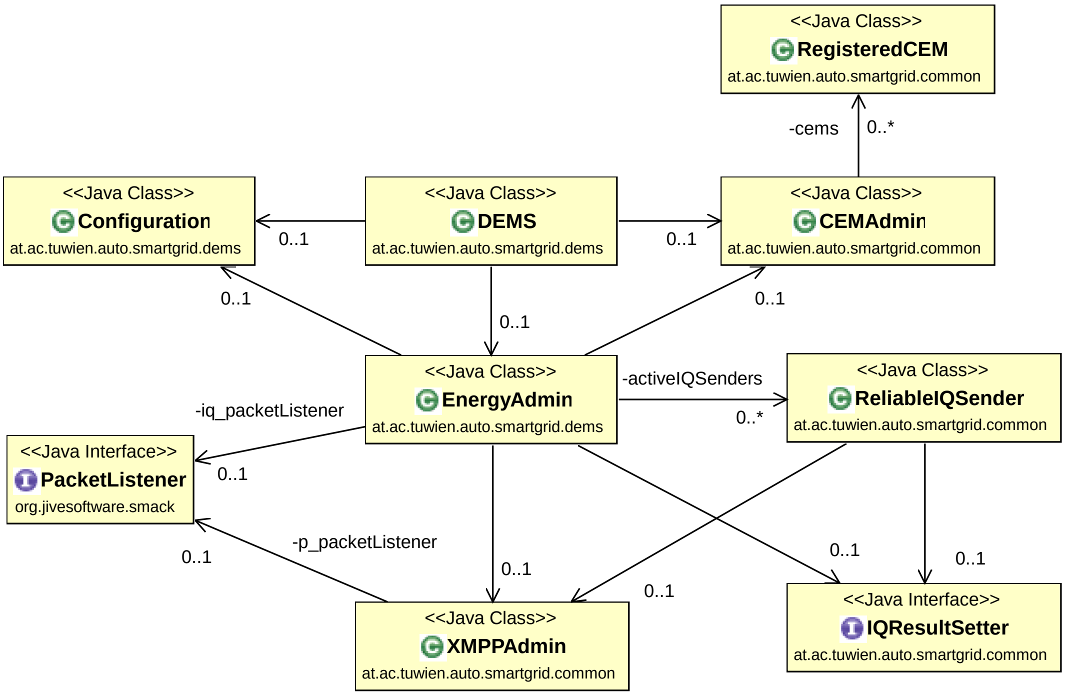

Figure 5.5: Simplified DEMS class diagram

5.7 Smart Low Voltage Grid Controller

As described in previous chapters, the SLVGC supplies the CEMSs with information about how to react in the case of voltage (or frequency) deviations. For demonstration purposes, voltage deviations have been chosen. On the other hand, it monitors the low-voltage power network and sends grid priority signals to the CEMSs of the affected grid section. Since there is no real power grid to inspect in the demonstration installation, sending either characteristic P(U) and Q(U) curves or a grid priority signal and its associated end signal has to be triggered manually by one of the following commands:

- *sendCurves* sends the P(U) and Q(U) curves to all registered CEMSs.
- *gridPriority* transmits a grid priority signal to the registered CEMSs.

- *endSignal* sends a signal, which indicates the end of a grid priority event.

In contrast to all the other signals, the characteristic curves are not defined by the use of the Energy Interoperation specification. When sending grid priority signals, the SLVGC knows which CEMSs have acknowledged the execution of the event. This information allows to send grid priority end signals exclusively to CEMSs which have reacted to the start signal. Although the grid priority signal supports events which last for a pre-defined duration, the signal is always sent with an open end. Therefore, it is necessary to send an end signal after some time. Otherwise the CEMSs would last in grid priority mode until the programs are terminated.

5.7.1 Configuration

In addition to common settings, configuration parameters for the SLVGC define the relative amount of requested load change in case of an emergency and all information which is included in the characteristic curve signal. A sample configuration for an active power curve is shown in Listing 5.4. The reactive power curve is configured in a similar way. All SLVGC specific configuration parameters are listed in Table 5.7.

```
<activePower>
  <Point voltage="180" power="-0.3" />
  <Point voltage="190" power="-0.2" />
  <Point voltage="200" power="-0.1" />
  <Point voltage="210" power="-0.05" />
  <Point voltage="220" power="0" />
  <Point voltage="230" power="0" />
  <Point voltage="240" power="0" />
  <Point voltage="250" power="0.05" />
  <Point voltage="260" power="0.1" />
  <Point voltage="270" power="0.2" />
  <Point voltage="280" power="0.3" />
</activePower>
```

Listing 5.4: Active Power P(U) curve

Parameter	Description
loadChangePercent	Requested load change in grid priority signals in percent, e.g. -50
gridSectionName	Name of the grid section
nominalVoltage	Nominal voltage in volt, e.g. 230
activePower	Contains data points for the active power curve
reactivePower	Contains data points for the reactive power curve

Table 5.7: SLVGC configuration

5.7.2 Architecture

In general, Figure 5.6 shows the standard architecture which is used for all agents. It is also shown that the active grid priority events are stored within the *EnergyAdmin*. These are required to find out which CEMSs are waiting for an end signal.

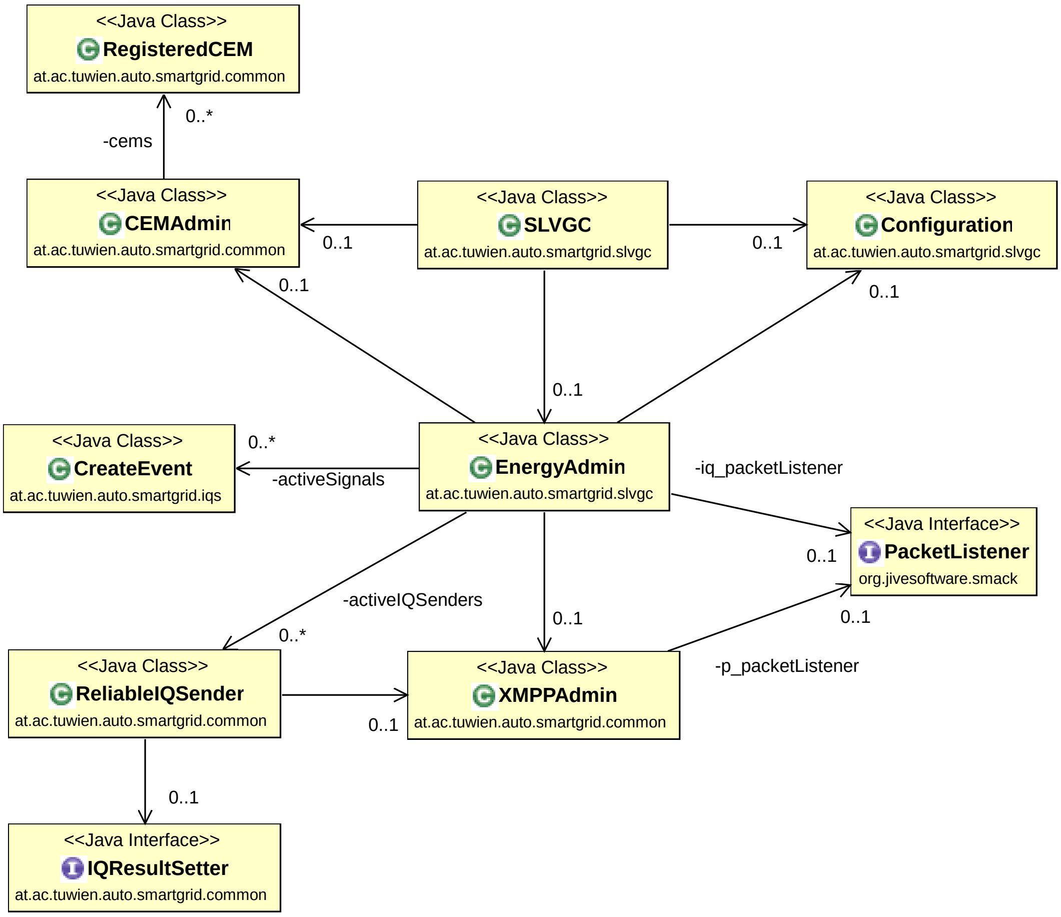

Figure 5.6: Simplified SLVGC class diagram

5.8 Customer Energy Management System

As already known, the CEMS tries to optimize the energy consumption of an entire building or just some facilities throughout the day. For this purpose, an energy schedule has to be maintained. This schedule describes how much energy is consumed by each device at any time. The devices are defined in the configuration file together with at least one operating mode. Such a mode has an identification number and is associated with an energy consumption value. The modes are ordered incrementally. Thus, modes with lower ID consume less energy. The energy schedule itself consists of time intervals covering a whole day, which are called slots. A slot in turn is composed of multiple tasks, which describe the operation of a device in a specific mode. At program start up, each task runs in its specified initial operation mode. Additionally, there are two more parameters which describe the flexibility of the task by providing a minimal and a maximal mode. The currently selected modes are affected by all use cases accept the characteristic curves.

At program start up, all necessary MUCs are joined after reading the configuration and connecting to the XMPP server. The name of the chats are defined within relations to the agents in the grid. When a day-ahead pricing signal arrives, the CEMS starts optimizing its energy schedule. For simulating that process, the schedule has an elasticity parameter which specifies the degree of impact of the electricity price. This parameter allows values from -1 to 1. A positive value would lead to more consumption when the price becomes higher. Therefore, it makes sense to set it to a negative value (e.g. -0.5), as the customers are interested in a cheap electricity bill. The desired change in consumption in each individual slot is calculated by the following formula:

$$\text{consumption change} = \text{elasticity} * \text{relative price change} * \text{current consumption}$$

When the first pricing-signal arrives, there is of course no previous price. In this case, the relative price change is set to 0. Hence, also the consumption change is 0. After that calculation, all tasks have to be set to a suitable operating mode. This is done by decreasing or rather increasing the mode until the desired consumption is reached. However, it must be ensured that the minimum and maximum mode are not exceeded. Therefore, the change is not always carried out in its entirety. This procedure is not well suited for real world usage, because interactions between different slots are not taken into account. Normally, a consumption change in one slot would lead to reverse changes in other slots. Here the chosen procedure provides a price elastic consumption rate, which is enough to demonstrate the communication between the agents. The resulting energy schedule is used for the creation of an energy balance forecast, which is sent back to the tariff server and to the flexibility operator, if available. In the case of reception of a flexibility request, the CEMS checks whether it is possible to decrease the operation mode of some tasks for the requested time intervals. This can only be achieved, if not all tasks are already running in its minimal mode. Subsequently, the potential savings are sent to the requesting agent (flexibility operator or DEMS). When it is not possible to provide any flexibilities, a rejection is sent instead. Additionally, the offer is temporarily stored until the first interval with an offered consumption adoption starts. This time is also known as the expiration date of the offer, which is included in the transmitted signal. Under normal circumstances, an order or a rejection according to the offer will be received somewhat later. While a rejection causes the deletion of the stored offer, the reaction to the order depends on the need for a further release in the case of activated 2-step ordering mode. If no release is required, the current energy schedule is adopted immediately. Otherwise, the order will be stored, and the changes take place when the associated release is received. Note, that the adaption of the energy schedule is nothing more than changing the operating modes of the tasks.

Grid priority signals from the SLVGC are executed immediately. The consumption is lowered (or increased) by the given relative value by changing the operating modes of all running tasks. In contrast to the reaction to flexibility requests, the minimal and maximal modes are ignored. Before the change is applied, a backup of the current energy schedule is created. This enables a return to normal operation when the end signal arrives. The reception of characteristic P(U) and Q(U) curves leads to a simple output of the contained data points to the console.

For demonstration purposes, the program also includes a thread which represents the building management system. The purpose of this thread is simply to output the current energy consumption to the console.

5.8.1 Configuration

Besides the common XMPP and virtual time settings, the configuration of the CEMS consists of relations to other agents, schedules, and devices. Each relation has an ID and includes the mRID and the JID of the corresponding remote agent. When required, also the information for the connection to a MUC is included. A section of an energy schedule configuration is shown in Listing 5.5.

```
<schedules>
  <schedule id="1" active="true" elasticity="-50">
    ....
    <slot time="12:00:00">
      <task device="light" initial="1" min="1" max="2"/>
      <task device="hvac" initial="3" min="2" max="3"/>
    </slot>
    <slot time="13:00:00">
      <task device="light" initial="1" min="1" max="2"/>
      <task device="hvac" initial="4" min="2" max="4"/>
    </slot>
    ....
  </schedule>
</schedules>
```

Listing 5.5: CEMS energy schedule

5.8.2 Architecture

A simplified version of the CEMS architecture is shown in the class diagram in Figure 5.7. Although the CEMS takes part in all use cases, most of the functionality is encapsulated in the *EnergyAdmin* class. Architectural specifics of this particular program are the schedule, the devices, the slots, the modes, and the *BMS* class which represents the building management system. Additionally, it can be seen that there are packet listeners for all unicasts and multicasts which are sent by other agents in the grid.

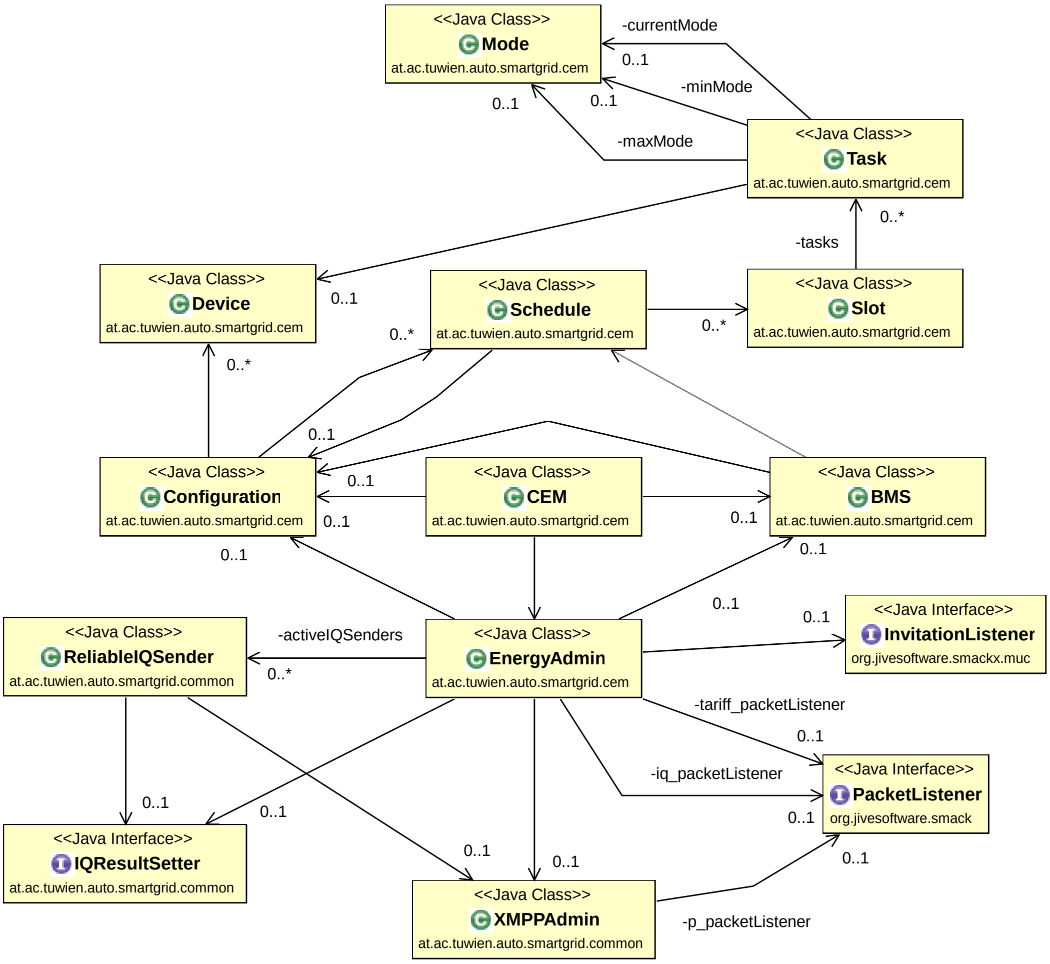

Figure 5.7: Simplified CEMS class diagram

5.9 Execution

In the following, an exemplary execution of the providing intraday flexibilities use case is shown from the perspective of the flexibility operator. First, Listing 5.6 illustrates the reception of energy balance forecast signals from CEMS 1, CEMS 2, and CEMS 3. Afterwards, bottlenecks are identified for six time slots. Therefore, the flexibility operator sends flexibility request signals to the CEMSs in the corresponding grid area (see Listing 5.7). Listing 5.8 shows the reception of flexibility offer signals from the CEMSs. In this case, all CEMSs are able to offer load curtailments. Finally, the flexibility operator orders the bidding from CEMS 2 and rejects all the other offers. Listing 5.9 includes the console output belonging to this actions together with the sending of a subsequent flexibility order release signal.

```
Energy Balance Forecast
Received at: 2051-10-22T09:00:42.780
RequestID: RCEMS20d766030-4622-4f70-afe3-bb77400f087e, CEMS: CEMS2, Tender: TCEM
S2254ffd72-8dfd-4baf-be8d-594d1ae22cea, Product: PCEMS230f9e550-51c3-41dc-b977-6
d413d19e277, Created: 2051-10-22T09:00:17
[Info]: Created Tender for request id RCEMS20d766030-4622-4f70-afe3-bb77400f087e sent
EBFs received: 1/ 3
[Info]: Energy Balance Forecast Timewindow started

Energy Balance Forecast
Received at: 2051-10-22T09:00:53.100
RequestID: RCEMS3eacfb46c-5696-44e5-a0bb-19559a8ee9cb, CEMS: CEMS3, Tender: TCEM
S315f07ccd-c9de-4b22-be26-abd8a1374da7, Product: PCEMS3ba57b9d0-bedb-4ed7-bdcb-a
1d2d932d0b3, Created: 2051-10-22T09:00:26
[Info]: Created Tender for request id RCEMS3eacfb46c-5696-44e5-a0bb-19559a8ee9cb sent
EBFs received: 2/ 3

Energy Balance Forecast
Received at: 2051-10-22T09:00:54.600
RequestID: RCEMS1c5764913-0788-4f5b-922d-2014e26ecc53, CEMS: CEMS1, Tender: TCEM
S1de00a5a4-1812-4bd0-8040-d0598cdb10d9, Product: PCEMS1c7a0dc63-8162-43d0-943c-7
a00b2713e3b, Created: 2051-10-22T09:00:25
[Info]: Created Tender for request id RCEMS1c5764913-0788-4f5b-922d-2014e26ecc53 sent
EBFs received: 3/ 3
```

Listing 5.6: Flexibility operator receives energy balance forecast signals

```
Identified Bottlenecks:
2051-10-22T11:00
2051-10-22T14:00
2051-10-22T15:00
2051-10-23T06:00
2051-10-23T08:00
2051-10-23T09:00
--------------------------------------------------------
[Info]: Timewindow for offers open until 2051-10-22T09:16:12.420
[Info]: Flexibility Request successfully sent to cem3@gridoperator1.xmpp.local/iectc57
[Info]: Flexibility Request successfully sent to cem2@gridoperator1.xmpp.local/iectc57
[Info]: Flexibility Request successfully sent to cem1@gridoperator1.xmpp.local/iectc57
```

Listing 5.7: Flexibility operator identifies bottlenecks

```
[Info]: Flexibility Offer received at 2051-10-22T09:01:15.120
Tender TCEMS30826110f-1b2e-42ed-b85b-a5538eef2508:
2051-10-22T11:00: -60.0
2051-10-22T14:00: -70.0
2051-10-22T15:00: -70.0
2051-10-23T06:00: -60.0
2051-10-23T08:00: -60.0
2051-10-23T09:00: -60.0
[Info]: Created Tender for request id RCEMS39fcffe42-e58f-4635-85d6-677d2f85e2c4 sent
[Info]: Offers received: 1/ 3

[Info]: Flexibility Offer received at 2051-10-22T09:01:15.420
Tender TCEMS29871c415-c3cd-435b-9d0a-0e7bd52f0eb4:
2051-10-22T11:00: -80.0
2051-10-22T14:00: -90.0
2051-10-22T15:00: -90.0
2051-10-23T06:00: -80.0
2051-10-23T08:00: -80.0
2051-10-23T09:00: -80.0
[Info]: Created Tender for request id RCEMS25a4f3f77-ff7b-4c95-ab86-94ac1598ab6f sent
[Info]: Offers received: 2/ 3

[Info]: Flexibility Offer received at 2051-10-22T09:01:16.980
Tender TCEMS13e3fc5bb-60f0-405f-a61b-8ceeb1c50516:
2051-10-22T11:00: -20.0
2051-10-22T14:00: -90.0
2051-10-22T15:00: -90.0
2051-10-23T06:00: -20.0
```

```
2051-10-23T08:00: -20.0
2051-10-23T09:00: -20.0
[Info]: Created Tender for request id RCEMS15187952e-11fa-4a6a-8477-b51fd143a84b sent
[Info]: Offers received: 3/ 3
[Info]: All Orders received , time window closed
```

Listing 5.8: Flexibility operator receives flexibility offer signals

```
[Info]: Sending Flexibility order to cem2@gridoperator1.xmpp.local...
[Info]: Sending Cancel flexibility offer to: cem3@gridoperator1.xmpp.local/iectc57...
[Info]: Sending Cancel flexibility offer to: cem1@gridoperator1.xmpp.local/iectc57...
[Info]: Flexibility Order signal successfully sent to cem2@gridoperator1.xmpp.local/iectc57
[Info]: Cancel Offer signal successfully sent to cem3@gridoperator1.xmpp.local/iectc57
[Info]: Cancel Offer signal successfully sent to cem1@gridoperator1.xmpp.local/iectc57
[Info]: Sending Flexibility release to cem2@gridoperator1.xmpp.local...
[Info]: Flexibility Release signal successfully sent to cem2@gridoperator1.xmpp.local/iectc57
```

Listing 5.9: Flexibility operator orders an offer

Additionally, the transmission of grid priority parameters to three CEMSs is shown in Listing 5.10 (console output of the SLVGC) and Listing 5.11 (console output of one CEMS).

```
[Info]: Grid control parameters successfully sent to cem2@gridoperator1.xmpp.local/iectc57
[Info]: Grid control parameters successfully sent to cem1@gridoperator1.xmpp.local/iectc57
[Info]: Grid control parameters successfully sent to cem3@gridoperator1.xmpp.local/iectc57
```

Listing 5.10: SLVGC sends grid priority parameters signals to 3 CEMSs

```
Grid control parameters with nominal voltage 230.0 received
Active Power:
    180.0 V: -0.3
    190.0 V: -0.2
    200.0 V: -0.1
    210.0 V: -0.05
    220.0 V: 0.0
    230.0 V: 0.0
    240.0 V: 0.0
    250.0 V: 0.05
    260.0 V: 0.1
    270.0 V: 0.2
    280.0 V: 0.3
Reactive Power:
    180.0 V: -0.32
    190.0 V: -0.22
    200.0 V: -0.12
    210.0 V: -0.052
    220.0 V: 0.0
    230.0 V: 0.0
    240.0 V: 0.0
    250.0 V: 0.052
    260.0 V: 0.12
    270.0 V: 0.22
    280.0 V: 0.32
```

Listing 5.11: CEMS receives grid control parameters signal

CHAPTER 6

Conclusion and Outlook

6.1 Summary

Within this thesis, appropriate protocols and emerging standards for communication between smart buildings and smart grids have been presented, compared, and finally implemented in form of a proof-of-concept implementation. The process of finding a suitable solution is summed up in the following paragraphs.

First, the term SGUI is introduced in Chapter 2 for the denomination of the interface where data exchange between electricity customers and power grid service providers takes place. Important data exchange across the SGUI include: demand response (DR) events like load sheds or load shifts, pricing signals, market transactions, meter data, load parameters (demand, load forecasts, or storage availability), or emergency signals provided by the grid operators. Afterwards, requirements, which have to be fulfilled by technologies for communication across the SGUI, are defined. Important requirements are the capability to integrate a large number of clients into the network, the optimization for a large number of small data packets, the usage of XML, asynchronous data exchange without the necessity for polling, and a decentralized network architecture. The next fundamental step is to identify and analyze standards for defining the structure of messages, which have to be exchanged across the SGUI. These standards are Energy Interoperation (EI), Open Automated Demand Response (OpenADR), Energy Market Information Exchange (EMIX), and the IEC Common Information Model (CIM). While EMIX is used in EI to represent products, quantity and price, OpenADR 2.0 defines concrete profiles of EI. The last part of Chapter 2 was the presentation of the Extensible Messaging and Presence Protocol (XMPP) as a promising application layer protocol for transportation of the messages.

Next, use cases for tasks which require information exchange between smart buildings and smart grids have been identified. Additionally, the involved communication partners are presented. These are the CEMSs in the sphere of the customer as well as flexibility operators, DEMSs, SLVGCs, and tariff servers on the grid side. The following use cases were determined as the most important ones in the European energy market.

Energy balance forecasts in order to inform power service providers about planned energy consumption.

Day-ahead pricing for propagation of dynamic pricing information for one day in advance.

Providing intraday flexibilities to negotiate energy savings in case of imminent bottlenecks in the grid.

Grid priority signals for commands to lower or increase energy consumption in case of emergency situations.

Voltage control characteristics are used as a more conventional method of grid priority signals. They include data about how to adapt load in case of voltage or frequency deviations.

Attendance or weather changes can change the planned energy consumption plans and therefore require the transmission of new energy balance forecasts.

All of them are elaborated in detail. Afterwards, the signals, which are necessary for the use cases, are defined. The first step of this task is to specify the information which has to be included in each signal. Additionally, the ability of the previously presented standards to model these signals is investigated. It turns out that EI and CIM can be utilized to construct most of the signals. For the definition of grid priority signals also OpenADR is a good option. Because of the reason that there is no standardized profile of CIM that defines communication with customers, EI has been selected as the best available solution. Also the possibility to easily extend the messages with additional information like carbon footprints or the origin of energy leads to this decision. The only standardization gap that is identified is the missing of a signal for voltage control characteristics.

Finally, a proof-of-concept implementation was realized to demonstrate the use cases and the underlying communication mechanisms. A model of a smart grid consisting of five XMPP servers, five CEMSs, a flexibility operator, two DEMSs, two SLVGCs, and two tariff servers is created with overall 18 Raspberry Pis. On of these devices is used for supporting services like DNS, NTP, or DHCP.

6.2 Future work

Based on this thesis, some open topics shall be discussed or implemented in the future. A possible configuration process for CEMSs is already presented in Chapter 3.3. However, a final, practicable procedure has to be defined in accordance with utilities and service providers. Moreover, the threat of denial of service attacks is already mentioned but needs to be addressed in detail.

Security is, in general, a crucial topic for communication between smart buildings and smart grids. The three classic aims of security are to ensure confidentiality, integrity, and availability.

Confidentiality defines a set of rules to prevent unauthorized parties from accessing information. Integrity means that data has to be consistent over time. Last but not least, availability targets at providing service without any breakdowns. Other security related topics that must be examined are data freshness, authentication, and authorization. All these qualities are expected from the smart grid infrastructure. A key technology to provide security could be cryptography. Additionally, transmitted messages can be signed. Availability is supported by the properties of XMPP. Especially redundancy and the decentralized architecture help to prevent a single point of failure. Overall, it will be an important task to develop a integrated security concept for smart grids.

Another open question that goes towards privacy is to clarify whether it is necessary to hide the participating users within MUCs. Remember that this work uses MUCs to propagate prices from tariff servers to customers. Although it is possible to hide JIDs in these MUCs, nicknames and the number of connected clients are still visible to all participants. If this is a problem, a possible solution could be provided by a periodic change of nicknames and the usage of a random number of dummy clients in order to avoid inferences to actual customers.

A useful extension of the proof-of-concept implementation could be an extended monitoring tool. The current demonstration environment displays information only in textual form on the command line. Such a monitoring tool could also use the XMPP infrastructure to exchange data between all Raspberry Pis. Additionally, scalability tests can and should be performed, e.g. by appropriate simulation scenarios. These tests are used to investigate the performance and the behavior of the presented concept for application in large grids with thousands of participants.

List of Figures

List of Tables

Listings

Bibliography

[1] OpenADR Alliance. *OpenADR 2.0 Profile Specification A Profile*, 2013.

[2] OpenADR Alliance. *OpenADR 2.0 Profile Specification B Profile*, 2013.

[3] German Association of Energy BDEW and Water Industries. *Smart Grids Traffic Light Concept - Functional Interaction between the Market and the Regulated Sphere*, 2014.

[4] State Grid Cooperation China. *SGCC Framework and Roadmap for Strong and Smart Grid Standards*, 2010.

[5] Federal Energy Regulatory Commission. *Assessment of Demand Response and Advanced Metering*, 2006.

[6] Federal Energy Regulatory Commission. *2007 Assessment of Demand Response and Advanced Metering*, 2007.

[7] B. Desruisseaux. *RFC 5545: Internet Calendaring and Scheduling Core Object Specification (iCalendar).* Internet Engineering Task Force, Sep 2009.

[8] J. H. Dudley and M. A. Piette. Solutions for summer electric power shortages: Demand response and its application in air conditioning and refrigerating systems. *In Refrigeration, Air Conditioning, & Electric Power Machinery*, 29(1):1–4, 2008.

[9] EURELECTRIC. *EURELECTRIC Views on Demand-Side Participation*, 2011.

[10] C.W. Gellings. The concept of demand-side management for electric utilities. *In Proceedings of the IEEE*, 73(10):1468–1470, Oct 1985.

[11] Girish Ghatikar and Rolf Bienert. Smart grid standards and systems interoperability: A precedent with openadr. In *Grid-Interop 2011*, Phoenix, AZ, 12/2011 2011.

[12] CEN-CENELEC-ETSI Smart Grid Coordination Group. *Sustainable Processes*, 2012.

[13] CEN-CENELEC-ETSI Smart Grid Coordination Group. *Overview of the main concepts of flexibility management*, 2014.

[14] Smart Grid Interoperability Panel B2G/I2G/H2G Domain Expert Working Groups. *Customer Energy Services Interface White Paper Version 1.0*, 2011.

[15] V.C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G.P. Hancke. Smart grid technologies: Communication technologies and standards. *In Industrial Informatics, IEEE Transactions*, 7(4):529–539, Nov 2011.

[16] IEC. *Framework for energy market communications - Part 351: CIM European market model exchange profile*. IEC, 2013.

[17] Electric Power Research Institute. *IntelliGrid Common Information Model Primer: Second Edition*, 2013.

[18] K. Kuroda, T. Ichimura, and R. Yokoyama. An effective evaluation approach of demand response programs for residential side. In *9th IET International Conference on Advances in Power System Control, Operation and Management (APSCOM 2012)*, pages 1–6, Nov 2012.

[19] R. Kyusakov, J. Eliasson, J. van Deventer, J. Delsing, and R. Cragie. Emerging energy management standards and technologies - challenges and application prospects. In *2012 IEEE 17th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–8, Sept 2012.

[20] F. Lobo, A. Cabello, A. Lopez, D. Mora, and R. Mora. Distribution network as communication system. In *SmartGrids for Distribution, 2008. IET-CIRED. CIRED Seminar*, pages 1–4, June 2008.

[21] A. Melnikov and K. Zeilenga. *RFC 4422: Simple Authentication and Security Layer (SASL)*. Internet Engineering Task Force, Jun 2006.

[22] R. Millard, P. Saint-Andre, and P. Meijer. *XEP-0060: Publish-Subscribe*. XMPP Standards Foundation, 2010.

[23] OASIS. *Energy Market Information Exchange (EMIX) Version 1.0*, 2012.

[24] OASIS. *Energy Interoperation Version 1.0*, 2014.

[25] National Institute of Standards and Technology. *NIST Framework and Roadmap for Smart Grid Interoperability Standards,Release 2.0*, 2012.

[26] P. Palensky and D. Dietrich. Demand side management: Demand response, intelligent energy systems, and smart loads. *In IEEE Transactions on Industrial Informatics*, 7(3):381–388, Aug 2011.

[27] IEC PC118. *IEC 62939 TR: Smart grid user interface*. IEC.

[28] D. Peintner, R. Kyusakov, J. Schneider, and T. Kamiya. *Efficient XML Interchange (EXI) Format 1.0 (Second Edition)*, 2014.

[29] Mike Pichler. Communication patterns for demand side flexibility. In *Symposium on Communication for Energy Systems*, 2014.

[30] Peter Saint-Andre. *RFC 3920: Extensible Messaging and Presence Protocol (XMPP): Core*. Internet Engineering Task Force, October 2004.

[31] Peter Saint-Andre. *RFC 3921: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*. Internet Engineering Task Force, October 2004.

[32] Peter Saint-Andre. *XEP-0205: Best Practices to Discourage Denial of Service Attacks*. XMPP Standards Foundation, 2009.

[33] Peter Saint-Andre, Kevin Smith, and Remko Tronon. *XMPP: The Definitive Guide Building Real-Time Applications with Jabber Technologies*. O'Reilly Media, Inc., 2009.

[34] D. Schachinger, S. Gaida, and W. Kastner. Smart grid communication at the interface of customer buildings with focus on demand response. *In 2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST)*, 2015. submitted/accepted.

[35] IEC TC57. *IEC 61968-11:2013 Application integration at electric utilities - System interfaces for distribution management - Part 11: Common information model (CIM) extensions for distribution*. IEC TC57, 2013.

[36] IEC TC57. *IEC 61970-301:2013 Energy management system application program interface (EMS-API) - Part 301: Common information model (CIM) base*. IEC TC57, 2013.

[37] IEC TC57. *IEC 62325-301:2014 Framework for energy market communications - Part 301: Common information model (CIM) extensions for markets*. IEC TC57, 2014.

[38] IEC TC57. *System interface between customer energy manager and power management system - Use cases and requirements*. IEC TC57, 2014.

[39] J.C. Vasquez, J.M. Guerrero, J. Miret, M. Castilla, and L.G. de Vicuna. Hierarchical control of intelligent microgrids. *In Industrial Electronics Magazine, IEEE*, 4(4):23–29, Dec 2010.

[40] Ye Yan, Yi Qian, H. Sharif, and D. Tipper. A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *In Communications Surveys Tutorials, IEEE*, 15(1):5–20, First 2013.