



TECHNISCHE  
UNIVERSITÄT  
WIEN  
Vienna | Austria



DISSERTATION

# Convergent Interoperability Stack for Smart Grid ICT Infrastructures

carried out for the purpose of obtaining the degree of Doctor technicae (Dr. techn.)  
submitted at TU Wien, Faculty of Mechanical and Industrial Engineering,

by

**Dipl.-Ing. Armin VEICHTLBAUER**

Mat.No.: 08920566

under the supervision of

**Privatdoz. Dipl.-Ing. Dr.techn. Thomas I. Strasser**

Institute for Mechanics & Mechatronics (E325)

Division for Control and Process Automation

reviewed by

**Univ.-Prof. Dr. Wolfgang Kastner**

Technische Universität Wien

Institute of Computer Engineering

Treitlstraße 3

1040 Vienna, Austria

**Univ.-Prof. Dr. Sebastian Lehnhoff**

Carl von Ossietzky Universität Oldenburg

Department of Computing Science

Ammerländer Heerstraße 114-118

26129 Oldenburg, Germany

Vienna, September 2023



I confirm, that going to press of this thesis needs the confirmation of the examination committee.

### *Affidavit*

I declare in lieu of oath, that I wrote this thesis and performed the associated research myself, using only the literature cited in this volume. If text passages from sources are used literally, they are marked as such.

I confirm that this work is original and has not been submitted elsewhere for any examination, nor is it currently under consideration for a thesis elsewhere.

I acknowledge that the submitted work will be checked electronically-technically using suitable and state-of-the-art means (plagiarism detection software). On the one hand, this ensures that the submitted work was prepared according to the high-quality standards within the applicable rules to ensure good scientific practice “Code of Conduct” at the TU Wien. On the other hand, a comparison with other student theses avoids violations of my personal copyright.

---

*Place and Date*

---

*Signature*

# Acknowledgment

First, I would like to thank my thesis supervisor, Thomas I. Strasser, for his invaluable support throughout the dissertation writing. His advice was always to the point, never too much, never too few, and guided me well through the whole process. Thomas pushed me, where necessary, and gave me the time and space, as I needed them. Moreover, he actively participated in the paper writing, ensuring that they meet the quality they required. Finally, he had the lead of the research project OpenNES, in which many of the main concepts of this thesis have been developed.

I also would like to thank all my colleagues in the projects OpenNES, VirtueGrid, and Storage Cluster South Burgenland - thank you Oliver, Ulrich, Filip, Friederich, Ferdinand, Peter, Julian, Lukas, Gerald, Michael, the Christophs, the Andis, and all the others who were participating in one or another form in the work conducted there. The projects formed the basis, on which my investigation could take place. Dominik and Ulrich I want to thank for preparing the ground for these projects.

Furthermore, I am grateful for the support I received in my company, the University of Applied Sciences Upper Austria, especially from Christoph, who gave me the required time to work on my personal development. I also want to thank Gerald and Gabi, for providing me help with project proposals and management, and Katherine, for many fruitful discussions. Moreover, I am grateful for the financial support which I received by company's dissertation program under the project name InterGrid, granted by the State of Upper Austria. Also, I am grateful for the grants financing the other mentioned research projects.

Finally, I want to thank my family for their backing and patience in the last years. My wife Christina was always perfectly understanding a dissertant's moods, as she was experiencing this already before. She was always open to discuss issues related to project work, publishing, and all the other aspects of a dissertation, and gave me confidence and belief in the final success. Last, but not least, my daughter Jana taught me not to lose the view on the important things in life.

# Kurzfassung

Das “Smart Grid”, also die Verbindung einer Infrastruktur zur Übertragung und Verteilung elektrischer Energie mit einer entsprechenden Informations- und Kommunikationstechnik (IKT) Infrastruktur, soll in Zukunft die Stabilität des Stromnetzes bei gleichzeitiger Integration erneuerbarer (meist volatiler) Energiequellen sicherstellen.

Die Grundproblematik dabei ist in erster Linie die starke Heterogenität der Stakeholder (Energieversorger, Netzbetreiber, Endkunden, Gerätehersteller, Anlagenbetreiber, Behörden, etc.). Eine IKT Infrastruktur muss all diesen Gruppen ermöglichen, ihre jeweiligen Applikationen unter definierten Qualitätsbedingungen (maximal erlaubte Latenz, benötigte Verfügbarkeit, etc.) auszuführen.

Trotz vieler Bemühungen zur Standardisierung fehlt ein generischer Ansatz (unabhängig von einem konkreten Use Case) zur Interoperabilität all dieser Teilsysteme nach wie vor. In abgeschlossenen Umgebungen wie in der Home Automation existieren Frameworks, die Funktionalitäten wie ein gemeinsames Datenmodell oder eine einheitliche Adressierung für Knoten bereitstellen.

Solche Frameworks sind im hochgradig verteilten Smart Grid jedoch kaum geeignet. Die vorliegende Dissertation strebt daher eine leichtgewichtige Lösung ohne zentrale Frameworks an. Dabei sollen die benötigten Middleware-Funktionen über einen geeigneten Protokollstack abgebildet werden, der im Wesentlichen eine End-to-End Charakteristik aufweist und wenige zentrale Services benötigt.

Somit kann eine offene Meta-Architektur für ein verteiltes “System of Systems” geschaffen werden, die minimale Zugangshürden für die beteiligten Stakeholder aufweist. Der Protokollstack zum Datenaustausch zwischen den Stakeholdern muss die notwendige Funktionalität abdecken, um Interoperabilität der Applikationen der Stakeholder sicherzustellen. Die Algorithmik beinhaltet daher Datenstrukturen genauso wie Zustandsinformationen und Protokollabläufe.

Schließlich wird der verwendete Protokollstack in dieser Dissertation auch anhand von konkreten Szenarien validiert und evaluiert. Die Evaluierung wertet die Ergebnisse der durchgeführten Testszenarien hinsichtlich Skalierbarkeit, Kopplungsstärke, Rechtemanagement, Konkurrenzbedingungen, Vertrauen, Sicherheit, Adressierung, Zustandsbehaftung, Overlay-Architekturen, und Möglichkeiten zur Virtualisierung aus.

# Abstract

The “smart grid”, i.e., the combination of an infrastructure for the transmission and distribution of electrical energy with a corresponding information and communication technology (ICT) infrastructure, is intended to ensure the stability of the power grid in the future while at the same time enabling the integration of renewable (mostly volatile) energy sources.

The basic problem here is primarily the strong heterogeneity of the stakeholders (energy suppliers, grid operators, customers, equipment manufacturers, system operators, authorities, etc.). An ICT infrastructure must enable all these groups to run their respective applications under defined quality conditions (limited latency, required availability, etc.).

Despite many standardization efforts, a generic approach (independent of a specific use case) for the interoperability of all these subsystems is still missing. In closed environments such as in home automation, frameworks exist that provide functionalities such as a common data model or consistent addressing for nodes.

However, such frameworks are hardly suitable in the highly distributed smart grid. Therefore, this dissertation aims at a lightweight solution without central frameworks. The required middleware functions are to be mapped to a suitable protocol stack, which essentially has an end-to-end characteristic and requires few central services.

Thus, an open meta-architecture for a distributed “system of systems” can be created with minimal access barriers for the participating stakeholders. The protocol stack for data exchange between stakeholders must cover the necessary functionality to ensure interoperability of the stakeholders’ applications. The algorithmic base therefore includes data structures as well as state information and protocol sequences.

Finally, the used protocol stack is also validated and evaluated in this dissertation, based on concrete scenarios. The evaluation analyses the results of the executed test scenarios with respect to scalability, coupling strength, rights management, concurrency, trust, security, addressing, statefulness, overlay architectures, and virtualization possibilities.

# Contents

<b>1</b>	<b>Overview</b>	<b>1</b>
1.1	Current Situation and Motivation . . . . .	1
1.1.1	Problem Statement . . . . .	2
1.1.2	Related Work . . . . .	3
1.2	Research Goals . . . . .	8
1.2.1	Detailed Research Questions . . . . .	8
1.2.2	Scope of Work . . . . .	10
1.2.3	Expected Results . . . . .	11
1.3	Methodological Approach . . . . .	12
1.3.1	Work Packages and Phases . . . . .	12
1.3.2	Research Design . . . . .	13
1.3.3	X-Architecture Model . . . . .	14
1.3.4	Requirements Analysis Process . . . . .	15
1.3.5	Use Cases for Demonstrators . . . . .	17
1.3.6	Realization of Demonstrators . . . . .	21
1.3.7	Security by Design and Rights Management . . . . .	26
1.4	Results of Work . . . . .	27
1.4.1	Test Execution and Validation . . . . .	27
1.4.2	Specification Items . . . . .	33
1.4.3	Analysis of Research Questions . . . . .	34
1.5	Summary of Scientific Publications . . . . .	40
1.6	Scientific Contribution . . . . .	43
1.7	Conclusions and Further Work . . . . .	44
<b>2</b>	<b>Publications</b>	<b>45</b>
2.1	Publication 1 . . . . .	45
2.2	Publication 2 . . . . .	81
2.3	Publication 3 . . . . .	109
	<b>Bibliography</b>	<b>131</b>

---

<b>A</b>	<b>List of Scientific Publications</b>	<b>139</b>
A.1	Peer-reviewed International Journals . . . . .	139
A.2	Peer-reviewed Conference Proceedings . . . . .	140
A.3	Invited Papers and Book Chapters . . . . .	141
<b>B</b>	<b>Abbreviations</b>	<b>142</b>



# Chapter 1

## Overview

The power grid of the future, the smart grid [1], represents a highly complex system of systems, with individual subsystems often created by different manufacturers. In order to take advantage of the smart grid, these different subsystems must be able to cooperate with each other. The interoperability of subsystems from different manufacturers requires standardization at various levels [1], from the networking of used components, over syntactic and semantic interoperability (related to transport or application layer protocols of the Open Systems Interconnection (OSI) model [2]), to common business models and aligned business goals.

The concrete technical problems range from standardized addressing schemes for smart grid resources, standardized data models, assurances of required service qualities (e.g., regarding latencies or availability of services) to multi-user authorization models [3]. This dissertation proposes a meta-architecture that enables the interoperability of different smart grid components on an end-to-end basis. This meta-architecture includes the definition of appropriate middleware services and the integration of suitable protocols. Thus, a protocol stack is created, which may be integrated into local devices' middleware solutions, and does not require centralized middleware frameworks.

### 1.1 Current Situation and Motivation

The upcoming smart grid combines power systems with a distributed Information and Communication Technology (ICT) infrastructure [4]. This enables distributed control algorithms, which again allow the power grid to include so called Distributed Energy Resources (DER), thus enabling prosumer participation and a rise of the share of renewable energy sources [5]. As many different stakeholders are interacting with such an ICT infrastructure, interoperability is a very crucial requirement [4]. In order to allow for a more generic approach of co-operation, it has to be clarified first, where additional standardization efforts are required [1].

However, standardization in the energy domain is mostly performed for very specific applications, as shown in [6]. Generic interoperability solutions should yet allow for collaboration of diverse subsystems, which has to be independent of the invoking applications. In other domains, some centralized frameworks (e.g., [7]) may offer a similar functionality; yet in these cases, the collaboration is then limited to applications using the same frameworks. The goal here is to broaden this approach in a way, that collaboration of different frameworks gets possible, provided they incorporate some relatively simple services, as will be defined by the approach at hand.

### 1.1.1 Problem Statement

The heterogeneity of applications and of the underlying infrastructure shall be granted in order to allow a widespread use of technologies provided by a variety of vendors. However, interoperability of solutions from different vendors shall easily be possible. In many approaches to similar problems in other domains such as Home Automation (HA) or Automotive (e.g., [8], or [9]), the solution is seen in a kind of “X-Architecture”, consisting of 3 tiers (compare Figure 1.2):

- The Application Tier holds the whole variety of smart grid related applications.
- The Middleware Tier provides a common ground for these applications.
- The Infrastructure Tier holds the functionality of accessing the infrastructure.

At Application Tier, business functions may pursue individual objectives – as long as the nature of the exchanged data is known, these functions may benefit from each other, even if they do not follow a common goal. At Infrastructure Tier, different technologies may be used to exchange data with field devices and data networks – as long as data is delivered under pre-defined requirements (e.g., real-time constraints), it will be useful for the respective recipients. Thus, the Middleware Tier is the one, which provides convergence of the different systems. A similar approach is detailed in Section 1.3.3 and in publications 1 and 3 of this thesis (see Sections 2.1 and 2.3).

From a deployment point of view, a middleware solution should run on each virtual or real device separately, i.e., on top of a real hardware or hypervisor and the respective machine’s Operating System (OS). Thus, hardware respectively hypervisor and OS constraints have to be taken into account. Applications, on the other hand, have to access hardware or hypervisor and OS resources *only* via the middleware, such that additional functionality as Role-based Access Control (RBAC) can be enforced. Bypassing this convergence tier would constitute security risks [10], as functions like rights management could be eluded.

From a systems engineering point of view, a common convergence tier would allow a better separation of concerns. Traditionally, systems engineers would have to design application functionalities, resource management, rights management, OS functionalities, and hardware properties in a common effort. By using middleware solutions, engineers can concentrate on the application, letting the middleware provide the rest [11].

The middleware however then has to utilize underlying infrastructures, which could be changing over time or lack the power to support the envisaged functionality. The middleware thus has to keep track of applications' requirements and to inform them, if the infrastructure is not capable of providing these resources. This has to be done according to respective Service Level Agreements (SLAs) [12]. Virtualization technologies like Software Defined Networking (SDN) [13] may provide means to ease the handling of resources by the definition of an intermediate abstraction layer.

The aim of this doctoral thesis is now to adopt existing middleware approaches (based on X-Architecture models) to the energy domain, and to provide such a flexible, performant, safe, and secure convergence tier for ICT infrastructures in smart grid environments (see Section 1.3.3). Hereby, it is not intended to provide yet another middleware framework containing all these functionalities, but rather to provide a definition of appropriate services, which could later on be integrated into existing frameworks. For the purpose of the thesis however, these services are realized and tested in some existing research environments ("demonstrators"), which allow an assessment of the specified functionalities with reasonable effort.

### 1.1.2 Related Work

In the smart grid domain, several proposals for organizing ICT contributions [14] have been made. Usually, these approaches follow a hierarchical abstraction of ICT functionalities, providing layered meta-architectures. Some existing frameworks provide middleware functionalities in the smart grid ecosystem or in related domains, using these or other abstraction schemes. Middleware technologies and architectures have to be addressed as well. For the communication between distributed entities, a number of useful grid-related protocols are widely used. Finally, some infrastructural aspects (containing hop-by-hop protocols [15]), which are influencing the middleware layer, have to be taken into account. All these aspects shall be revisited in this short survey.

#### Meta-models and Interoperability Layers

As already mentioned, interoperability of different smart grid subsystems (most likely produced by different vendors) has several distinguishable aspects, ranging from physical connections and plugs up to different business models of several stakeholders. As

these aspects are related to different abstractions levels, a natural approach to handle corresponding interoperability issues is the definition of layered (hierarchical) abstraction schemes. One of the first approaches, which got widely accepted in the energy domain, is the GridWise Interoperability Context-Setting Framework [16] from the GridWise Architecture Council (GWAC).

The hierarchies are a little less granular in the Smart Grid Architecture Model (SGAM) [17], a meta-architecture model defined by the European Union as an outcome of the M/490 [18] mandate. Nevertheless, this model is very comprehensive and reflecting the smart grid ecosystem to a big extent. The reason for that is its 3-dimensional nature, which does not only represent a hierarchical view on interoperability, but also includes zones and domains, which cover the dimensions of the automation pyramid as well as the power system value-chain from bulk generation to customer premises.

Finally, relations to a very mature layered system, the OSI reference model [2] can be discussed. This model reflects only the lower interoperability layers of the GWAC or SGAM approach, but similarities in the layer definitions are obvious. However, the OSI reference model realizes interoperability rather with concrete protocols on each layer that with abstract interoperability definitions. Also, each layer provides its functionality as a service to the next higher layer via Service Access Points (SAPs).

Table 1.1 shows a comparison of these layered approaches, and gives also a reference to the tiers of the mentioned X-Architecture model. The bold entries refer to the Middleware Tier, which is in the focus of the work at hand. As this relates to OSI layers 4 (“L4”) to 7 (“L7”), the work at hand in its essence aims at providing an “L7-SAP”, which is a synonym for a middleware Application Programming Interface (API). That API provides its services to the smart grid applications, which are represented with “Layer 8” (an informal term containing application specific issues). The latter can be associated to the business context specific Application Tier (which can be considered already as pragmatic in terms of the GWAC model).

Table 1.1: Comparison of Layer Systems

X-Tiers	SGAM (partly)	GWAC (partly)	OSI
Application	Function	Business Context	“8”
<b>Middleware</b>	<b>Information</b>	<b>Semantic Understanding</b>	<b>6,7</b>
<b>Middleware</b>	<b>Communication</b>	<b>Syntactic Interoperability</b>	<b>4,5</b>
Infrastructure	Communication	Network Interoperability	2,3
Infrastructure	Component	Basic Connectivity	1

## Middleware Technologies and Frameworks

For the purpose of setting up middleware services, there are a couple of useful technologies available. However, none of them fulfill all the requirements for a comprehensive Middleware Tier listed in publications 1 and 2 (see Sections 2.1 and 2.2), such that combinations of these technologies and respective protocol stacks have to be envisaged to realize a comprehensive solution. The basic issues to look at are the semantics and the syntax of data to share, store, or process in smart grid environments.

Interoperability may be realized in a loosely coupled (co-operation of stakeholders with independent goals and activities) or strongly coupled (collaboration of stakeholders with common goals and activities) manner; however, communication (the exchange of data) is the key issue for both forms. Communication has a static aspect (syntax: how is data structured; semantics: what meaning do the parts have) and a dynamic part (syntax: which course of actions is needed when data items are exchanged between communicating partners regardless the content of the payload; semantics: how does the content of the data influence this course of actions).

In the following, technologies for both aspects are considered. Many of them are standardized, often from International Electrotechnical Commission (IEC) or Institute of Electrical and Electronics Engineers (IEEE). First of all, a unique data model for smart grid data objects is highly desirable. The standard IEC 61970 [19] provides a Common Information Model (CIM) for energy systems, which defines the structure and the content of energy data items to be used in Intelligent Energy Devices (IEDs). The CIM is already there for a couple of years [20], but seems to receive more esteem in recent years. Modelling the payload of data exchange activities with CIM is thus recommended (e.g., the IEC 61850 standard suite [21] allows for compatibility with CIM data formats). The Device Language Message Specification (DLMS)/Companion Specification for Energy Metering (COSEM) suite [22] also combines protocol and data definition aspects.

Presentation and session layer technologies are required to represent the data models defined by CIM, to be able to store data objects in data bases and to serialize and deserialize complex data objects for lower layer protocols. For that purpose, Open Process Control Unified Architecture (OPC UA) [23], seems perfectly suitable [20]. OPC UA provides a Service Oriented Architecture (SOA), which allows IEDs to communicate over predefined services. Yet, individual courses of actions for complex application scenarios are not supported. Basically, request/response patterns are available; however, publish/subscribe patterns are gaining importance [24].

Extensible Messaging and Presence Protocol (XMPP) [25] and Data Distribution Service (DDS) [26] constitute alternatives to OPC UA. A lightweight solution dedicated for session handling only would be Session Initiation Protocol (SIP) [27]. For using structured data, Extensible Markup Language (XML) [28], JavaScript Object Nota-

tion (JSON) [29], and Yet Another Markup Language (YAML) [30] are popular tools. The definition of data structures could be performed by XML Schema Definition (XSD) [28]. Some engineering tools, such as the IEC 61499 [31] based 4DIAC [32], are also using Abstract Syntax Notation One (ASN.1) [33] for their communication interfaces. All of these solutions share the main shortcoming of OPC UA, i.e., the lack of individual interaction patterns.

In general, middleware frameworks often support data exchange over OPC UA, XMPP, or ASN.1. Many of them also support the SOA paradigm, where applications are able to access functionalities over an API providing a number of well-known and addressable services. As sketched by [34], many of these frameworks rely on Open Services Gateway initiative (OSGi) [35], which allows to encapsulate service functionalities as “bundles”, which can be deployed, started, and ended continuously. Some of these frameworks (e.g., OpenHAB [8]) are only loosely bound to the energy domain, as their origin is rather in the field of HA (like OSGi), but are nevertheless worth to be considered, as they often incorporate Customer Energy Management Systems (CEMS) functionalities.

### Grid-related Protocols and Communication Infrastructure

In the smart grid ecosystem, individual interactions are often provided by applications instead of a common infrastructure. For that purpose, applications have to keep track state information regarding data exchange issues, they have to know addresses of communication partners, etc. However, at least for some concrete scenarios, there are open application layer protocols, which can be used by end systems and applications from different origin. The DLMS/COSEM suite [22] is useful for smart metering and in Advanced Metering Infrastructures (AMIs) [36]. Open Automated Demand Response (OpenADR) [37] defines Demand/Response (DR) activities for DERs. From its idea, Open Smart Grid Protocol (OSGP) [38] has been defined more generic, but also is used mainly in home environments. It specifies also the lower layers, using Control Network Protocol (CNP) [39] (formally known as Local Operating Network (LON)) for that purpose. IEC 60870 [40] and IEC 61850 [21] have gained acceptance in the transmission grid infrastructure and the substation automation, respectively. Many of these protocols, at least those which are routable, rely on the classical Internet protocol stack with Transmission Control Protocol (TCP) [41] and Internet Protocol (IP) [42]. However, IPv6 [43] is an interesting alternative for its presence in Internet of Things (IoT) environments, especially in combination with header compression technologies like 6LoWPAN [44].

Also for transport protocols, some alternatives should be considered, like the unreliable, but efficient and stateless User Datagram Protocol (UDP) [45], or the UDP based Quick UDP Internet Connections (QUIC) [46], which might overcome TCP’s latency issues

while still providing reliable transmission. Multipath TCP [47] could also be used for that purpose. Communication infrastructures often use virtualization technologies to abstract the physical infrastructure from middleware services and applications, which is called Network Function Virtualization (NFV) and basically provides network functions “as a Service”. SDN seems to be the most popular of these technologies, especially in the field of smart grids [48]. However, SDN goes beyond the pure abstraction of network services and allows forwarding rules to be based on domain specific logics.

In routed networks, Software Defined Wide Area Networking (SD-WAN) [49] provides a wide-area overlay to the public underlay networks as the Internet, outperforming the classical Multi-protocol Label Switching (MPLS) [50] approaches. Nevertheless, there is still a need to configure and manage these networks, which is usually done by single vendors. Here, vendor-independent decentralized solutions would be beneficial (which would then be an approach very similar to that provided in this thesis). An alternative with even better configuration potentials, but less maturity and market adoption is given with Programming Protocol-independent Packet Processors (P4). An overview over such technologies is given in [51].

In the Internet, real-time capability is of increasing importance, as has been announced under the keyword “Tactile Internet”, often in combination with 5G as last mile technology [52] (assuming sufficient performance in the backbone). Also in Local Area Networks (LANs), the use of real-time technologies is increasing, not only in smart grid environments. Typical examples of such technologies use synchronized scheduling, such as Time Sensitive Networking (TSN) [53] or Ethernet Powerlink [54]. Finally, virtualization is also done in access networks, especially in 5G networks. Often, this is referred to as “Network Slicing” [55]. Unlike SDN, this is a form of NFV which does not offer higher layer logics as influence parameters for forwarding decisions.

### Shortcomings of Existing Approaches

The state of the art review has shown many relevant and interesting technologies for setting up appropriate ICT solutions in the field of smart grids; yet, it has also revealed shortcomings and open issues:

- Existing frameworks lack interoperability to other frameworks.
- Application layer protocols are not generic, but application-specific.
- The handling of state information is committed to applications.
- The interdependence to the infrastructure is not considered sufficiently.
- Quality aspects (dependability, performance) are not considered sufficiently.

Dependability hereby can be expressed in terms of reliability, availability, safety, security, etc., whereas performance relates to the keeping of timing constraints, especially in (hard) real-time environments, as well as scalability issues. The given shortcomings now pave the way to the research goals of the thesis at hand in a natural way.

## 1.2 Research Goals

The thesis shall explore, to which extent the named potentially useful technologies are applicable as a part of an intended integrative solution. This solution however shall try to handle the named shortcomings in a sufficient way, i.e., to fulfill functional and non-functional requirements defined in publications 1 and 2 (see Sections 2.1 and 2.2). Thus, the conducted research shall reveal answers to the following overall research question:

*How can a solution for a convergence layer in a smart grid environment be set up, which provides the required middleware functionalities in a sufficient quality, while allowing for re-use of existing technologies wherever possible?*

### 1.2.1 Detailed Research Questions

This overall research question can now be broken down into a couple of concrete sub-questions:

#### *Research Question RQ01: Coupling*

How strict shall the coupling provided by the middleware be? Loose coupling allows for more genericity and thus easier integration of new components and a better human understanding; however, strict coupling has advantages regarding performance.

#### *Research Question RQ02: Trust*

How can trust between different partners at remote sites be ensured without involving a central trust authority? Can block-chain based mechanisms be utilized as a potential approach for that?

#### *Research Question RQ03: Timing Issues*

How can security and reliability of information exchange be provided, while still keeping timing constraints? Authentication and encryption are costly, as are Automatic Repeat ReQuest (ARQ) mechanisms. Especially when utilizing connection-oriented transport protocols, connection handling at transport *and* application layers may endanger real-time capability.



*Research Question RQ04: State Handling*

How shall state information be handled? Stateless communication is easier and faster to provide; however, as long as most considered applications are control applications, the keeping of state information at both ends of the communication (controlled and controlling device) is inevitable.

*Research Question RQ05: Collaboration*

How can collaboration between different users and user groups be organized? Can priorities sort out potential interferences between users, and how can race conditions be eliminated?

*Research Question RQ06: Rights Management*

How can users' rights be guaranteed? How can Policy Decision Points (PDPs) and Policy Enforcement Points (PEPs) be realized in a central way, and which degree of details (e.g., which data granularity) has to be processed to answer users' queries?

*Research Question RQ07: Addressing*

Which generic addressing scheme can be used to address resources present in the smart grid ICT infrastructure? How does this scheme relate to Medium Access Control (MAC) addresses, IP addresses, Unified Resource Identifiers (URIs), etc.?

*Research Question RQ08: Overlay Networks*

How shall overlay networks (application layer routing) be handled? The simplest way is via virtual connections (e.g., treating a TCP connection as it was a direct link) between each pair of end nodes; however, as this does not scale, other architectures have to be considered (e.g., message brokers).

*Research Question RQ09: Scaling*

How do the considered technologies scale? When adding numerous end nodes, the number of potential communication acts increases exponentially. Mechanisms to deal with that have to be defined.

*Research Question RQ10: Virtualization*

How can virtualization technologies be utilized to satisfy the requirements on the underlying infrastructure? The middleware must be able to define the requirements on the ICT infrastructure in terms of SLAs. A virtualized infrastructure might help to guarantee the service levels, as it is able to shift virtualized network functions to other resources.

These research questions define the scope of the research for the given topic, as it was also given in Table 1.1. Several additional clarifications should be made in order to describe the environment, where the given topic shall integrate into.

### 1.2.2 Scope of Work

First, the “Business Context Layer” according to GWAC [16] is not in focus, as it relates to application specific use of semantic information in business related processes. This is not seen as functionality for a generic Middleware Tier, but rather for a business specific Application Tier. This tier is sometimes considered as part of the SGAM Information Layer (e.g., in the M/490 Framework Document [56]); but in principle, the context is organization-specific – and thus pragmatic rather than semantic. Consequently, in literature it is also associated with the SGAM Function Layer (e.g., [57]).

OSI L6 and L7 (see Table 1.1) cover data structures and their interpretation, hereby ranging from simple big or little endian data item representation until defining whole object models and their respective data points. Here, interoperability refers to a common notion of interpreting these data portions, i.e., a common semantic understanding. However, semantic interoperability can be handled in various ways:

- A pre-defined semantics allows for efficient communication (mostly in combination with binary payload), but lacks genericity. The protocol includes the semantics (which makes it a strongly coupled system), and does therefore not need for a language, that describes the object model. The protocol stack not only defines the data structure (L6) and the data objects and interactions (L7), but also the interpretation of the data within potential applications. It is also possible to leave the definition of data objects (using the structure elements from L6) to application-specific “profiles” (e.g., SunSpec [58] would provide such a definition to Modbus/TCP for exchange of inverter data). By using such profiles, the semantics are still pre-defined, but the structure elements can be kept more generic, which makes it more flexible.
- A self-descriptive semantics (e.g., using XML) allows for defining the meaning of the exchanged data portions in the same way as the data itself is exchanged, i.e., using tag/value pairs. However, the tags have to be known in advance (e.g., using XML-Schema); they thus form a domain-specific language. This is usually based on Unicode Transformation Format (UTF)-8, but not on binary data. Such a language can be used for describing the object model at L6. At L7, the objects themselves and their interactions can be defined. The meaning of the data items however is still dependent on the interpretation by the application using the respective protocol stack.

- A compromise between these two approaches may be used by pre-defining all semantics in the upper layers of the OSI system. This includes the structure, how real objects are represented as data objects (e.g., using ASN.1). The interpretation of data objects however is still left on the applications utilizing the protocol stack. The “pre-definition border” is located in L7 here. Depending on the heterogeneity of applications, concrete object definitions and available interactions may be defined in L7 or at the applications accessing the L7 protocol.

L5 covers functionalities like billing, for which it must be known who accesses which data in which granularity. This is seen as metadata to the actual payload, and thus it holds syntactic information. Nevertheless, session issues play an important role for middleware solutions, as rights management and billing are typically bound on sessions: Authentication, Authorization, and Accounting (AAA).

Protocols as Hypertext Markup Language (HTML) or Simple Object Access Protocol (SOAP) are considered transport protocols here (although widely seen as application layer protocols). For instance, HTML serves as application layer protocol only in the context of being used by web browsers containing an Human-Machine Interface (HMI); for the use within web services for machine communication, it plays another role, i.e., providing end-to-end transport. Hence, we have to consider syntactic issues of HTML, but the semantics are associated with object models transported within the HTML payload. Syntactic interoperability, however, is also a major issue for the planned research, as is depicted in Table 1.1.

Finally, only end-to-end protocols are considered here; the way in which packets are routed within the ICT infrastructure is not in focus, as this should be application-neutral. However, this does not hold for any kind of overlay network. For such overlay networks, semantic information could be used for routing between nodes of this overlay network (which are considered endpoints for the underlying ICT infrastructure). This semantic-aware routing has indeed to be considered for the work at hand, as the overlay network can be interpreted as a service to the calling applications [59], thus providing a Network as a Service (NaaS) approach (relevant for the middleware API).

### 1.2.3 Expected Results

The desired results should allow for novel scientific findings and thus providing answers to the formulated research questions. In concrete, these findings can comprise empirical quantitative measurements (based on proof-of-concept realizations of cyber-physical systems including simulated or emulated parts), theoretically founded qualitative statements, as well as specifications of relevant parts of the intended solution. In concrete, results should comprise:

- **Specification Items:** First, a meta-architecture based on the X-Architecture model has to be given. This may include a global function bus connecting virtual or real IEDs. Then, an algorithm has to be defined to allow for time-efficient data delivery while keeping availability over a specified limit. Third, an application layer protocol has to be specified for allowing grid control applications to access delivery services in a defined manner, or generic addressing schemes. Finally, a generic addressing scheme has to be provided in order to be able to identify IEDs in cloud and edge computing environments.
- **Validation Items:** Proof of concept implementations shall be given in order to validate the named approaches and to evaluate selected quality measures, especially timing results. This includes an implementation of the function bus and its API, the provision of security means, the provision of remote accessible RBAC functions, the implementation of overlay architectures as SDN [60], as well as control applications and their respective system access (consumption of the API provided by the middleware stack).

Finally, it is expected that answers to the research questions can be derived from the gained results. In the evaluation section (see Section 1.4.3), the gained results are compared to these expectations.

## 1.3 Methodological Approach

In order to answer the defined research questions, several demonstrators had to be set up. Appropriate use cases have been defined to elicit the requirements of the demonstrators. After realizing the respective testbeds, several test suites have been conducted according to the afore defined use cases. Finally, in the evaluation a comparative analysis revealed the envisaged answers.

### 1.3.1 Work Packages and Phases

Concretely, the following work packages have been performed consecutively, thus constituting methodological steps:

1. **Research:** Comprehensive state-of-the-art research, including academic research as well as experiences from technology vendors,
2. **Modelling:** Refinement of basic X-Architecture model approach: Definition of middleware services, middleware API, RBAC model, and data model,

3. **Design:** Definition of functionalities for practical demonstrators, including middleware services and API functions,
4. **Realization:** Implementation of the defined middleware services and API functions in cooperation with other research projects,
5. **Validation:** Definition and execution of scenarios to be conducted with the practical demonstrators in order to answer the mentioned research questions,
6. **Evaluation:** Analysis of the results of the conducted test scenarios regarding the required answers.

### 1.3.2 Research Design

The research design results directly from the stepwise approach: Considering the shortcomings of existing solutions, a common X-Architecture model for the energy domain had been developed, as described in Section 1.3.3. On that base, three demonstrators had been created that cover the functional logic of the envisioned middleware convergence layer as far as possible. With the help of these demonstrators, answers to the research questions had been obtained by defining, executing and then evaluating appropriate test scenarios.

The approach at hand is not to provide yet another middleware framework which is again incompatible with others, but to provide means to extend interoperability for existing middleware solutions in form of proposed standards and protocols. These means shall allow frameworks to provide appropriate APIs to smart grid applications (especially to control applications), which may then use the services provided by that APIs for the engineering of suitable applications [61].

Thus, the dissertation provides an architectural approach for a convergence layer that can be easily mounted into various frameworks. As mentioned, this approach has been evaluated utilizing the Proof of Concept (PoC) implementations of the three demonstrators in order to determine in which way the functionalities of the convergence layer achieve the intended goals with regard to the interoperability of different subsystems – or, in other words, to provide answers to the research questions (which are in fact related to these goals).

Figure 1.1 shows the research design in a simple graphical scheme. After a common state-of-the-art research (see Section 1.1.2) and X-Architecture model definition (see Section 1.3.3), the work is split into the mentioned three demonstrators. As the intended research was designed to be performed in a cumulative way, each of the demonstrators is described in an own journal paper (see Sections 2.1 to 2.3), covering test scenarios, concrete realizations of the X-Architecture model, and a description of the respective

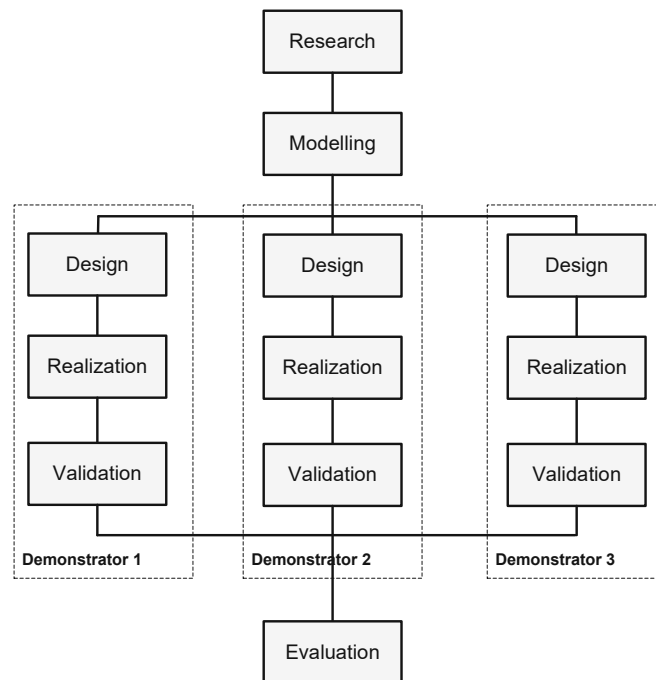


Figure 1.1: Research Design

validation results. Finally, a wrap-up is provided (see Section 1.4.3), including answers to the research questions by performing a comparative analysis of the conducted test scenarios and the obtained results.

### 1.3.3 X-Architecture Model

The basic foundation of the three demonstrators is the abstract X-Architecture model, as depicted in Figure 1.2. Here, each IED is split into the named tiers, where the Infrastructure Tier provides access to sensors and actuators to interact with physical processes (and thus to generate cyber-physical systems), but also network equipment, processor power or any other kind of resources. These resources may be virtualized and accessible as a service, called Infrastructure as a Service (IaaS). This applies not only for devices; with SDN, the whole network infrastructure is abstracted to a Hardware Abstraction Layer (HAL).

It is important to understand, that this is still part of the Infrastructure Tier, and thus SDN's *northbound* interface relates to the infrastructure API. SDN's "applications" provide logics which are performed in the Middleware Tier then. With that, technologies like "application layer routing" can be performed here without the necessity to directly

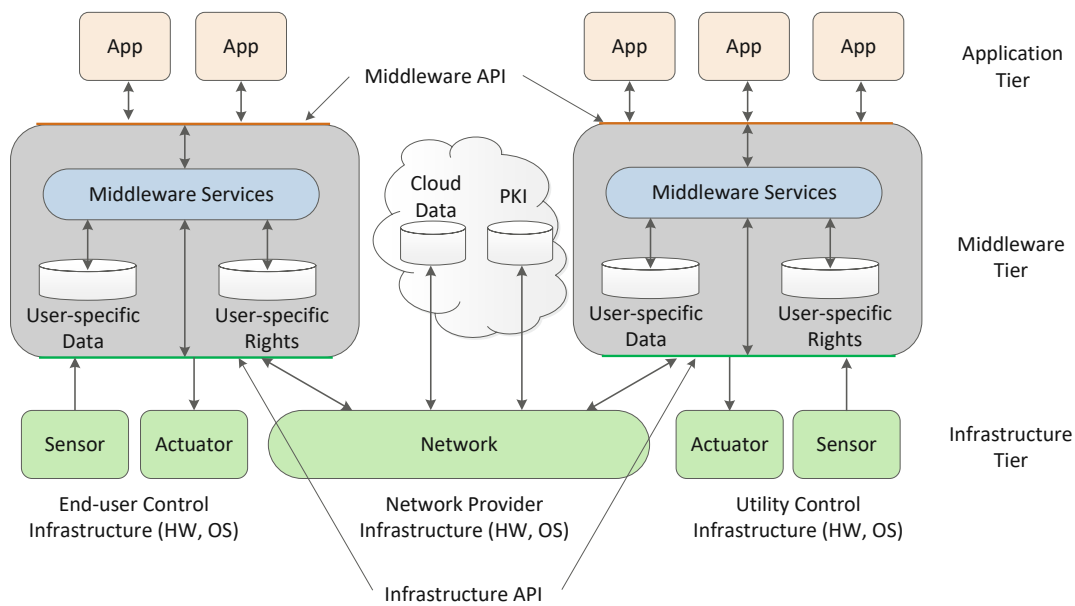


Figure 1.2: X-Architecture Model (see also Section 2.3)

involve control applications. This way, the Middleware API can for instance hide the details of the routing decisions from the actual applications, which can concentrate on their actual functionality disregarding any “middleware issues” such as routing, service orchestration, security or RBAC.

The convergence in the Middleware Tier, which gives the X-Architecture its name, allows to handle these named issues on a per-device level, whereas multiple resources may be utilized, and multiple applications (which may indeed also be distributed) are accessing the middleware services on a device. However, user data and rights management issues are often not limited to one device. Here, access to distributed rights management technologies or to location independent user data (e.g., Public Key Infrastructures (PKIs) and cloud data) must be granted also. Then, IEDs have to double check the access rights, as a local check has to be performed at any IEDs (which may be aligned with the cloud based system). Access to these cloud resources can yet be granted via the infrastructure API only.

### 1.3.4 Requirements Analysis Process

After clarifying the basic methodology and research design for the dissertation, the question had to be solved how concrete demonstrators can be used to derive answers to the mentioned research questions. To this end, it was necessary to determine which

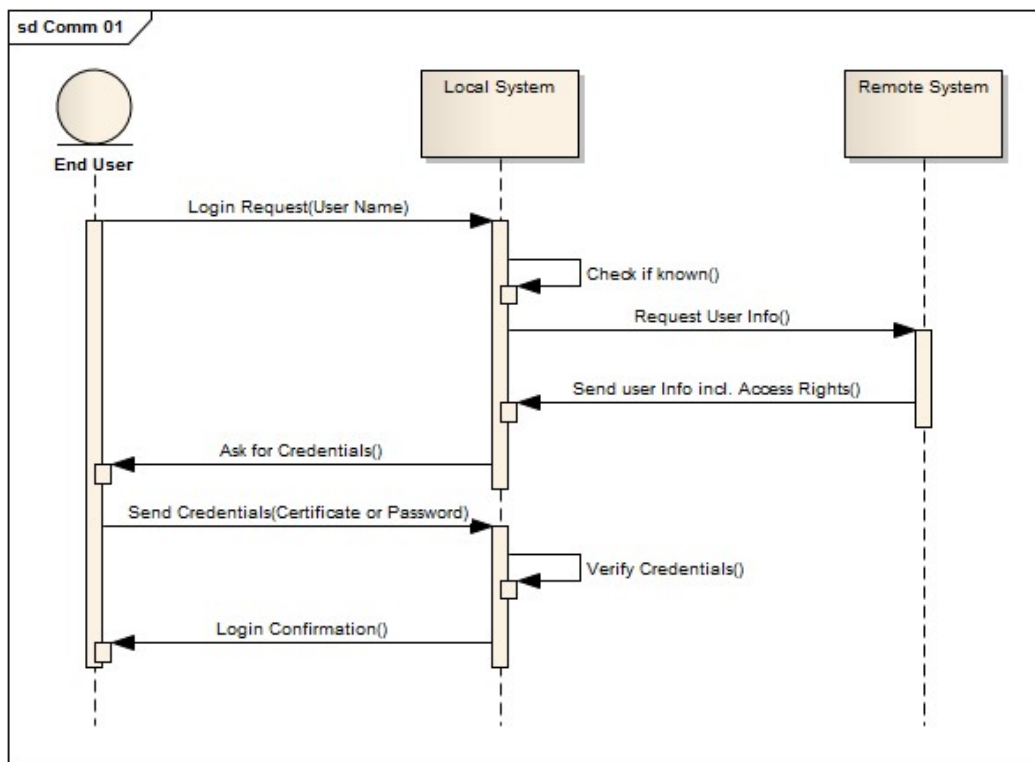


Figure 1.3: Sequence Diagram of Authentication Use Case (UC11)

requirements are given for which demonstrator, respectively for each test scenario which should be conducted with the respective demonstrator.

As usual, the functional requirements have been elicited during requirements engineering by defining suitable use cases. The process of eliciting requirements from the associated use cases has been formalized using the so-called IntelliGrid methodology [62], a systematic method for the elicitation of the use cases, which has been specified especially for the smart grid domain and has been widely used in recent years. The IntelliGrid methodology does not only specify the procedure, but it also provides templates for the elicitation of use cases.

The formalization of these use cases has been done using the formal description language Unified Modeling Language (UML) [63]. UML provides various graphical representations for this purpose, such as use case diagrams or sequence diagrams. A crucial concept here was the definition of actors, i.e., human beings, institutional bodies, or “neighboring” technical systems that interact with the system or subsystem under consideration in a defined way. As an example, a sequence diagram of the authentication use case (UC11, see Section 1.3.5) is given in Figure 1.3.



The interactions may be annotated in order to provide a clearer description. The more formalized these interface descriptions are, the clearer the specifications for the later implementations can be derived. In some existing use case repositories, these formalized representations are mixed with additional information in purely textual form. Of course, this may lead to some vagueness in the definition; however, for human understanding it may be inevitable to add descriptive parts in the specification of use cases. Below, such annotations for the explored use cases are given.

Non-functional requirements constitute a further problem in requirements elicitation. These requirements comprise e.g. quality requirements for the artifact (such as performance or safety requirements), but also economic constraints (such as prices or compatibility with legacy systems), and finally compliance with regulatory conditions (such as legal constraints or the keeping of standards). For the demonstrators of this thesis however, these issues are less in focus, with exception of security, confidentiality, and performance aspects (especially with respect to real-time requirements).

However, the mentioned non-functional requirements can be elicited with the given use cases as well, such that no separate methodological approach has become necessary for this. With specifying the use cases for the demonstrators at hand, the requirements are thus implicitly formulated. In order to validate the selected approaches, Section 1.4.3 explains how the defined use cases could be used to address the mentioned research questions.

### 1.3.5 Use Cases for Demonstrators

In the following, the results of the requirements analysis process for the mentioned three demonstrators is given.

#### Use Cases of Demonstrator 1

In demonstrator 1, a whole series of use cases has been defined; for the thesis, the use cases related to the communication subsystem have been considered relevant. These requirements serve to validate the functionality of the distributed X-Architecture approach as such, yet still without concrete applications.

##### *Use Case UC11: Authentication*

End users who wish to access resources must authenticate themselves, such that it can be determined whether or not they have appropriate access rights to the inquired resource. This can be accomplished by requesting credentials (such as username/password, smart-card/pin, or biometric measurements) prior to any further action. The information provided is then compared to a list of valid credentials that is stored centrally and

cached locally. If the provided credentials are found in the list of valid credentials, the corresponding user interface is determined and displayed. Details of the exchanged information are visualized in Figure 1.3.

*Use Case UC12: Authorized access*

Any request from an authenticated entity to a service or resource must first be validated using a permission store to prevent unauthorized entities from gaining access to resources whereas authenticated and authorized entities can access the resources they need. Only after the role-based permission store has confirmed the validity of the request, the service processes the request and responds accordingly. Additionally, the permission store may or may not allow the allocation of certain resources to requesting entities; e.g., high-resolution data is delivered only to Distribution System Operators (DSOs), whereas all other requests (if they are authorized at all) are responded to only with overview data.

*Use Case UC13: Security*

To enable access to resources with special confidentiality requirements, the transmitted data must be encrypted to prevent eavesdropping or manipulation by unauthorized actors. This applies in particular to all actions in which physical devices are directly or indirectly controlled. Also all actions for the management of devices and/or rights management databases must be secured to prevent damage to the integrity of the target system. This can be achieved by implementing current cryptographic protocols such as Transport Layer Security (TLS). Encryption of the data itself must also be considered (as required, for example, in the Smart Meter Gateway Protection Profile of the German Federal Office for Information Security [64]). Suitable methods for key exchange must be provided; therefore, a combination of symmetric and asymmetric cryptography should be considered (e.g., PKIs).

*Use Case UC14: Device maintenance*

Each resource (an endpoint device, an active network component, an appliance, a back-end system, a controller, a database, a function, a service, etc.) can be accessed by a defined administrator account that has all the operational access rights required for this purpose. The administrator account can be assigned to one or more natural persons. This allows to apply settings to the resources that differ from the provider's default settings in order to meet requirements for the respective resource, e.g., to optimize energy consumption or to set higher confidentiality levels. In addition, all necessary updates (especially security-related updates) have to be applied to the resource via the administrator account.

*Use Case UC15: Rights management*

Each resource can be accessed from a variety of different requesters. For this reason, a suitable RBAC model must be defined such that all accesses can be classified accordingly. This is the responsibility of a defined owner (this is the role that has all rights to a resource and is responsible for its management), who must then delegate all other access rights. In order to be able to technically configure the resources accordingly, an administrator account (i.e., an account with all the necessary operational rights) is provided by the owner. However, the right to add and remove accounts and to configure their access rights remains solely with the owner.

**Use Cases of Demonstrator 2**

While demonstrator 1 focuses on infrastructural use cases (for the theses, concentrating on the communication subsystem, yet still being very generic), demonstrator 2 is based on concrete case studies from the smart grid. Hereby, the required flexibility of the middleware was provided by virtualization technologies.

*Use Case UC21: Virtualized redundancy*

Virtualized redundancy aims at building a communication environment, which is robust against hardware failures, by providing a redundant, fail-safe ICT subsystem, the Virtualized Communication Infrastructure (VCI). This means, that in the event of a failure within the communication subsystem, other parts of the VCI can take over full functionality. Virtualization with SDN can be used to create a redundancy solution that is application-independent and thus largely protocol-independent. However, attention must be paid to compliance with real-time conditions – especially for processes, which cannot act locally, but contain centralized components. For instance, forwarding rule updates are slower at an SDN controller compared to local “fast failover” mechanisms at the participating SDN switches.

*Use Case UC22: Commissioning*

Commissioning deals with the integration of IEDs into the process network. In this context, process network means the (usually TCP/IP-based) Operating Technology (OT) communication network of a utility. Commissioning involves identifying the IEDs that apply for integration into the process network, assigning appropriate roles, defining the access rights associated with those roles (see use case UC15), the policy enforcement (i.e., enforcing policies to ensure legal access while blocking all unauthorized requests, see PEP), providing the required service availability, as well as maintaining the process network infrastructure.

*Use Case UC23: Grid-based routing*

Grid-based routing is intended to forward measurement data to different receivers depending on the current electrical situation in a power grid; for instance, only those data sinks should receive the measurements which need the data as input for control decisions. A concrete scenario with several data sources (e.g., voltage measuring points) and several data sinks (e.g., transformer controls), which are distributed over several substations, was considered. Electrically, these data sources and sinks are interconnected via the power network topology. Depending on the current configuration of the power network topology (positions of the switches contained in the network), the data sources can be assigned to different data sinks; accordingly, different routes must then be set for data transmission in the communication network.

*Use Case UC24: Anomaly detection*

Anomaly detection deals with the detection of power system anomalies resulting from human activities (malicious or negligent) or from equipment failures. Also, measurement anomalies that may be the result of compromised software, firmware or application components on the field devices are considered. These power system anomalies shall be identified by analyzing available information from the communication network. Hereby, the analysis of the data traffic is performed based on the flow statistics in SDN switches. The SDN controller collects statistical information from all switches and performs the subsequent analysis as a controller application. In this way, typical attack or error vectors shall be detected, but also deviations of the respective flow from stored well-known traffic profiles. This means that not only intentional attacks can be detected, but also defective and incorrectly configured components.

**Use Cases of Demonstrator 3**

The use cases in demonstrator 3 have been used to test the extent to which a suitable message-based middleware solution with a distributed message queue can implement the requirements of both DSOs and households. Thereby, the focus has been set on flexibility requests from a DSO to several participating consumer or prosumer households. By doing so, it was possible to validate the approach with realistic scenarios and control applications.

*Use Case UC31: Self-consumption optimization*

Hereby, the connected Low Voltage (LV) grid is in equilibrium. Accordingly, no DSO requests are transmitted to individual households. Thus, the specific subscriber systems do not have to adapt their operation to a higher authority and can operate their appliances according to their own needs. The focus here is on self-consumption opti-

mization. This is achieved through intelligent load management, but also through the use of appropriate energy storage systems. A community storage shall also be considered, since load balancing within a larger community offers more possibilities than in separate individual households (due to the lower simultaneity factor).

*Use Case UC32: Low-priority control*

In low-priority control, the price is used to influence the behavior of the participating households. This is done when the LV grid is in a small power imbalance; this imbalance is sent as an input to the participating households. Based on this input, balance has to be created by reducing or increasing the electricity purchased from the participating DSO in order to increase the stability in the LV grid.

*Use Case UC33: High-priority control*

In the case of high-priority control, the DSO attempts to counteract a stronger imbalance by means of direct technical control, which is again sent as an input to the respective load management of the participating households. In this case, still, it is up to the local control algorithm to what extent the DSO's requests are realized. In this way, losses of comfort can widely be avoided. However, certain reserves (which have been defined in advance per SLA, e.g., the possible temperature band for heating) must be held available for this case. These reserves may only be used for strong control – the load management is yet still free to ignore the requests, i.e., not or only partially react on the requested flexibilities to prevent comfort losses.

### 1.3.6 Realization of Demonstrators

With specifying the use cases for the three demonstrators, the requirements on these demonstrators have been implicitly set, as the demonstrators have to be able to conduct respective test runs. The concrete test scenarios (i.e., the parameterization of input data) hereby have been defined in a way to allow for receiving answers to the specified research questions, as sketched in Section 1.4. First, the concrete architectures of the demonstrators had to be specified. These are all derived from the generic X-architecture (see Section 1.3.3), but differ a little in their realization. In particular, distinctions must be made according to the type and number of components (applications, subsystems, used infrastructures) and their interfaces (system calls, protocol stacks).

#### Architecture of Demonstrator 1

The system architecture of demonstrator 1 includes an electrical network, whose components (i.e., IEDs) are connected by a communication network, as pointed out in

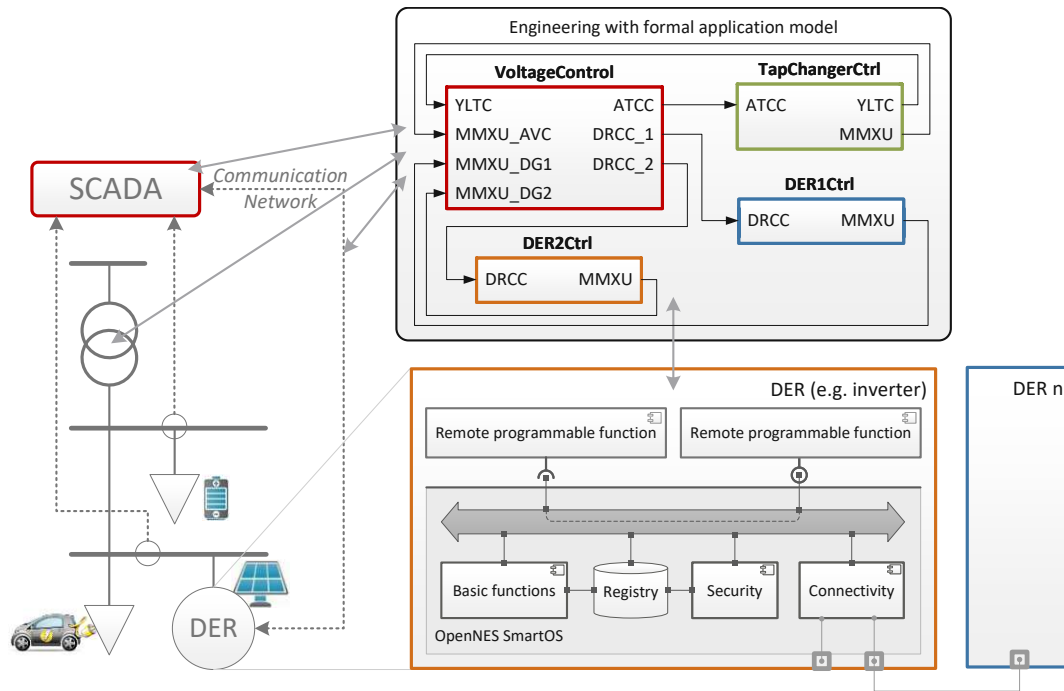


Figure 1.4: System Architecture of Demonstrator 1, adopted from [5]

Figure 1.4. This communication network enables distributed applications, such as the control of DER. For the IEDs themselves, a middleware was developed, which was brought to execution in all participating IEDs.

This middleware is referred to under the name “SmartOS” in the following. The name may be somewhat misleading; in fact, the SmartOS is a middleware, as it requires an underlying local operating system on the respective IED. However, the SmartOS provides all the additional functionalities needed, such as RBAC, sandboxing, etc. Thus, it can well be identified with the generic X-Architecture’s Middleware Tier.

The core of the SmartOS is the so called Virtual Functional Bus (VFB), which takes over the function of an Enterprise Service Bus (ESB) within the component. It provides a corresponding API to the Application Tier, via which the applications can communicate to the system, but also to other applications. The applications themselves run in sandboxes provided by the SmartOS, and can only interact with their environment via the defined API. This applies both to communication within an IED as well as to external communication. For the latter, the SmartOS provides appropriate drivers (e.g., for Ethernet, for serial interfaces, and some more).

Each communication process is checked by the security subsystem for the corresponding authorizations. For legacy protocol stacks, the SmartOS architecture may require the use of adapters. The SmartOS may also include some basic domain-specific functions,

e.g., common inverter logics for controlling DERs. Inverters are usually controlled by local voltages, such as in Q(U) control. This functionality is encapsulated via a base component and thus under full control of the SmartOS.

The last point to mention here is the network architecture. The individual IEDs here are all realized as Virtual Machines (VMs), which are connected in different local networks via a common IED router. This is connected to the operator network via an optional path simulator (which can for instance simulate latencies or packet losses), but also to an internal administration network, as well as to the public Internet.

The hypervisor that hosts the VMs is also publicly accessible on the Internet. This is a very flexible solution for testing purposes; however, for commercial solutions other realizations may be preferred. Also, a Virtual Private Network (VPN) server is accessible in the administration network, which establishes the connections to the IEDs in the respective private networks. The network management and the credential store for managing access authorizations are also located in the same administration network.

### Architecture of Demonstrator 2

In demonstrator 2, this basic architecture was extended to include the aspect of virtualization of system components. Virtualization was primarily applied to those parts of the infrastructure, which require the greatest possible flexibility; this applies in particular to the communications infrastructure, as this subsystem must respond to the rapidly advancing digitalization (especially in the lower grid levels), and the respective increase of the number of system components.

In terms of technology, the demonstrator mainly bases on SDN, in order to be able to combine the necessary flexibility in the configuration of new components with appropriate reconfigurability in case of a component failure [51]. Additionally, SDN also offers guaranties for Quality of Service (QoS) provision as well as strict separation of individual traffic streams to support data protection and security.

A rough overview of the system architecture of the demonstrator is shown in Figure 1.5. The typical separation of control plane and data plane in SDN is clearly visible, complemented by the application plane for the integration of specific application logics that originate from the use cases under consideration. SDN provides a rule-based forwarding of frames through SDN switches or SDN routers. The rules are defined by the SDN controller in the control plane and passed on to the devices in the data plane; these check the rules from the highest to the lowest priority, and if the corresponding prerequisites are met, the frames are forwarded according to this rule.

Using the “Northbound Interface”, various applications can trigger the creation of rules in the control plane. Thus, this basic architecture can be adapted very easily to the requirements of different scenarios or use cases. However, this only allows to virtualize

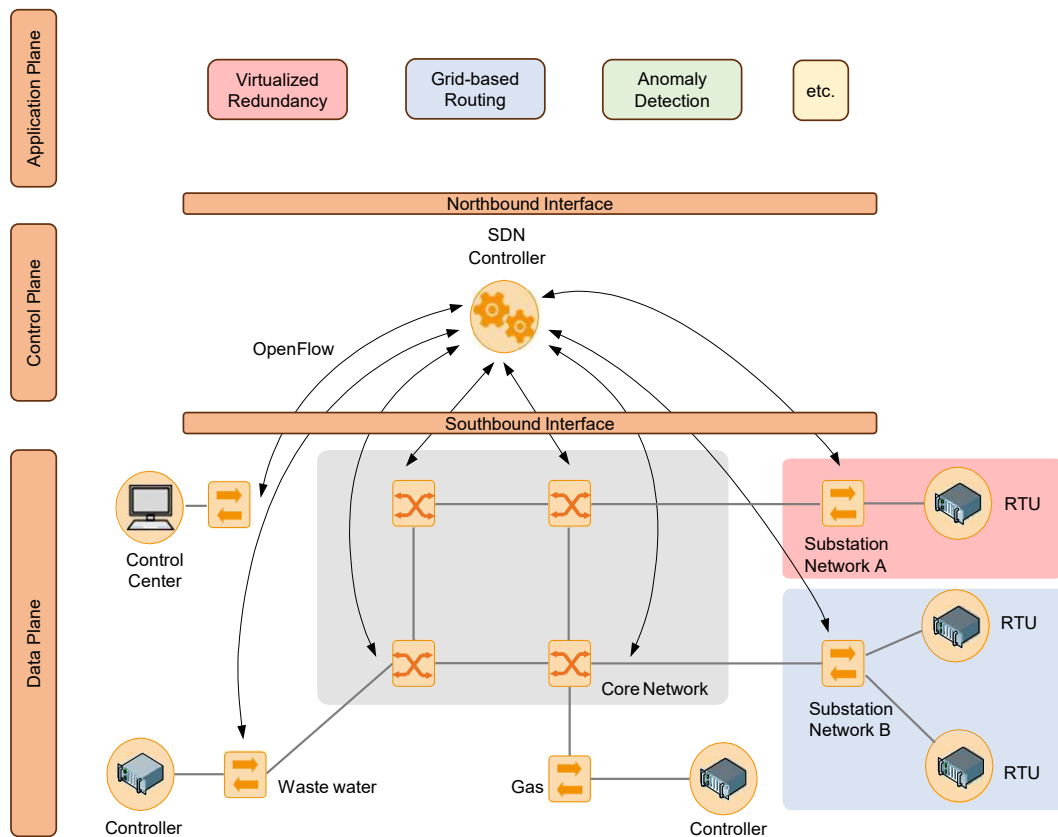


Figure 1.5: System Architecture of Demonstrator 2, adopted from [65]

the network devices and connections; the IEDs themselves, such as Remote Terminal Units (RTUs) here, remain in hardware. Schematically, it can also be seen that the communications infrastructure, together with its virtualized parts, is connected to the physical infrastructure of the power grid. This can be quite different for different concrete use cases.

In demonstrator 2, not only one PoC implementation was carried out, but several in parallel, since the individual use cases are based on different implementations. These differ primarily in terms of the physical components, but also the individual SDN switches, routers and controllers were assembled and configured differently. The system architecture shown in Figure 1.5 is thus still a meta-architecture (as is the generic X-architecture). However, the relation to the generic X-Architecture can easily be depicted. The Northbound Interface of the SDN controller basically provides the same kind of API as the infrastructure API in the X-Architecture; various middleware services can access the communication infrastructure via this interface in order to use it for communication operations.



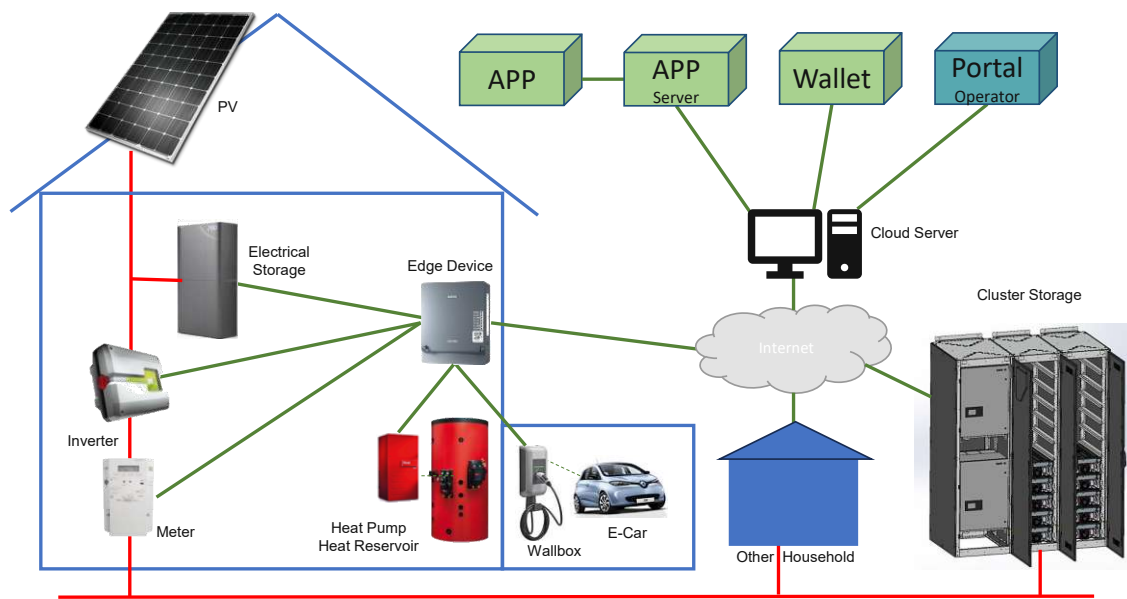


Figure 1.6: System Architecture of Demonstrator 3

### Architecture of Demonstrator 3

The focus in demonstrator 3 was set on concrete applications from the power domain, adapting the generic X-Architecture model to application specific requirement while keeping the principle idea. The testbed should be used with the applications self-consumption optimization and flexibility management in energy communities. For this purpose, a “cluster operator” was defined that could send flexibility requests to the energy community. The latter then had to be able to respond to the requests by corresponding savings or additional consumption. In addition, a neutral state was defined that the energy community could use for self-consumption optimization. In addition, also a cluster storage was integrated for the mentioned purposes.

Controllable loads were installed in the test households that participated in the energy community, which could be operated differently depending on individual requirements. These loads primarily included charging points for Electric Vehicles (EVs) as well as heat pumps as the households’ heating systems. The advantage of these appliances is, that they can be shifted in time easily, and in some cases they also can be operated at different power values. Furthermore, electrical storage units were installed in the individual households, as these can be operated both as an energy source and as an energy sink, and can thus cause a time delay in consumption or feed-in. Finally, a Photovoltaic (PV) system was also integrated as a volatile renewable energy source. The main system components are shown schematically in Figure 1.6.

All these system components are connected both electrically and informationally. The electrical connection is symbolized here by the red line (for simplicity, this has been only sketched within the household). Data connectivity is represented by the green lines. In addition to the households, which are connected via “IoT Edge Devices”, there is also a backend that contains the cluster operator (which is indeed simulated), the evaluation units (which calculate the incentives for the responses to flexibility requests), and an app server (which provides user relevant data for applications on mobile devices).

The communication infrastructure in this project was deliberately kept simple to ensure scalability and real-time capability, and also to provide an easy integration of new system components. Internal communication within a household is done using the protocols that the components natively support; in most cases, these are proprietary Modbus connections. Externally, only the lightweight Message Queuing Telemetry Transport (MQTT) protocol is used.

With regard to the generic X-Architecture, it is noticeable here that the external communication only takes place via a messaging protocol, i.e. a loose coupling is perfectly realized here. The edge devices, but also the backend server, represent the individual IEDs here, which only communicate via the MQTT broker and thus do not require any further protocol adapters. However, all management services usually offered by middleware solutions used in IEDs are restricted here to the native capabilities of the MQTT broker; this is due to the demonstrator’s focus on the application logic.

### 1.3.7 Security by Design and Rights Management

As a critical infrastructure, security is an essential feature for the smart grid [66]. Although measures to increase security are not the focus of this thesis, fundamental security aspects had to be considered from the very beginning. In concrete terms, this means that, although no research was carried out in the direction of new security architectures, the existing state-of-the-art had to be taken into account in an appropriate manner. In order to implement this, measures to increase the standard protection goals of Confidentiality, Integrity, Availability (CIA) [66] had to be considered.

Accordingly, the design had to be performed in a way that these measures could be easily integrated, an approach known as “Security by Design”. For the design of the present demonstrators, the approaches described below have been taken into account [67] in order to be able to ensure the stated protection goals in sufficient quality. As the most important measure, RBAC was introduced. With RBAC, each access request is examined to determine whether the accessing actor in its currently active role has the corresponding access rights or not; in the negative case, access is blocked. Here, only technical subsystems are eligible as actors; human actors can only access system resources through such a subsystem, e.g., a Graphical User Interface (GUI).

Authentication is evidently a precondition for RBAC. Thus, unauthorized reading (which would compromise confidentiality), but also unauthorized writing (which would compromise integrity) can be avoided. For this purpose, a registry was defined that contains all information on users, groups, roles, and their respective permissions. This applies to all access requests to resources of the infrastructure under consideration, i.e. to IEDs as well as to network resources, sensors, actuators, etc.

Furthermore, it must also be ensured that no access can actually take place when no access permission exists. This can be achieved by generating “closed ecosystems”; i.e., by restricting any access to subsystems, which have applied for that purpose and received the permission by an operator. Such, these subsystems get signed and are subsequently well-known to the system. However, this means that applications and human users have no direct access options (and thus no unauthorized access is possible). For applications, a so called “sandbox” has been defined for this purpose, which can only communicate via selected and permitted interfaces.

In addition, dedicated (not publicly accessible) communication networks should be used wherever they are available, since this makes physical access (eavesdropping) more difficult. The communications infrastructure should also be equipped with firewalls and Intrusion Detection Systems (IDSs) to also make Denial of Service (DoS) attacks more difficult and to ensure the high availability of the system. Finally, all communications should be encrypted. This not only prevents eavesdropping on exchanged messages, but also “man-in-the-middle” attacks. Similarly, messages should also be stored in encrypted form such that stealing physical storage does not create a security flaw. Key management can be performed by a PKI.

## 1.4 Results of Work

After having defined the use cases for the three demonstrators, and setting up according testbeds to validate the respective functionalities, concrete test scenarios had been defined, which then could be executed in these respective environments. Besides those validation items, also some specification items could be derived, which are described afterwards. A comparative analysis (evaluation) concludes this section.

### 1.4.1 Test Execution and Validation

For the test execution, the concrete input data and framing conditions for all single test runs as well as comprehensive test suites have to be specified. The validation here is the proof, that the tested functions are running as expected and can thus be used to analyze the research questions.

### Test Scenarios for Demonstrator 1

In the context of the thesis at hand, only those test scenarios are mentioned here, that relate to the development of the basic infrastructure, being the first realization of the X-Architecture (see Section 2.1 for more details). Test scenarios for the application logic of demonstrator 1 were not considered here, since the properties of the architecture itself had to be checked first in the infrastructural use cases. However, the test runs of demonstrator 3 (see below) also concern operational use cases and can thus prove the practical suitability of the concept in a real-world environment.

In order to be able to test the relevant components, a series of individual tests with increasing complexity was defined, in order to integrate the involved components into the tests one after the other. This made it possible to keep the added complexity per test run within acceptable limits. Also, the evaluation of the tests was easier to manage, as potential flaws could be located in the newly added components. At the same time, it was possible to ensure that all required features could be systematically tested and validated. The following individual functional tests have been conducted with demonstrator 1:

1. Two software components communicate locally: This minimal test showed the basic functionality of the VFB, as well as the “Security” and “Registry” subsystems, as depicted in Figure 1.4.
2. Two software components on different IEDs communicate via TCP/IP: This extends the previous test run by the integration of the “Connectivity” subsystem and its subordinate TCP/IP adapter. For the “Open Application Layer Protocol (OpenALP)” (the application layer protocol defined in the work at hand), a variety of communication patterns is available, such as “Push” or “Request/Response”, as depicted in Section 1.4.2.
3. Two software components on different IEDs communicate via ASN.1: This test run showed the transparency and interchangeability of the underlying protocol. This message based communication uses ASN.1 for serialization and deserialization and can for instance be used to send commands and set values to actuators in a “cyber/physical system” under control.
4. Two software components on different IEDs communicate via secure TCP/IP: This test run extended the scope of the previous test runs in a way that encryption and authentication were integrated via a central “Trust Center” (however, only a dummy encryption has been tested).

5. A software component communicates with a legacy device via Modbus/TCP: This test run finally demonstrated the support for existing legacy environments on the example of Modbus.

In all described test cases, the condition for passing the test was, that a message could be exchanged according to the chosen communication pattern. As a result of passing all these tests it can be stated, that the middleware providing communication services for the defined patterns, and using the defined address scheme, provides the desired functionality which is needed for the realization and evaluation of the use cases of demonstrator 1.

### Test Scenarios for Demonstrator 2

For demonstrator 2, actually four different testbeds have been realized, corresponding to the four defined use cases. In the context of the work at hand, the “Grid Based Routing” use case was of particular interest, since it bridges the gap to the operational use cases, though still being classified as infrastructural use case (see Section 2.2 for more details). Hereby, a virtualized infrastructure has been used as a HAL. The forwarding of data in this underlying infrastructure has been performed depending on the state of the application. In the concrete example, the application was the control of the Medium Voltage (MV) grid in the Gurktal valley in Carinthia, as sketched in Figure 1.7.

The application scenario was first simulated in different laboratory setups; subsequently, also a field test has been conducted (yet, without really controlling the IEDs). To determine the specific test runs, it was first necessary to clarify the input and output parameters (sensor respectively actuator parts of the testbed). For this purpose, the components of the testbed have to be examined individually:

- The components labeled  $T_x$  represent so called On Load Tap Changers (OLTCs). They are adjusting their transformation ratio depending on the distribution of the local voltages in the lower order grid (i.e., the MV grid, here).
- The components denoted by  $U_x$  represent voltages which are collected by corresponding volt meters.
- The components labeled  $P_x$  represent power lines; these lines are either functional or out of order.
- Finally, the components marked with  $A_x$  represent switches; these are either open or closed.

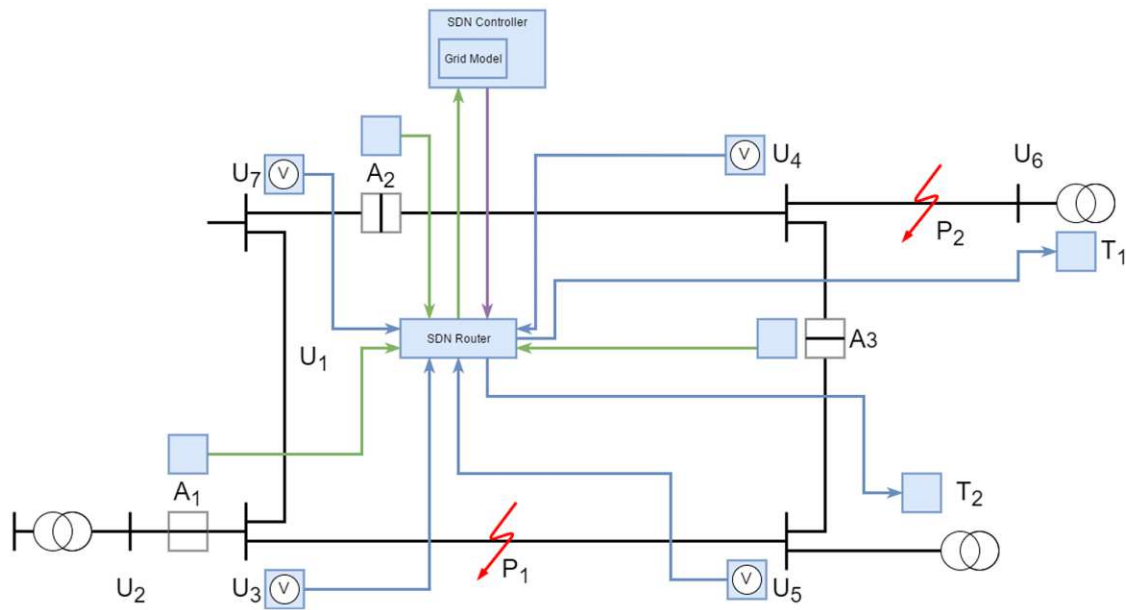


Figure 1.7: MV Testbed in the Carinthian Gurktal, adopted from [68]

Now, the goal of the tests was not to evaluate the control applications itself (i.e., calculating the correct ratio of the OLTCs dependent on the voltage measurements, and setting the switches depending on outage detections), but to verify that the measured data from the meters was available at the right transformers at the required time. The question is therefore, whether under a given environmental situation (given by switch positions and possible line disturbances) the measured data (local voltage values) are sent to the transformers in timely manner, such that the local control strategy which is implemented at the transformers can take place. To examine the behavior, a couple of functional tests with escalating complexity have been conducted with demonstrator 2 in a simulated test environment. Additionally, the first test has also been conducted in a real-world testbed.

1. In the initial variant, there were no disturbances and the switch positions were given as in Figure 1.7. The transformer  $T_2$  needed the voltages measurements from volt meter  $U_5$ ,  $U_3$ ,  $U_1$ , and  $U_7$ . The other meters did not matter for the control of  $T_2$ , because  $U_2$  was too far away (it was closer to another transformer), and  $U_4$  as well as  $U_6$  were not connected, since the corresponding switches  $A_2$  and  $A_3$  were open. For transformer  $T_1$ , on the other hand, only  $U_4$  and  $U_6$  were of interest, since the rest of the network was not accessible due to the switch positions. The SDN router was able to deliver the measurements as foreseen, since its rule set represented an implicit topology knowledge.

2. In the topology change variant, switch  $A_1$  was opened and switches  $A_2$  and  $A_3$  were closed. Accordingly, the relevant measurement data for the controllable transformers changed. For instance, if switch  $A_3$  is closed, the measured values of  $U_5$  also become interesting for the control of  $T_1$ ; vice versa,  $U_4$  and  $U_6$  become relevant for the control of  $T_2$ . In consequence, the SDN infrastructure had to be able to deliver these control-relevant data accordingly. Hereby, the switch changes had been made in the power simulation, and the SDN controller got informed about that. It calculated a new rule set and pushed it to the SDN router. After that, the forwarding took place again as foreseen.
3. In the line disturbance variant, it was assumed that a line break occurred on lines  $P_1$  and  $P_2$ . Depending on the switch position, entire network sections could be de-energized. To prevent this, a central Supervisory Control and Data Acquisition (SCADA) system had to be able to detect the interruptions and switch the breakers in the simulation such that the feeders could be supplied with power again. The SDN controller got again informed about that change, and in consequence, the SDN infrastructure was able to again adjust the forwarding of the measured data accordingly.

In all described test cases, the condition for passing the test was, that the respective voltage measurement values have been forwarded to those transformers which required them (and only those). In the latter variants, it was therefore necessary to investigate whether the infrastructure was capable of gaining awareness of the situation of the power network in acceptable time to ensure the forwarding in timely manner. As a result of passing all these tests it can be stated, that a virtualization of the communication infrastructure can be used to incorporate domain specific knowledge (here, about the grid topology) in forwarding decisions. The distributed control applications, however, can apply their control logics without regarding these infrastructural and middleware aspects. For a testbed of this size, a simple flooding might still have worked. However, this approach is expected to scale quite poorly, especially when considering LV grids.

### Test Scenarios for Demonstrator 3

Demonstrator 3 finally deals with operational use cases, i.e., applications which use middleware services for their own purposes. The goal of the validation here was to demonstrate the practical value of the approach by choosing realistic scenarios for these applications (see Section 2.3 for more details). Hereby, the functionality and scalability of an environment built according to the X-Architecture scheme was tested in the control of several consumer and prosumer households, taking into account stability requirements of the associated DSO. As mentioned, these stability requests have

been simulated (yet under consideration of plausible values), whereas the rest of the demonstrator used real world measurements and real commands to actuators.

In the following, the concrete test runs are described, broken down to the individual appliances. The considered appliances include heat pumps, wallboxes, and electric storage units. For the tests, also different environmental parameters had to be taken into account (e.g., seasonal influences, since thermal factors also play a role for room heating). Again, a couple of functional tests with increasing complexity has been performed to validate the demonstrator, covering all appliances. In this case, however, each functional test consisted of several single test runs in order to cover different environmental conditions for the appliances, wherever necessary:

1. For the basic tests, all five defined states of the LV grid (“Flex Up”, “Net Price High”, “Standby”, “Net Price Low”, and “Flex Down”, see Section 2.3) have been tested. The tests included the reaction of the controller to each of these states regarding all appliances of a household. As different environmental situations had to be taken into account, this included following conditions: outside temperature low (winter conditions), outside temperature high (summer conditions); EV plugged with immediate charging request, EV plugged but no special charging request, EV not plugged, EV got plugged during an existing state; State of Charge (SoC) of the power storage  $> 80\%$ , between  $80\%$  and  $20\%$ ,  $< 20\%$ .
2. Subsequently, tests with a sequence of states have been conducted. These tests lasted for several days, with 96 states per day, such that the total number of states was some hundred states per test. The sequence of states hereby has been defined in such a way, that it follows the energy consumption of a typical household in a weekday (H0 profile); for each day, the same sequence has been used. Also here, different conditions (outside temperature, cloudiness) have been taken into account.
3. Finally, a long-term test (running for more than one month) had been conducted, with an arbitrary sequence of states. The sequence has been generated with a randomizer in order to cover all possible state transitions (in reality, e.g. a direct change from Flex Up to Flex Down would be very rare, yet not impossible). This kind of test in fact exceeds pure functional testing, as this test includes test coverage aspects. The long-term test has been conducted only once (in fall 2021), as for that long period different conditions could be expected. However, the highest and lowest outside temperatures have thus been omitted here.

Basically, a distinction was made here between tests to validate the logic in controlling the individual appliances, and longer-term test suites to evaluate the effects of the



applied logic on the households. The former tests served to validate the system environment (does the gathering of sensor values and the sending of control commands work), the latter to validate the control logic (can the requested flexibilities be provided, and does the control also work for the households themselves in the desired way). Additionally to these tests, which used real components (except, as mentioned, for the DSO's flexibility requests), also a number of simulations have been conducted. With these, some special conditions could be examined (e.g., an EV is charged primarily during days to maximize self-consumption when a PV is present), as sketched in [69].

In all described test cases, the condition for passing the test was, that the sensor values could be read in time, the control commands could be calculated according to the defined policies and sent to the appliances, and that the appliances react as foreseen on these commands. As a result of passing all these tests it can be stated, that a message queue based middleware, as it has been utilized in this demonstrator with XMPP, is capable of providing control commands in a close-to-reality scenario in timely manner and with sufficient availability. Also, adding another household did not impair the operation of the system, indicating for a satisfactory scaling behavior.

### 1.4.2 Specification Items

First of all, the generic X-Architecture model has proved its usefulness in a number of use cases, to be run in different concrete testbeds. The architectures of three demonstrators have been described in Section 1.3.6, all of them being realizations of the abstract model, which has thus served as a “blueprint” for the demonstrator architectures. The main components as well as the main middleware services, i.e., the middleware API, the communication patterns, the data access, as well as the RBAC components, can be identified in any case. Modifying the original approach, also cloud components have been allowed in the realizations of the X-Architecture model (though requiring respective access policies) to allow e.g. for central data storage or the usage of PKIs. Details of the used meta-model are given in Section 1.3.3.

This is necessary, as for traceability purposes, generated data often has to be persisted. Especially in demonstrator 3, this was done in a large scale in order to be able to document the results of various application scenarios. Many of these data represent time series; accordingly, time series databases such as InfluxDB were also used for persisting. In order to also be able to address the individual data points of the data objects used, an open address scheme called OpenAddress was developed, formulated in Extended Backus-Naur Form (EBNF) [70], as described in Section 2.1. In addition to the generic addressing, virtualization of infrastructure elements wherever possible (as shown in demonstrator 2), provides further abstraction means and allows e.g. network functions to be used as services (NFV).

Additionally, a concept for an open application layer protocol has been developed, providing some elementary communication patterns, such as “Push 1:1”, “Push 1:N”, “Request/Response”, and “Publish/Subscribe”. These patterns can be made available to applications as middleware services, as shown in demonstrator 1. Individual courses of actions can be compiled from that elementary patterns on demand of an application, either by the application itself, or by the middleware, which then provides composite services (dedicated to specific application domains). Hereby, the handling of state is possible to both the middleware and the application. As a design principle, it is recommended that applications handle domain specific states, whereas the middleware should handle all states which are associated with some more or less generic services (e.g., authentication of a single user or user group).

Finally, some algorithms had to be developed throughout the work on this thesis for providing requested functionalities. This is especially true for those parts of the work, which deal with the logics of control applications, e.g., for the control of household appliances in demonstrator 3 (the respective algorithm is listed in Section 2.3). However, also the deduction of concrete forwarding rules dependent on information about the grid, as it has been done in demonstrator 2, is an algorithmic contribution of this work. Lastly, the implicit logics of existing protocols have been utilized where appropriate.

### 1.4.3 Analysis of Research Questions

This section now clarifies the extent, to which the requirements defined by the given use cases of the three demonstrators are able to provide answers to the research questions outlined in Section 1.2.1. The left column in Table 1.2 provides a mapping of the defined research questions to the mentioned use cases, which shows that the use cases address issues which are appropriate to investigate the research questions. Furthermore, an analysis of the results from the execution of respective tests (which have been described in Section 1.4.1) is given. Thus, in the right column, the derived answers to the research questions are sketched.

Table 1.2: Answers to Research Questions

Mapping to Use Cases	Assessment of Results
<b>Research Question RQ01: Coupling</b>	
The use cases UC32 and in particular UC33 are intended to demonstrate the extent to which control for low-voltage grids can be supported on the basis of message-based middleware. As	By successfully validating this infrastructure, it could be demonstrated that loose coupling in the smart grid for control purposes is possible (see Section 2.3). It should be noted, however,

<p>this middleware is loosely coupled, these use cases deal with the applicability of such approaches in control environments.</p>	<p>that due to very tight time constraints, this does not address the area of protection technology (which is, however, provided locally anyway). The OpenAddress format used for this purpose is comparatively compact and thus well suited for control and regulation tasks, but still allows semantic operations on the addresses specified with it, such as a semantic search (provided that the data object models are known). The middleware API and message-based communication (e.g., with MQTT or XMPP) generate a SOA. Depending on used data object models, additional annotations (tags, textual descriptions) are possible, which can be used by humans to deepen their understanding.</p>
<p><b>Research Question RQ02: Trust</b></p>	
<p>In order to establish a trustworthy infrastructure, a corresponding distributed rights management was defined in UC15, which specifies the respective access rights for all involved stakeholders as well as their respective roles. The required authentication was defined in UC11. This also allows the management of credentials to be validated.</p>	<p>Each access to any resource is checked for the respective access rights, and based on the result these requests are allowed, restricted or prevented (see Section 2.1). For communication processes, the checks are made on sender and receiver side. Thus, an RBAC model has been set up on a per IED basis. Without the usage of central instances, this can be considered a distributed ledger model (if all users/roles are known to all IEDs). As the rights are still considered a static property, this approach is yet not a blockchain (as it is not transaction based). However, for practical reasons (consensus mechanisms cause much effort), central approaches have also been enabled. Thus established technologies as PKIs for key management and Lightweight Directory Access Protocol (LDAP) for user management can still be utilized.</p>
<p><b>Research Question RQ03: Timing Issues</b></p>	
<p>The effects of security means on real-time capability can be well examined in the operational use cases, especially in use cases UC32 and UC33. Additionally, the infrastructural use cases UC21 and UC23 deliver useful timing information in the case of topology changes (e.g., due to line failures).</p>	<p>Real-time capability is always a question of defining the tolerable latencies; apart from protection technology, the specifications in the power domain are in the range of at least seconds (e.g., primary control defines the time range up to 30s). In demonstrator 3 (see Section 2.3), 60s update cycles have been utilized.</p>

	<p>With security and privacy means implemented as described above (e.g., encrypted and authenticated message-based communication), these timing limits usually can be kept easily. In addition, virtualization technologies as SDN (see Section 2.2) also offer even higher-performance solutions (e.g., SDN Fast Failover); however, these mechanisms have not been investigated here, since they cannot guarantee the necessary flexibility for operational use (these fast reactions need to be defined in advance). Protective devices of course have much stricter timing requirements (as discussed above); yet they are operating locally and automatic, thus they have not been in focus of this research.</p>
<p><b>Research Question RQ04: State Handling</b></p>	
<p>As can be seen in UC31, the implementation of distributed control logics requires situational awareness of all controller instances involved. Here, the respective controllers at application level must be able to specify their respective state information. The more it is important to provide these instances with as much assistance as possible. For instance, appropriate protocol adapters may be used to enable generic communication, as done in UC12. Virtualization technologies may hide infrastructural aspects from the applications, as in UC21.</p>	<p>The realization of stateless communication is hardly possible, be it only for the extensive requirements on security and privacy. However, for state information, the main goal of a system like proposed in this work is not to totally avoid states (as this would limit the functionalities of applications to an unacceptable extent), but to allow separation of concerns; i.e., application states should not bother middleware or infrastructure entities, and vice versa. In the realization of UC31 and UC12 it could be shown, that no state information is exchanged over the OpenALP API (see Sections 2.1 and 2.3). All domain relevant states are handled with the applications; all security related data can be handled in the middleware layer and do not affect the applications. If needed, applications may implement own courses of actions dependent on application states. When using stateful protocols such as IEC 61850, this can be done at the protocol level. Again, it has then to be distinguished between application layer protocols (containing domain know-how, e.g., controller states) and transport protocols (containing infrastructural and middleware data, e.g. protocol adapter states). In demonstrator 2, infrastructure states can be hidden to applications by the use of SDN (see Section 2.2).</p>

<b>Research Question RQ05: Collaboration</b>	
<p>A basic requirement for collaboration between different stakeholders is an RBAC model. As mentioned above, the rights management is already well covered by UC15. In the operational use cases, for example in UC32 or UC33, this is examined using typical example scenarios.</p>	<p>The realization of an RBAC model, as examined in demonstrator 1, organizes the users' rights to access resources and services, and thus serves as a PDP (see Section 2.1). The model specifies access rights of users (potentially with different roles), groups, and resources. Applications are handled as resources, which also have access rights. They might be triggered by different users with different access rights; users can access the system only via access applications (HMIs). However, this access model only defines the rights of the involved stakeholders, but not the semantics of the cooperation. Questions as "Which application needs access to which data?" or "Who has the right to control or regulate a system and when?" relate to application specific logics and can only be clarified in the applications themselves. Yet, in the considered operational use cases in demonstrator 3, attention has been paid on typical problems of control systems, such as oscillation behavior and race conditions (see Section 2.3). Oscillations and/or race conditions might for instance result from different stakeholders influencing the system's behavior; in the validation, no such effects have been observed, as the trigger (the flexibility request from the associated DSO) only changes each 15min.</p>
<b>Research Question RQ06: Rights Management</b>	
<p>The implementation of the described access rights, i.e., ensuring that there are no further interactions with the system apart from the authorized accesses, is the task of security. Encryption, as in UC13, can be used to efficiently prevent unauthorized reading, while authentication and authorization, as in UC12, can also be used to restrict write access. In addition, anomalies in network traffic can be investigated, as in UC24.</p>	<p>Rights management, security, and privacy issues have been implemented in demonstrator 1 by means of a security module, which is part of the middleware. It uses authentication to identify the requester of a service or resource (the latter indeed being accessible only <i>via</i> a service) and queries the registry (i.e., the PDP) for appropriate access rights; in case of success, the VFB grants the access then (thus, actually the VFB serves as a PEP in this context). The question, which users and applications should receive which data in which granularity (reading access), or should even be allowed to inter-</p>

	<p>vene in a controlling manner (writing access), is to be answered by the respective system, and can not generally be answered. However, the realization of demonstrator 1 allows to allocate respective access rights (see Section 2.1). Thus, the system’s operator has the possibility to decide on a case-by-case basis. For simplicity, the decision has been done in a binary way. For further details (e.g., different data granularities), separate requests have to be issued. Additionally, information about requests and granted access could be collected by an appropriate middleware (e.g., by SDN “applications”). With that information, anomalies and even deliberate attacks could be detected, as has been demonstrated by [71].</p>
<b>Research Question RQ07: Addressing</b>	
<p>The suitability of a generic addressing scheme for the mentioned requirements can be verified in UC12, since the authorized access to a remote resource requires the discovery of that resource. The same is true for resource maintenance, which is specified in UC14.</p>	<p>“OpenAddress” was defined in publication 1 as a generic addressing scheme (see Section 2.1). The objective here was to define a compact scheme (which is therefore suitable for control tasks) that would still allow semantic searches across distributed nodes. OpenAddress provides local and global addresses; both have been used and validated in demonstrator 1. The generic nature of the scheme also allows mappings to other address schemes; hereby, mappings to Modbus and IEC 61850 addresses have been provided; of course, also a mapping to plain IP addresses would be possible (i.e., queries are routable in the Internet, if needed). As an abstract scheme, mappings to MAC addresses or URIs would as well be possible.</p>
<b>Research Question RQ08: Overlay Networks</b>	
<p>A challenge in defining smart grid overlay networks is the clear separation of data streams, as is discussed in UC13. Another topic is routing; for application layer routing, forwarding decisions can be made according to other criteria than topological ones. This includes semantic criteria such as the importance of the data for certain nodes. This is examined in detail in UC23.</p>	<p>For the separation of data streams, several technologies (including cloud, fog, and edge technologies) may be used as an infrastructure abstraction, as outlined in [51]. In demonstrator 2, this is mainly addressed with SDN, which additionally allows for separating the forwarding from the control of the data streams (see Section 2.2). With that, domain specific knowledge can be integrated in forwarding decisions.</p>

	<p>However, the rules which shall embody the domain knowledge may only use L2 and L3 data (i.e., no payload) for the forwarding decision. P4 may overcome these limitations in future. With SD-WAN, wide area overlay networks can be constructed; hereby, location may play a role for efficient services (i.e., data delivery may be dependent on the grid's state). However, as the construction is done over existing TCP/IP based networks, the performance of the underlay can not be guaranteed here. For overlay networks, addressing is also an important issue to consider. The OpenAddress scheme, as already described above, may constitute an easy way to define addresses in overlay networks, without the need for knowing details of the underlay. Finally, for scaling see the results below (research question RQ09).</p>
<b>Research Question RQ09: Scaling</b>	
<p>In the context of the work at hand, the issue of scaling is mainly concerned with the number of IEDs involved in a communication network, the number of connections between these IEDs, and the number of communication operations / data transfers (e.g., for maintenance, as described in UC14, or for picking new nodes, as discussed in UC22).</p>	<p>Scaling is an important challenge to deal with (not only in the context of overlay networks). Many networks provide a full mesh topology of participating end nodes (i.e., IEDs here). However, in the demonstrators at hand, middleware technologies are used, which rely on intermediate devices such as brokers (for message queue based middleware) or SDN devices (also for SD-WAN). In both cases, the topology is changed to an extended star topology which provides linear scaling (regarding the number of nodes, connections, as well as data transfers). Except for SDN controllers (which may be replicated anyway) and PKIs (which may use distributed architectures such as the “Web of Trust”), no central components are used in the demonstrators at hand; thus, they are providing “scalability by design”. In demonstrator 3, the cooperation of distributed nodes via a suitable overlay network (which was based on a message broker infrastructure) was investigated (see Section 2.3). Thus, scalability can be demonstrated in a practical implementation. Even more nodes have been used in the simulations [69], also showing proper scaling behavior.</p>

### Research Question RQ10: Virtualization

Virtualization is expected to provide an important flexibility gain with respect to smart grid ICT infrastructures. The requirements on those can be functionally divided into different parts that roughly correspond to the layers of the OSI reference model [2]. However, routing is considered the most relevant part here, as it influences the performance of data transfers, and can at the same time *be* influenced by middle-ware technologies, as intended with UC21 and UC23.

Various virtualization approaches have been investigated for this purpose within demonstrator 2: Cloud/edge computing, message queues, Virtual Local Area Network (VLAN)/Virtual Extended Local Area Network (VxLAN), SDN, SD-WAN, as well as P4 (see Section 2.2). As the most promising solution, SDN was considered in particular, since P4 still has little market penetration and the other candidates are only locally effective, do not support QoS agreements (SLAs) and/or lack the desired flexibility in cases of rapidly changing requirements. In demonstrator 2, the suitability of SDN was examined specifically with regard to failures of parts of the infrastructure (grid failures in UC23 and ICT failures in UC21). In this context, it was examined to what extent SDN can be used to operate a “fail operational” network, i.e., to enable full maintenance of functionality in the event of a failure, and to operate the whole underlay completely transparently for the applications. In both use cases, SDN is able to provide the necessary flexibility and changed the routing accordingly, within the given time limits (30s). In addition, UC22 provides NFV by applying SDN for functions like intrusion detection.

With these specific answers to the defined research questions, the demonstrators showed their ability to provide the features needed for an integrative realization of the X-Architecture model. However, as the work at hand intended to evaluate interesting features of that approach, rather than to provide an all-in-one solution, a comprehensive implementation is left over to frameworks which have the interest to provide openness and interoperability to a broader community.

## 1.5 Summary of Scientific Publications

This section gives a short summary of the publications used in this thesis (see Sections 2.1 to 2.3 for more details). The respective scientific contributions of these publications are sketched in Section 1.6.



### Publication 1

In publication 1 (see Section 2.1), a concrete realization of the generic X-Architecture model has been described. Hereby, the Middleware Tier has been implemented in form of a so called VFB, which basically constitutes an intra-device service bus within an IED. Applications can interact with this bus only via defined interfaces. For these interfaces, the generic application layer protocol OpenALP has been defined and described in detail. For addressing, also the generic OpenAddress format has been defined and an EBNF description has been provided.

The implementation of the middleware furthermore includes a registry component, which is able to provide user and role data on an LDAP basis, thus working as a PDP. The VFB queries the LDAP data base on request of a communication action, whether or not this is inline with the given fine granular access rights. In case of confirmation, the communication action can be executed; thus, this makes the VFB an appropriate PEP (see Section 1.3.6).

In the validation section, the paper describes a number of tests which have been conducted with that prototype. Five of these tests are related to communication actions in increasing complexity, reaching from intra-device communication until communication with remote legacy devices, where the use of gateways/adapters is required. These tests basically demonstrate the viability of the X-Architecture approach, showing the correct working of the communication actions under the realm of a strict RBAC system.

### Publication 2

In publication 2 (see Section 2.2), the Infrastructure Tier has been complemented with the possibility to add virtualization technologies to suitable parts of the infrastructure. Especially, the communication infrastructure (i.e., the ICT part of the smart grid) was accomplished with a HAL in form of an SDN approach. The SDN network serves as an overlay over the physical infrastructure, which allows for improved possibilities to control the communication flow and to configure the network infrastructure.

In such a scenario, SDN allows “applications” to integrate a higher logic for the control of data flows, which enables the integration of domain specific issues to decide for the forwarding of data items. As the actual smart grid applications should be separated from any infrastructural aspects (this was one of the key design goals of the work at hand), the domain logics have to be represented in an extra layer. The X-Architecture’s Middleware Tier (see Section 1.3.3) is the ideal choice for that.

Publication 2 describes a testbed, where measured states (e.g., on or off states of load switches) of a concrete MV grid (in the Carinthian Gurktal) influence the routing of data packets over the network infrastructure. The idea behind this is, that applications at

distributed IEDs react on sensor measurements and set actuators accordingly. However, sending all measurements to all IEDs is not only inefficient, it could also cause a not affected IED to perform control activities which could interfere with those activities which should regularly take place.

The solution lies in the definition of a Middleware Tier which decides about the forwarding of sensor data only to affected IEDs. For the middleware, an SDN based solution was compared with a simpler message queue solution based on Coaty. The electric grid was firstly simulated with Bifrost and Pandapower; after the tests of the forwarding have turned out successful, also a field test has been conducted. However, the IEDs have not been controlled with that data, but they documented to have received the correct data, thus validating the approach.

### Publication 3

In publication 3 (see Section 2.3), a realistic application scenario has been investigated. Depending on the current situation in a LV grid (which is depicted in five different states, ranging from power shortage to power overflow), a DSO sends flexibility requests to several customers, who are equipped with CEMSs in order to be able to react on these requests. The reactions are rewarded, when they contribute to the stability of the grid; however, the decision is kept locally at customer's side.

The messaging infrastructure of that prototype was based on XMPP, containing brokers at all participating households, which ensures for scalability of the communication subsystem. This is completely in line with the definition of the Middleware Tier (see Section 1.1.1). XMPP also provides means for security and privacy, as authentication and encryption. However, the connection to the single appliances had to be done with respective drivers, e.g., Modbus/TCP. Thus, the local controllers had to be equipped with respective translators (gateway functionalities).

Several tests have been conducted to validate this setup. These tests comprised long-term tests, where e.g. the effects of different weather conditions could be tested (e.g., because of the influence of external temperatures on the attached heat pump). The test site was also equipped with a PV system, generating volatile amounts of power. Also these effects could be studied in the long-term tests.

Besides that, tests for specific situations have been conducted to prove the validity of the control algorithm. However, the flexibility requests from the DSO have been simulated only (yet, considering realistic conditions, e.g., power shortage at noon of cold and cloudy days). In all of the tests, not only the control logics showed the desired effects, but also the communication subsystem demonstrated the performance and flexibility as required.

## 1.6 Scientific Contribution

The research work at hand proposes the usage of an X-Architecture meta-model for integrating multi-vendor systems-of-systems to a seamless solution for smart grid applications, especially regarding distributed and flexible control applications. The concept has been realized and tested in three demonstrators, each of them represented in one journal publication (see Sections 2.1 to 2.3).

1. Demonstrator 1 is concerned with the addressing format OpenAddress, and the generic application layer protocol OpenALP. It shows that these concepts can easily be implemented and deployed, such that it can be integrated into future frameworks without much effort. Its modularized internal setup (using “middleware services”) also allows an integration into micro-service architectures. Thus, the functionality of these middleware services as well as the working of the middleware API could be demonstrated.
2. Demonstrator 2 adds a HAL for the Infrastructure Tier, thus allowing for a virtualization of grid and/or network components. By doing so, standardized interfaces can be used as infrastructure API. In concrete, SDN is used as virtualization technology, such that SDN’s northbound interface can be used for the infrastructure API. Thus, SDN “applications” (here serving as middleware services) can be used to perform forwarding based on logics coming from the power domain (e.g., delivering commands to active actuators only).
3. Demonstrator 3 finally is dedicated to test message based communication, which is the basic ground of all protocol stacks, with real applications from the smart grid domain. In concrete, the control of HA appliances on the basis of a DSO’s flexibility requests has been chosen for the validation scenarios. Furthermore, MQTT is used as message queue, and Modbus as driver technology for the appliances. With the execution of these validation scenarios, the applicability of the approach to real-life environments can be shown.

With these three demonstrators, not only a basic evidence of the functionality can be demonstrated (as a PoC), but also qualitative and quantitative assessments of the approach at hand can be given. These assessments were used to give answers to the mentioned research questions, which are detailed in Section 1.4.3. Summarizing, the thesis at hand examines many existing issues with smart grid middleware solutions (covering functionalities as distributed control, rights management, state handling, addressing, virtualized infrastructures as well as quality properties as coupling, scaling, performance, security, and privacy), and proposes feasible solutions based on existing data representations and protocols.

## 1.7 Conclusions and Further Work

The overall research question of the work at hand addressed the setup of a middleware solution dedicated for the smart grid domain. Hereby, the goal was not to develop a completely new framework, but to re-use existing technologies in a proper way to meet functional and non-functional (quality) requirements of smart grid ICT infrastructures. These technologies have been successfully integrated and tested in different environments (demonstrators), such that an integration of the concepts or parts of it in commercial solutions is possible for the future.

Beyond that, the tests provided some useful answers to a couple of detailed questions on such middleware solutions, showing not only the feasibility of the approach, but giving additional information on best practices, as well as limitations which have to be taken into account. For instance, the mentioned timing issues showed the applicability for distributed control, yet not for protective devices. A summary of these answers can be seen in Section 1.4.3. In comparison with existing approaches, it overcomes the shortcomings of related technologies like OPC UA (which lacks generic communication patterns) as well as XMPP (which is limited to XML based data structures). The openness of the approach with the own address format OpenAddress even allows the integration of non-IP based legacy communication stacks.

Some topics can yet be identified for future work: For instance, an integration into large-scale real-life testbeds should be done to evaluate the validity besides the chosen concrete environments and with different kinds of applications. Hereby, the limits of the approach regarding scalability should be further investigated, especially when it comes to distributed real-time applications. Another interesting aspect for the future might be the consideration not only of runtime aspects, but also of engineering time aspects. First approaches in that direction have already been published [72]. Hereby, the middleware is intended to not only contain runtime services, but also services to be used in collaborative engineering of smart grid applications, thus providing interoperability already by design.

Finally, with the rise of new technologies in the field of artificial intelligence, an integration into existing frameworks as middleware service should be explored. Especially for production and consumption prediction, this could provide major benefits to distributed control applications. Also, developments in networking technologies have to be considered. P4 provides more flexibility compared to SDN; the number of supporting devices is very limited now, but maturity and market penetration are expected to grow. New technologies might emerge in future, which have potential to be used in smart grid environments; with the approach at hand, the flexibility is given to integrate them as services into existing middleware solutions.

# Chapter 2

## Publications

### 2.1 Publication 1

A. Veichtlbauer, O. Langthaler, F. Prössl Andrén, C. Kasberger, T. I. Strasser:  
**Open Information Architecture for Seamless Integration of Renewable Energy Sources.**

*In: Electronics, Vol. 10, No. 4, Article Number 496, Feb. 2021*



#### Own Contribution

The candidate defined the overall paper structure and provided the main parts of the abstract, as well as the introduction and the related work. He also gave the definition and description of the requirements on the prototypical solution, and the main parts of the architectural approach. Furthermore, he provided the interface definition and the description of the OpenALP meta-protocol, and the description of the data syntax. Finally, he contributed to the results evaluation and the conclusion.



Article

# Open Information Architecture for Seamless Integration of Renewable Energy Sources <sup>†</sup>

Armin Veichtlbauer <sup>1,2,\*</sup> , Oliver Langthaler <sup>2</sup>, Filip Prössl Andrén <sup>3</sup>, Christian Kasberger <sup>4</sup>  
and Thomas I. Strasser <sup>3,5,\*</sup> 

<sup>1</sup> Department of Mobility and Energy, Campus Hagenberg, University of Applied Sciences Upper Austria, 4232 Hagenberg, Austria

<sup>2</sup> Center of Secure Energy Informatics, University of Applied Sciences Salzburg, 5412 Salzburg, Austria; oliver.langthaler@fh-salzburg.ac.at

<sup>3</sup> Electric Energy Systems—Center for Energy, AIT Austrian Institute of Technology, 1210 Vienna, Austria; filip.proestl-andren@ait.ac.at

<sup>4</sup> Fronius International GmbH, Solar Energy, 4600 Wels-Thalheim, Austria; kasberger.christian@fronius.com

<sup>5</sup> Institute of Mechanics and Mechatronics, Faculty of Mechanical and Industrial Engineering, TU Wien, 1060 Vienna, Austria

\* Correspondence: armin.veichtlbauer@fh-hagenberg.at (A.V.); thomas.i.strasser@ieee.org (T.I.S.); Tel.: +43-50-804-22825(A.V.); +43-50-550-6279 (T.I.S.)

<sup>†</sup> Andrén, F.P.; Strasser, T.; Langthaler, O.; Veichtlbauer, A.; Kasberger, C.; Felbauer, G.: Open and Interoperable ICT Solution for Integrating Distributed Energy Resources into Smart Grids. In Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Germany, 6–9 September 2016.



**Citation:** Veichtlbauer, A.; Prössl Andrén, F.; Langthaler, O.; Kasberger, C.; Strasser, T.I. Open Information Architecture for Seamless Integration of Renewable Energy Sources. *Electronics* **2021**, *10*, 496. <https://doi.org/10.3390/electronics10040496>

Academic Editor: Apel Mahmud

Received: 22 January 2021

Accepted: 16 February 2021

Published: 20 February 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Abstract:** Electric power systems are currently confronted with a fundamental paradigm change related to its planning and operation, mainly caused by the massive integration of renewables. To allow higher penetration of them within existing grid infrastructures, the “smart grid” makes more efficient use of existing resources by integrating appropriate information technologies. Utilising the benefits of such smart grids, it is necessary to develop new automation architectures and control strategies, as well as corresponding information and communication solutions. This makes it possible to effectively use and manage a large amount of dispersed generators and to utilise their “smart” capabilities. The scalability and openness of automation systems currently used by energy utilities have to be improved significantly for handling a high amount of distributed generators. This will be needed to meet the challenges of missing common and open interfaces, as well as the large number of different protocols. In the work at hand, these shortcomings have been tackled by a conceptual solution for open and interoperable information exchange and engineering of automation applications. The approach is characterised by remote controllable services, a generic communication concept, and a formal application modelling method for distributed energy resource components. Additionally, the specification of an access management scheme for distributed energy resources, taking into account different user roles in the smart grid, allowed for a fine-grained distinction of access rights for use cases and actors. As a concrete result of this work, a generic and open communication underlay for smart grid components was developed, providing a flexible and adaptable infrastructure and supporting future smart grid requirements and roll-out. A proof-of-concept validation of the remote controllable service concept based on this infrastructure has been conducted in appropriate laboratory environments to confirm the main benefits of this approach.

**Keywords:** smart grid; information and communication technologies; networking infrastructure; automation; control application; renewable energy; system integration; overlay networks

## 1. Introduction

In recent years, the electric power grid had to face enormous challenges on the way to a more environmentally friendly infrastructure, supporting decarbonisation and higher

shares of renewables. This is a world-wide political goal; for example, in the year 2009, the European Union defined a set of target values for the energy industries, commonly known as 20-20-20 goals [1]. The objective was to reduce the emission of greenhouse gases, increase the share of renewable energy sources and improve energy efficiency by 20% each by the end of 2020. Renewables for power generation include technologies, like photovoltaics (PV) and wind energy plants. Since these technologies are very volatile and can only be predicted to limited extent, grid control becomes significantly more laborious and complex. Especially for PV, the use of small roof-top generation sites is very common nowadays; thus, they are feeding electricity into low voltage (LV) grids. This causes severe stability problems for LV grids, as they are massively distributed, i.e., control applications are needed for the coordination of a large number of participating nodes using respective communication means [2]. Furthermore, system maintenance (e.g., in case of security audits, technical updates, functional outages) and the need for adaptation to changing and locally different regulatory and legal conditions result in additional effort. Semi-automatic management, which can be done remotely, may alleviate the required amount of effort significantly.

Thus, the work at hand intends to contribute to an information and communication technology (ICT) infrastructure useful for smart grids, i.e., power grids with additional means to exchange, store, interpret, and edit data and to use it for control purposes [3]. This data may be used for control commands (switching consumers or producers on and off, etc.), measurement values (instantaneous power, or energy consumption in a 15-min slot, etc.), additional relevant information (price curves, weather forecasts, etc.), as well as the deployment of software components (for updates or new functions). The large number of different stakeholders (consumers, producers, network operators, facility owners, device manufacturers, maintenance engineers, authorities, certification bodies, sales organisations, etc.) generates high demands regarding openness and interoperability but also regarding the security of this infrastructure. A proof-of-concept prototype has been developed, which provides functionalities, such as a unified application programming interface (API) for control applications, a management interface for integrating new or updating existing functions, an engineering tool for developing control functions, a communication subsystem containing adapters for common smart grid protocols, easy to use integration of physical interfaces (sensors, actuators), and a role-based access control (RBAC) system for users and applications, including a policy enforcement point, providing necessary security and privacy of the operated data.

The basic ideas of this work have been previously published in Reference [4]. There, the concept and the prototype architecture have been outlined. Partial solutions have been also shown in Reference [5], where security by design aspects of the prototype architecture have been explored, or in Reference [6], where extensible messaging and presence protocol (XMPP) communication has been integrated into a distributed, standard-compliant control environment. For the engineering of control applications in such environments, Reference [7] applies a model-based system engineering (MBSE) approach. Anyhow, the present work introduces the whole concept with all corresponding details. It also provides a comprehensive discussion about the achieved results which goes far beyond the content of the aforementioned papers where mainly high-level ideas and first proof-of-concepts have been outlined.

This paper is organised in the following way: Section 2 describes the related work and state-of-the-art in the targeted field of smart grid information architectures. In addition, existing technologies and solutions which are potentially useful for the work at hand are outlined. Section 3 recapitulates our approach to overcome existing shortcomings, including the architecture of the developed prototype. A special focus on interfaces, address formats, and protocols, as well as the applied communication logic, is given in Section 4. After describing the realisation of the proof-of-concept prototype and the validation scenarios in Section 5, the gained results from conducting the respective tests are discussed in Section 6. Finally, Section 7 concludes the paper.

## 2. Related Work

The massive roll-out of distributed energy resources (DER), like wind turbines, PV systems, small hydro power plants, biomass generators, etc., over the last decades has led to a fundamental paradigm change in terms of operation and planning of the electric power system [8]. As a consequence, the electricity generated from renewable sources has become visible in power transmission and distribution systems, creating additional challenges for the management of the power system, mostly due to the large numbers of systems, the variable power output and uncoordinated response to changing conditions of the power grid [9]. In some regions of the world, the deployment of renewable energy sources (especially PV) has been reaching or even exceeding the hosting capacity of distribution grids. In order to solve this issue, an effective integration into the power system is required, which becomes even more critical for the development of future DER components [7].

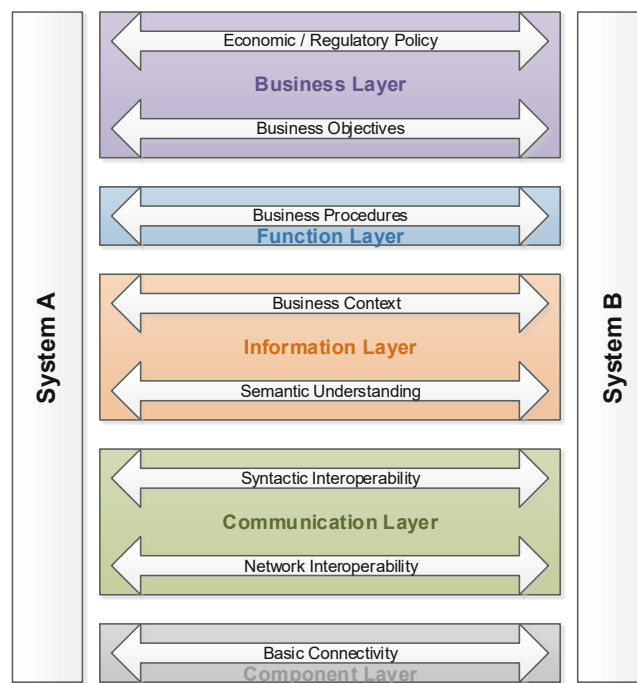
### 2.1. Meta-Models for ICT Architectures

The so-called smart grid [10] constitutes a very promising solution to use existing grid infrastructure more efficiently, which allows a higher degree of deployment of renewables [11]. All components (generators, loads, storage facilities, lines, substations, on-load tap change transformers, DER, measurement devices, smart meters, etc.) in the smart grid are interconnected via a powerful communication network and corresponding automation infrastructure in order to monitor, manage, and optimise the electric infrastructure in a more intelligent manner. To be able to fully utilise the benefits of smart power grids, it will be necessary to develop new control strategies. These strategies will make it possible to manage the large number of dispersed generators more effectively and to better utilise their “smart” capabilities, such as the ones that can be provided by inverter-based DER. For example, an overview of various research and development activities in the area of smart grids and DER integration on European and American level is discussed in Reference [12,13]. Here, ICT plays a key role for the implementation of such a smart and coordinated system approach [14]. In the dynamically evolving field of ICT solutions for smart grids, the development of standards is of crucial importance mainly due to interoperability requirements. Several standardisation organisations and various international projects have analysed this topic already [15,16].

On an international level, the International Electrotechnical Commission (IEC) provides a collection of common rules for the planning and operation of smart grids. Thus, the “IEC Smart Grid Standardization Roadmap” [10] suggests core standards to be used for the realisation of smart grids. This includes standards for data models, like the common information model (CIM), as defined in IEC 61968 and IEC 61970 [17], for power utility automation (e.g., IEC 61850 [18]), and many more. Gungor et al. [19] provide an overview of relevant smart grid standards, as well as roadmaps for further development. An impressive visualisation of standards is given in Reference [20]. The smart grid typical interaction of energy technology and ICT was specifically focused on in Reference [21].

In the smart grid domain, several proposals for organising ICT contributions have been made. The 3-dimensional smart grid architecture model (SGAM) model [22] is probably the most popular among these structuring meta-models. It includes not only interoperability layers but also zones (which refer to the parts of the automation pyramid) and domains (which incorporate the electricity grid’s value-chain from bulk generation to customer premises). An older model for interoperability layers (1-dimensional, but more granular) is the “Gridwise interoperability context-setting framework” [23] from the GridWise architecture council (GWAC). Figure 1 shows a comparison of these models. The lower layers of these models (up to the “Semantic Understanding” GWAC layer) are furthermore covered by the International Organisation for Standardisation (ISO) open systems interconnection (OSI) reference model [24] for computer networks.





**Figure 1.** GridWise architecture council (GWAC) and smart grid architecture model (SGAM) layer models (adopted from [22]).

A special focus for smart grid information architectures should be set on the “middle-ware” layers, i.e., the “Syntactic Interoperability” and “Semantic Understanding” layers of the **GWAC** stack, respectively, the **OSI** layers 4 to 7. However, if the Internet is not used as common infrastructure, lower layer technologies have to be taken into account, as well. An overview of possible technologies for appropriate communication stacks is given in Section 2.3.

## 2.2. Architectures, Technologies, and Frameworks

Before considering the communication stack, the technologies for realising smart grid applications have to be assessed: First, the applications define the requirements on infrastructure and middleware. Second, the used technologies influence the interface and, thus, indirectly also the functionalities and the quality attributes of smart grid applications. Especially control applications are often realised using technologies, such as IEC 61131-3 [25] or IEC 61499 [26]. IEC 61499 is a promising technology for the smart grid domain, as it allows for distributed and event-driven control logic [27]. In case of IEC 61499, layer 7 functionality (and due to the hierarchical abstraction of the **OSI** scheme also the respective included lower layer services) would be incorporated in so-called service interface function blocks (**SIFBs**).

The question here is which communication services and functionalities can be included into such a function block, and how this can be done (i.e., which architectural approach is taken for integrating middleware and infrastructure services). When looking at adjacent domains, similar activities can be identified. In home and building automation (**HA/BA**), an “X-Architecture” using a convergence middleware layer has, e.g., been deployed in Reference [28]. Tooling support with connection to smart grids has been provided, e.g., by Reference [29]. The automotive domain has come up with a similar middleware architecture named AutoSAR [30].

The existing approaches have varying amounts of impact, yet it can be said that, in the smart grid domain, there are no widely accepted solutions for generic data exchange. Instead, a plethora of data standards and protocols has been defined, each of them tailor-made for a very specific use case, e.g., device language message specification (DLMS)/companion specification for energy metering (COSEM) [31] is widely used for smart metering and advanced metering infrastructures (AMI) [32]. These approaches are very useful but lack the genericity to serve as a basis for a generic interoperable information architecture.

Furthermore, there are numerous approaches for smart grid frameworks, which can be considered as middleware solutions, as they implement an (X-architecture like) intermediate tier between applications and hardware/operating system, like OGEMA [29] or OpenHAB [33]). Many of them deploy Open Services Gateway initiative (OSGi) [34] as basis, as surveyed by Pichler et al. [35]. This may be traced back to the origins of OSGi, which can be located in the areas of HA/BA. However, these installations are intended to work at a local site mainly, although they may be collecting data from remote sources (like sensors in the field). The main problem is their lack of interoperability [36], i.e., the collaboration of instances of different frameworks usually fails. Here, the middleware should embody common notions for data exchange between these instances, i.e., a common protocol stack.

Still, there are numerous approaches for distributed middleware solutions besides smart grid related infrastructure, which could be deployed generically, such as the common object request broker architecture (CORBA) [37]. In the field of industrial Internet of things (IIoT), solutions, such as the data distribution service (DDS) [38] or open process control unified architecture (OPC UA) [39] (sometimes combined with time sensitive network (TSN)), are common. Distributed applications based on these technologies are still limited, as they allow cooperation of distributed partners, but only when these partners are using the same base technology. Thus, commonly agreed protocols can yield enormous benefits for all stakeholders, e.g., user rights management could be based on lightweight directory access protocol (LDAP) [40].

### 2.3. Communication Concepts and Protocols for Energy Systems

According to the OSI model [24], protocols provide a northbound interface called service access point (SAP). The SAP of OSI layer 7 protocols (application layer, L7), thus, provides a generic API for whatever applications of different vendors want to use the underlying middleware and infrastructure. One of the most widespread realisations of such a generic API is the email system. The series of actions for sending, querying, and delivering emails is given by the L7 protocols post office protocol 3 (POP3), Internet message access protocol (IMAP), and simple mail transfer protocol (SMTP), respectively. Furthermore, the handling of email attachments is described by the multipurpose Internet mail extensions (MIME) definition [41], which gives information about the structure of the data that has to be exchanged. This is a very powerful and flexible system, allowing for data exchange not only between humans but also for machine communication—for instance, it can be used for exchanging calendar updates [42].

The success of the email system is its simplicity and genericity; the question is whether this concept can be transferred to the smart grid domain. Thus, in the following, potential technologies to be used as basis for such a smart grid information architecture, mainly covering functionalities located in the layers L4 to L7, will be examined. L7 defines an application specific series of actions (e.g., for control tasks); this has to be provided via the aforementioned generic API for the L7 functionality. Just like SMTP provides an L7 SAP to email clients, this API shall provide a similar point of communication service provision for smart grid devices and applications. State information may be held in the application itself, or in the L7 protocol. Application layer protocols, as mentioned, are tailor-made for the concrete use cases; however, standards, such as IEC 60870 [43] and IEC 61850 [18], have gained prominence in many smart grid-related data exchanges, predominantly in the transmission grid infrastructure. IEC 61850 allows for a tree-like data object structure

(thus also defining L6 functionality). Whereas IEC 61850 is connection-oriented (and thus stateful), IEC 60870 uses (stateless) telegrams; this is more lightweight but may lack functionality needed for control applications. As for the semantics of the L7 payload, we seek a unique data modelling. This could be provided by the mentioned CIM [17], which has gained importance in the smart grid world over the last few years [44].

Furthermore, L6 technologies are needed which are able to represent the data models defined by CIM. A good candidate for this is OPC UA [39], which is a common technology to be used for connecting “nodes” (communicating end systems) in distributed industrial environments. It constitutes a message-oriented middleware [45], which already provides service orientation. In addition, the combination of OPC UA and CIM has already been proven to work well [44]. As for L4/L5, the focus will be set on real-time capable end-to-end protocols, preferably stateless protocols, as state information will be needed already for the L7 protocol, and a doubling could be too inefficient for the desired solution. Session handling could be performed by session initiation protocol (SIP) [46] or XMPP [47]. For the transport layer, user datagram protocol (UDP) [48] might be an alternative to the predominant transmission control protocol (TCP) [49], especially for use cases involving strict latency requirements. However, mechanisms to ensure delivery of control commands have to be defined; automatic repeat request (ARQ) could be inappropriate due to latency properties—yet reliability is also an important topic for control applications.

Regarding the communication infrastructure (i.e., the lower part of the SGAM Communication Layer), Internet protocol (IP) should be taken as granted. The only question is, which version is to be used. IPv6 [50] might have advantages over traditional IPv4 [51] for the huge number of possible addresses, which makes it more suitable to IoT environments. This applies especially to the smart grid, due to the numerous devices in the field (e.g., power sensors [52]). However, field devices often face performance problems, which could be worsened by using 128 bit long addresses—6LoWPAN [53] yet provides IPv6 with header compression. Depending on the requirements of L2, this might be an option. As for L2, IP support is strongly desired. Thus, IP capable field buses, like Modbus (in the variants Modbus/TCP and Modbus/UDP) [54], will play an important role. Furthermore, L2 should be real-time capable—so real-time Ethernet variants, such as Ethernet Powerlink [55], could be considered. This would also define L1 (the physical layer).

#### 2.4. Open Issues and Shortcomings

To summarise, there are a couple of open and unsolved issues when it comes to the integration of DER units into smart grid solution. From an automation and ICT-point of view, the most relevant topics for the present work are:

- missing common automation application modelling concepts for power and energy systems,
- scalability and openness of automation and control applications with a high share of DER components is only partly addressed,
- lack of common and open communication interfaces in smart grids impede scalable and distributed automation solutions,
- missing possibilities to update and extend DER services, and
- available proprietary automation concepts in smart grids prevent efficient reuse of control software; thus, the engineering costs exceed admissible costs by far.

In the following sections, a potential approach that overcomes those issues is introduced and discussed in detail.

### 3. Approach and Architecture

In order to overcome the mentioned shortcomings, a concept for the desired ICT infrastructure has been developed, and a proof-of-concept prototype has been implemented and validated. This section outlines the elicitation of requirements on the solution and the design of an appropriate information architecture, as well as the basic functionalities covered by the prototype components.

### 3.1. Use Cases and Requirements

Potential application scenarios of the approach at hand have been defined and analysed on the basis of a use case analysis in order to derive respective functional requirements. This use case methodology is widely used in ICT engineering and provides a structured approach to identify requirements for a system under design in a technology independent way. In addition, in the domain and energy systems domain, the usage of such a methodological approach becomes more and more important. The “IntelliGrid Methodology for Developing Requirements for Energy Systems”—which is covered by the IEC 62559 standard [56]—defines a corresponding approach for defining use cases and deriving requirements especially for the smart grid. This very well known methodology usually separates the concepts of user requirements from technical specifications; therefore, user requirements define *what* is needed without taking any specific designs or technologies into account. On the other side, technical specifications define *how* the specified systems have to be realised in order to fulfil the user requirements.

The definition of actors is one of the crucial parts in the process. An actor is an entity having a defined behaviour when interacting with the system under discussion. In principle, actors can be humans, organisations, (technical) components, and systems (or even systems of systems). The identification of actors has already been carried out in various smart grid related activities so far (DKE [57], M/490 [58], etc.). Therefore, a survey of existing projects has been carried out and relevant actors have been collected and evaluated. Useful actor definitions have been included in a use case management repository hosted at DKE [57]. On that basis, 25 relevant use cases in 6 use case clusters as visualised in Figure 2 have been collected and analysed. Hereby, the operational use cases cover some intended control applications, whereas the process-oriented use cases describe the interaction with the proposed system itself.

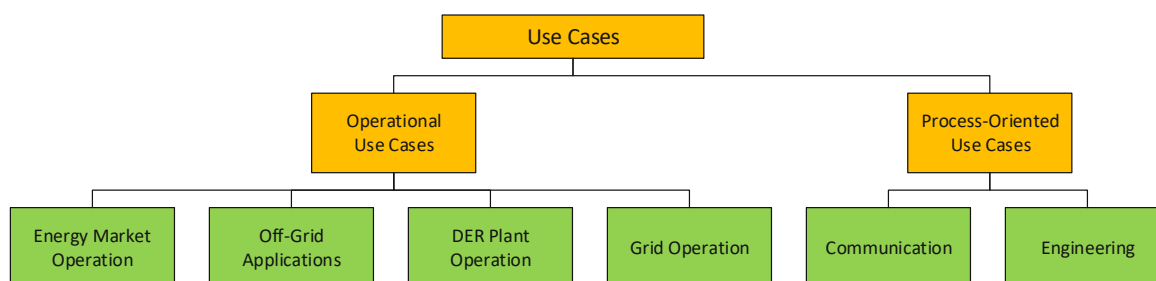


Figure 2. Overview of identified use case clusters.

The process-oriented use cases yielded up to about 40 requirements, categorised into rights management, communication, and application requirements, as depicted in Figure 3. Finally, validation scenarios have been derived (see Section 6), which cover one or more of the mentioned use cases, as well as validation environments, which hosted the tests for these validation scenarios.

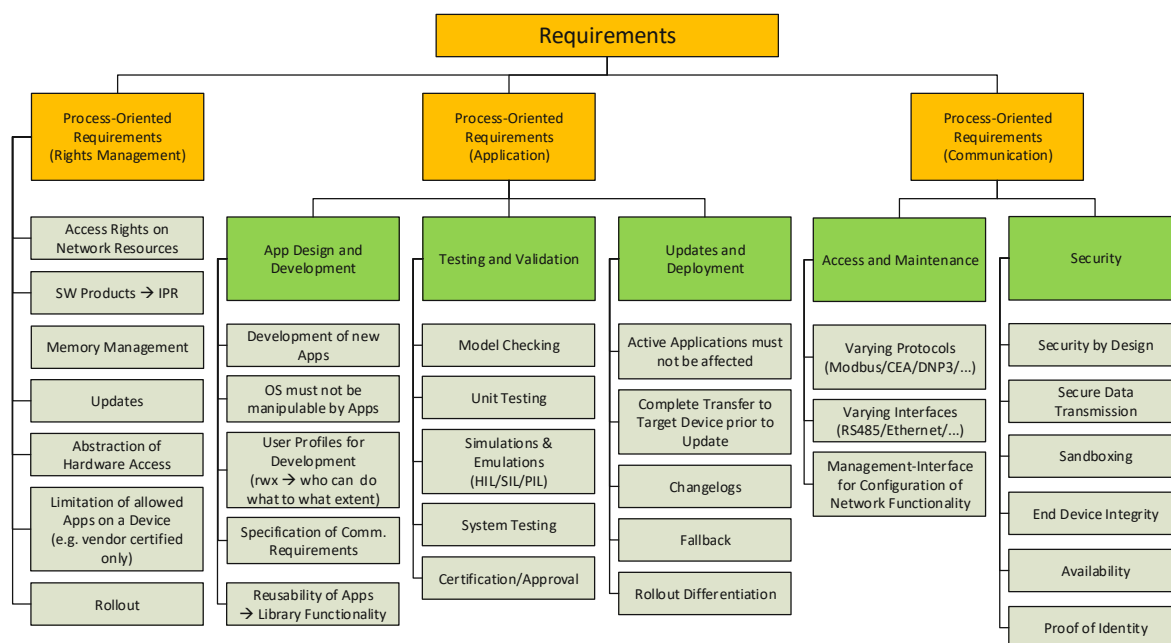


Figure 3. Overview of derived requirements.

### 3.2. Overall Concept and System Architecture

Based on the aforementioned requirements, a prototypical solution has been designed and implemented in order to serve as a proof-of-concept. The overall system architecture of this prototype is depicted in Figure 4. Hereby, the power flow (solid lines) and the data flow (dotted lines) between different intelligent electronic devices (IEDs)—be it transformers, DERs, remote terminal units (RTUs), or any other kind of IED—are visualised. Each IED comprises hardware and a middleware named “SmartOS”, as well as “remote controllable services (RCSs)”. The SmartOS again contains hardware control, security and access management, as well as connectivity functions. The RCSs access the middleware via a defined API only, and again provide functionality, such as voltage measurements, needed by smart grid applications, like Volt/var control. These services have been designed to run in sandboxes; thus, they can communicate with their environment only via the middleware.

In case such RCSs want to communicate with other RCSs on different IEDs, the SmartOS has to provide the connection to a remote IED via an appropriate connectivity module. Figure 5 shows the respective module for external communication in the context of the overall system architecture depicted above. One of the main goals of the proposed architecture is to provide flexibility with regard to the integration of different smart grid-related protocols since there is currently no unified solution available. Typically, protocols, like Modbus, DNP3, or IEC 61850, are used, and the proposed approach allows to integrate all of them concurrently, as depicted in Figure 5.

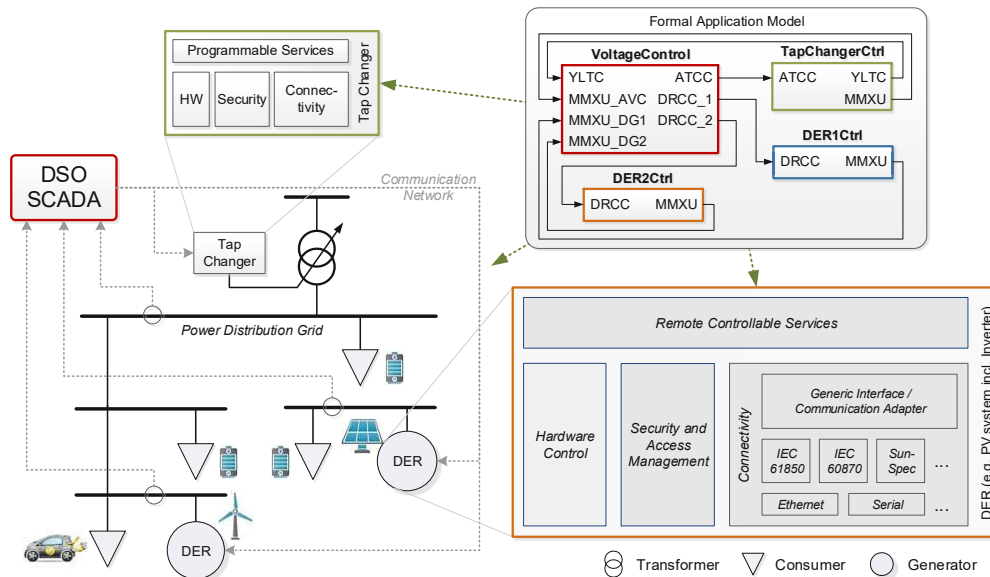


Figure 4. Overview of the overall system architecture (adopted from Reference [4]).

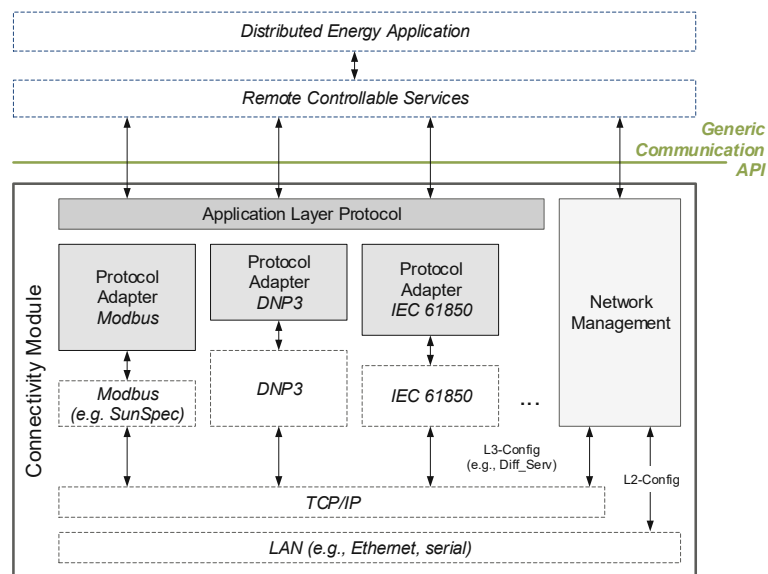


Figure 5. Elements of the connectivity module.

The main parts of this module are:

1. a common application layer (L7) protocol called “open application layer protocol (OpenALP)” (see Section 4.3) providing a unique access point (L7 SAP) to connectivity functionality for RCS,
2. legacy protocols widely used in the power and energy systems domain,
3. adapters for missing functionality of these protocols (e.g., for keeping state information),
4. existing network infrastructures (e.g., on the basis of the TCP/IP/Ethernet stack), and

- network management functionality to provide the possibility of (re-)configuration during runtime.

### 3.3. Subsystems and Functionalities

An overview of the software within one IED is given in Figure 6. Each IED contains the SmartOS middleware, which serves as container for hardware control (e.g., access to sensors and actuators), connectivity (communication to other IEDs), and security (especially access control to restrict functionality to users with respective rights), as well as a registry for services (in an OSGi-like manner) and users (based on LDAP [40]). Despite its name, the SmartOS, thus, actually constitutes a middleware rather than an operating system (OS).

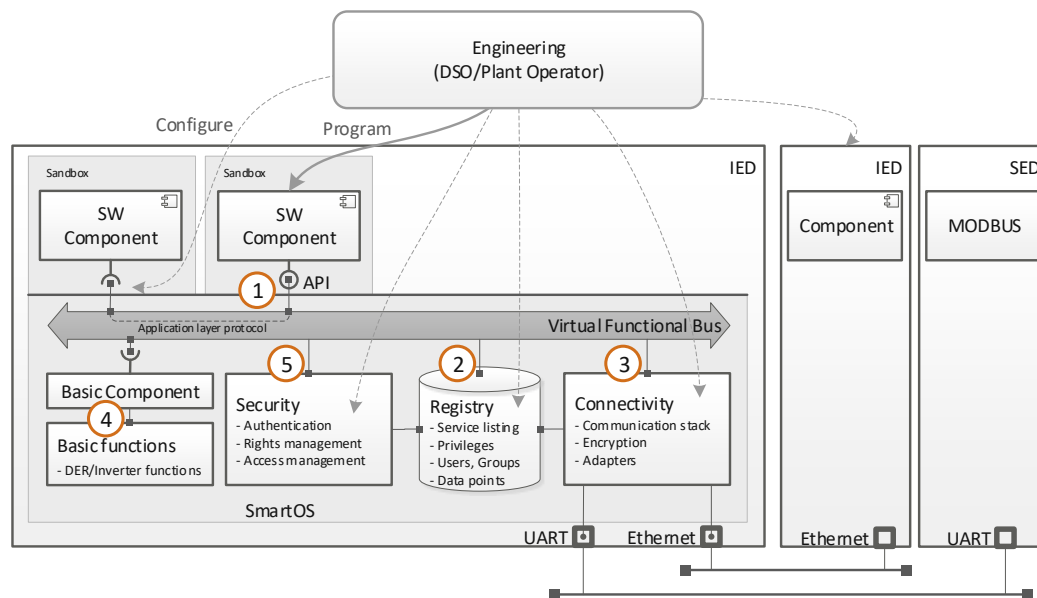


Figure 6. SmartOS and Virtual Functional Bus (VFB) (adopted from Reference [4]).

#### 3.3.1. The Virtual Functional Bus

The core subsystem of the SmartOS is the “virtual functional bus (VFB)”, which provides intra-IED communication for different RCSs denominated as “software components”. This is done via the specifically developed OpenALP (see Section 4.3). Furthermore, the VFB connects the SmartOS modules, which are described in more detail in Section 3.3.2. The numbers shown in Figure 6 relate to the interfaces provided by the VFB. As only exception, interface 4 connects the basic component (which is a pre-defined software component encapsulating direct hardware access) to its respective basic functions, i.e., sensor and actuator access. To fulfil the requirements, the VFB needs to provide an ecosystem which is acting as a message oriented middleware (MOM) designed for a highly distributed environment.

Consequently, IEDs and their software components, as shown in Figure 6, will exclusively be provided with a predefined address scheme (see Section 4.1) for communication. They do not require information about the location of their communication partners or the appropriate protocol and physical adapter needed for communication. Such an infrastructure can be provided by a classical messaging system, which is explained in more detail in the following. Furthermore, the VFB represents the central point for dispatching messages by querying the registry and security modules and finally forwarding the messages to the correct recipient(s).

The messaging system is the basic functionality of the VFB. As such, it must provide a level of functionality similar to message queues, where each possible component is provided its own input and output queue. They need to be based on classical first in first out (FIFO) mechanisms which can be filled with a specific number of messages and ensure sequential delivery. Messages are received and sent by the VFB itself, by software components, or by the connectivity module.

The SmartOS middleware allows adding RCSs on a plugin basis. Ideally, they can be deployed as a package without the need for copying many different files. Third parties should be able to develop such software packages and they should be reviewed and certified by the hardware manufacturer of the device they are intended to run on. When certified for the brand/device, such software components ideally are signed by the owner and by the company reviewing the software component, using a commonly known and established signature methodology. Signed software components finally receive standard permissions on a device to run in the sandboxed context of the VFB, which limits system access to the provided VFB API described in Section 4.

The design of the software component plugin system should allow hot-plug features, such that new applications can be loaded and updated dynamically during runtime. As the VFB is based on a messaging system, a software component needs to be able to access this system. To achieve this, the VFB infrastructure provides the mentioned common API (interface number 1 shown in Figure 6), which is accessible to internal components (e.g., the basic component), as well as to the loaded third-party applications. Furthermore, communication should be restrictable for an individual or a group of communication parties. This has to be conducted according to the permission system handled by the security module and the device registry. However, the VFB itself acts as the respective policy enforcement point (PEP) which processes or discards the messages accordingly.

### 3.3.2. Modules

The SmartOS architecture contains two main important modules which are:

#### Connectivity Module

IEDs may communicate with other IEDs or RTUs via the connectivity module, using various kinds of protocols. The connectivity module has the ability to establish new connections or handle in-coming messages from established servers (instantiated by the VFB) via communication technologies, such as Ethernet, serial connections, or field busses, like controller area network (CAN) bus. To do this, the connectivity module is aware of the necessary adapters and protocols; these are determined via registry entries. Furthermore, adapters are implemented in a way to allow non-blocking operation and multiple simultaneous connections, if the respective protocols require such functionalities.

If the connectivity module establishes a connection, some mapping needs to take place to check if there is already an open channel for that particular communication partner to continue communicating on that channel. This is handled via a simple lookup-table. When reconnecting on the other hand, an adapter and a protocol need to be chosen; this is done by translating the system address into an adapter address. Finally, it is possible to execute protocol specific tasks to prepare the open channel for communication. This is achieved by the respective protocol adapters. When receiving new connections, the adapter and the protocol need to be pre-defined because an incoming request is handled by the particular server implementation of the adapter. Thus, an incoming message needs to be verified (if applicable) and forwarded to the VFB.

#### Security Module and Registry

The security module is one of the central elements ensuring security within the infrastructure at hand. Therefore, it needs to be able to impose a permission strategy on the messaging system, thus serving as a policy decision point (PDP). It has also to restrict loading only to signed software components and to provide the necessary information to



the sandbox in order to allow or deny a specific function call. Additionally, the security module needs to know which functions and configuration parameters are accessible and to which extent. An important pre-condition to provide an appropriate rights and access management is to authenticate requesting modules and their respective users in cooperation with the registry. On each SmartOS capable device, a persistence and configuration layer needs to be maintained by a central authority. This part is referred to as the registry. It needs to contain a variety of information required for the normal operation of the device. These configuration entries include the following types of information:

1. Device: Device name, device interface, application folder,
2. Address: All accepted outbound and inbound addresses,
3. Registered applications: Registered running applications, including their data points,
4. Data points per application: Manually registered data points,
5. Adapter: All adapter implementations which run as a server thread,
6. Users: All registered human or machine users and their respective access rights, and
7. Permission: Permissions based on (at least) a 3-tuple of sender, recipient, and communication type, supporting wildcards.

### 3.3.3. Internal Interfaces

Here, the most important interfaces between the various components internal to the IED are considered. Internal in this case refers to the fact that under normal circumstances, they will not be particularly relevant to any third parties implementing software components or adapters for the connectivity module. The internal interfaces are accessed by the VFB only, i.e., they connect the VFB to a functional module. The interface between the VFB and the security module (interface 5 in Figure 6) can essentially be limited to a single function call: For any message arriving via the connectivity module, the VFB must query the security module (which in turn must query the local database and/or a central permission store, such as an LDAP server), for existence of the appropriate permission. If a corresponding entry is found, true should be returned and the message can be passed on by the VFB. Otherwise, the message has to be discarded by the VFB.

The VFB and the connectivity module (interface 3 in Figure 6) should be linked in a sequential and non-blocking way, e.g., via a set of message queues. Communication needs to be completely encapsulated and only possible over the messaging system specified above from and to the connectivity module. In order to enable software components to register and de-register data points, the VFB needs to be able to access the registry (interface 2 in Figure 6) as persistent data storage. Furthermore, since the VFB acts as a PEP, it must be able to indirectly access the registry via the security module which, in turn, acts as a PDP relying on the corresponding entries in the permission table.

### 3.4. Engineering Approach

Besides the prototypical realisation of an IED-based ICT infrastructure, an appropriate tooling was also necessary to provide. This tooling had the task to support engineering of smart grid applications from design over implementation and testing up to deployment and maintenance. Thus, this engineering environment should support the whole product lifecycle, e.g., by hot plug&play of new or modified services and applications in an OSGi-like manner, but tailor-made for the smart grid domain, rather than for HA/BA.

The SmartOS concept of RCS sandboxing (see Figure 6) in combination with access right enforcement allows for using services in the intended flexible way. Thus, access to system resources is granted via a controlled access way only but can yet be modified easily during runtime. However, these modifications have to be performed solely by system services in order to be done again under complete access control (it has no sense to enforce access control rights for applications, when these rights can be manipulated without control), while being easily invoked by legitimate users. This means system services are engineered the same way as energy services.

Energy services can then be utilised to engineer higher order applications, as for instance Volt/var control in LV grids (see Section 6). To enable engineering in an easy to use way, as well as for allowing reuse of existing functionalities, a model-driven architecture (MDA)-based approach has been chosen, as given in Reference [7]. According to MDA, the platform independent model (PIM) is created using a domain specific language (DSL), here called the power system automation language (PSAL). PSAL is intended as a tool for SGAM compatible use case design and at the same time it provides a basis for further code generation. Thus, PSAL is also the starting point for all design activities in the provided engineering methodology. As an example, in Reference [7], IEC 61499 function blocks (FBs) are used to realise a Volt/var control application, engineered with the help of PSAL.

Since SGAM is already divided into different layers, it makes sense to keep this structure in PSAL, as well. When studying the SGAM layers, it is clear that the component layer and the communication layer are more “static” (in the sense of related to existing infrastructures) than the information, function, and business layers. According to this, PSAL was divided into one static and one dynamic model, the latter also called the functional model. A graphical representation of the main parts of PSAL is provided in Figure 7. In the static part, the component layer and the communication layer are represented, whereas, in the dynamic part, the information layer, the function layer, and the business layer are represented.

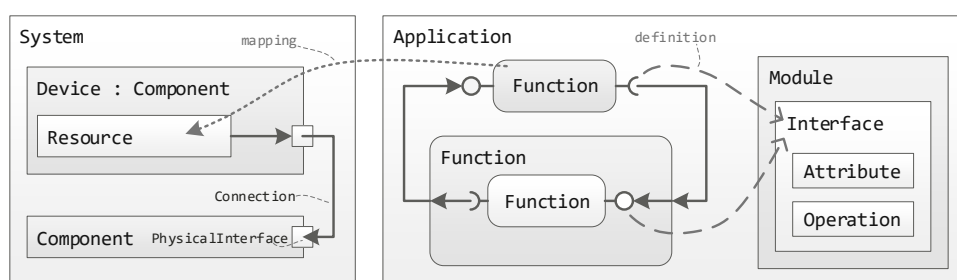


Figure 7. Elements of the Power System Automation Language (PSAL) [4].

Another important requirement is platform independence. An application or functionality should be independent from its possible execution platform. Expressed in SGAM, this would correspond to using the same business, function, and information layers, while using different communication and component layers. From a user perspective, this means that it must be possible to define a functional model independently from the static model.

According to Reference [7], the static model is represented by the *system*, a container for *components* and *connections*. Together, these can be used to describe the ICT system, as well as the electrical power system. In this work, the emphasis is on ICT and automation equipment. Examples of ICT equipment are routers and switches, while automation equipment is mainly represented by IEDs and embedded controllers. Each *device* contains one or more physical communication interfaces. Part of the device is also the *resource*, a function container similar to the logical devices (LDs) used in IEC 61850. By means of multiple resources, it is possible to logically group functionality of a device. To complete the component layer, devices can be connected to each other through connections. Each connection can be defined through a number of *attributes* (e.g., communication protocol and performance requirements). With the system, the component layer and the communication layer of SGAM are modelled.

The *functions* are defined in the *application*, the dynamic model of PSAL as discussed in Reference [7]. Here, the exchanged information is also defined. Using the SGAM methodology, it is possible to roughly describe functionality in different unified modelling language (UML) diagrams. Use case, activity, and sequence diagrams are tools that are often used for this purpose [22]. However, in order to automatically generate code from the

use case description, more details are needed in the descriptions of the functions. In *PSAL*, this is modelled by the functions, which are defined through a software component model, a containment model that supports different levels of detail. The component model was developed with focus on the function layer rather than on the business layer. Each function is a component that can contain other functions. This allows the developer to choose the level of detail that is necessary for the use case. Furthermore, each function can provide or request services that are defined by an *interface*. The information of an interface is defined using attributes and *operations*. The application represents the business, function, and information layers of *SGAM*.

According to Reference [7], the connection between the system and the application is realised by a function-resource mapping. By performing such a mapping, all provided and requested services of the respective function are also available on the interface of the resource. Consequently, these services are also available on the physical interface of the device. One of the advantages of using a textual syntax is that comments can be used anywhere in the source code. As a result, the documentation of the constructed use case is simplified. A major part of *SGAM* is the placement into domains and zones. However, this placement is mainly intended for documentation purposes and does not directly affect the implementation. For the execution of code, it is not important if a device is located in a station or a field zone. With a textual *DSL*, the definition of domains and zones can be achieved through the use of annotations (e.g., @DER and @Station).

#### 4. Data Semantics and Interface Definition

To realise a concept as stated above, some data semantic issues have to be considered. In addition, some formal definitions have to be given, especially when it comes to the integration with external (system context) components. In the concrete prototype environment, addressing objects uniquely within in the whole envisaged system is a crucial precondition to allow for wide area communication. Furthermore, defining the data semantics when exchanging or processing data is inevitable. However, in order to allow for integrating elements of different legacy, adapters to the used mnemonics have to be used. Thus, this section intends to document all the necessary working conditions for the prototype being able to proceed.

To reduce the complexity of software components, a specific “OpenALP”, along with an address scheme, has been developed. It is intended to enable software components to communicate with other devices or software components with little or no knowledge about the various possible protocols which may be involved in cross-device communication. First of all, the specification of the developed “OpenAddress” format is provided.

##### 4.1. Address Format

The following definition, which is formulated in extended Backus Naur form (EBNF), shows how device addresses are formed in the work at hand:

```
OpenAddress = Device "/" Namespace ( "." Namespace ) * "$" Variable ( "$" Variable ) *
```

Hereby, the terms describe the following semantics:

Device	The name of the target device
Namespace	A namespace within a device—Namespaces are defined by the software components; each namespace typically corresponding to one software component.
Variable	The address of a data point (which can be a simple variable but also a structure) within a namespace—Variables are also defined by the software components.

The following example shows a complete address in OpenAddress format:

```
IED1/Measurements.POC.Voltages$PhaseToNeutral$phsA
```

#### 4.2. Mapping between OpenAddress Format and Legacy Addresses

To be able to translate an OpenAddress into a hardware address which can be used by a protocol adapter, an address mapping needs to be created and stored as an address mapping entry in an address translation table. In theory, several available routes with different adapters may be possible, so each address mapping allows for the declaration of multiple adapters. The corresponding EBNF definition is as follows:

```
AddressMapping = OpenAddress ":" "{" AdapterName "[" AdapterAddress  
]" (" AdapterName "[" AdapterAddress "]" )* "
```

Again, the meaning of the terms is as follows:

**OpenAddress** The OpenAddress of a data point; see above.  
**AdapterName** The name of the corresponding adapter (e.g., "MODBUS", "IEC61850").  
**AdapterAddress** The internal address of a data point within the adapter/protocol.

The syntax of AdapterAddress is adapter specific, two examples are specified below:

```
AdapterAddress = ModbusAdapterAddress | Iec61850AdapterAddress | ...
```

The ModbusAdapterAddress is the address scheme for the MODBUS [54] adapter, which is defined as follows:

```
ModbusAdapterAddress = ModbusEntity "," ModbusAddressRange  
ModbusEntity = "Coils" | "DiscreteInputs" | "HoldingRegisters" |  
"InputRegisters"  
ModbusAddressRange = INTEGER (".." INTEGER)?
```

Here, the ModbusEntity specifies the register type of the given address, whereas the ModbusAddressRange is the range of valid addresses.

The Iec61850AdapterAddress denotes the address scheme for the IEC 61850 [18] adapter and is defined as follows:

```
Iec61850AdapterAddress = LDevice "/" LN "." DO "." DA ( "." BDA )*
```

Here, the adapter address consists of an IED name, a logical node, and several hierarchical structures of datapoints.

To aid comprehension, the following examples show complete addresses in OpenAddress format and how the contained variables are derived from adapter datapoints:

```
IED1/Measurements.POC.Voltages$PhaseToNeutral$phsA  
: {MODBUS[HoldingRegisters, 4523], IEC61850[DERCtrl/  
MMXU1.PNV.phsA.cVal.mag.f]}  
IED1/Measurements.POC.Voltages$PhaseToNeutral$phsB :  
{MODBUS[HoldingRegisters, 4524..4525], IEC61850[DERCtrl/  
MMXU1.PNV.phsB.cVal.mag.f]}
```

#### 4.3. Open Application Layer Protocol

The prototype specific L7 protocol OpenALP is the second key element to ensure low development overhead within software components. In combination with the OpenAddress format, software components require little or no knowledge about the various possible protocols which may be involved in cross-device communication. Regarding

**OpenALP**, this property is achieved by abstracting the generic communication patterns from the underlying communication protocols which are considered to be required within the prototype architecture. These communication patterns form the basis for the **API**, which is provided to the **RCS** by means of the SmartOS (see Section 3).

- **Push 1:1**: Refers to a message being transferred to a single destination software component or device. This functionality may be used, e.g., for setting values, thereby influencing the operation of a remote system. A real-world analogy is the shipment of a letter.
- **Push 1:N**: Is very similar to Push 1:1, the only difference being that with Push 1:N, more than one recipient may be specified. This functionality is an abstraction from broadcast or multicast messages, which exist in several protocols. It may be used, e.g., for setting values in multiple devices simultaneously or for transmitting notifications which concern more than one device. A real-world analogy is the shipment of bulk mail.
- **Request/Response**: Describes the generic process of requesting information from a target. The request is sent and a response containing the requested information, or at least an error message, is expected. It may, for example, be used to retrieve data relevant to local control decisions, such as measurement values or price information, from a remote system. The most similar real-world analogy is a registered letter.
- **Publish/Subscribe**: This communication pattern is used whenever automatic updates about a data point are desired, the requesting entity being denoted as subscriber and the entity sending the updates as publisher. The conditions under which messages are published may vary. Typically, a message is sent whenever the subscribed value changes, but minimum and maximum time intervals between messages may be defined. A minimum time interval may be used to prevent excessive traffic, while a maximum time interval may serve as notification of continued availability of the publishing party. The most similar real-world analogy is a newspaper or magazine subscription.

## 5. Prototype, Testbed, and Validation

Following the definition of the conditions and the system context of the intended prototypical solution, a proof-of-concept implementation has been performed. The prototype has then been used in an appropriate testing environment. This section informs about both the prototypical implementation, as well as the testbed environment in which the prototype has been used and tested.

### 5.1. Prototypical Implementation

A platform specific prototypical implementation of the SmartOS framework has been performed in order to provide a test environment suitable for validating the approach (see Figure 8). In the scope of the work at hand, this test environment has been developed based on Java, as Java allows for a maximum of platform independence and many useful libraries, for example, for serialisation and queuing functionalities or for legacy protocols, are available. In addition, in combination with **OSGi**, Java has been used for multiple platforms in **HA/BA**, as well as in smart grid environments [35].

While the prototype is not a fully-fledged implementation suitable for productive use, it provides all of the functionality required for validating the approach (see Section 5.3). It is designed to conceptually cover all aspects of the presented architecture, thereby ensuring the viability of the envisioned solution.

In order to meet the previously defined requirements and provide as much flexibility as possible, a highly modular design approach has been taken. Figure 8 provides an overview of the different components and the respective interfaces of the implemented prototypical software.

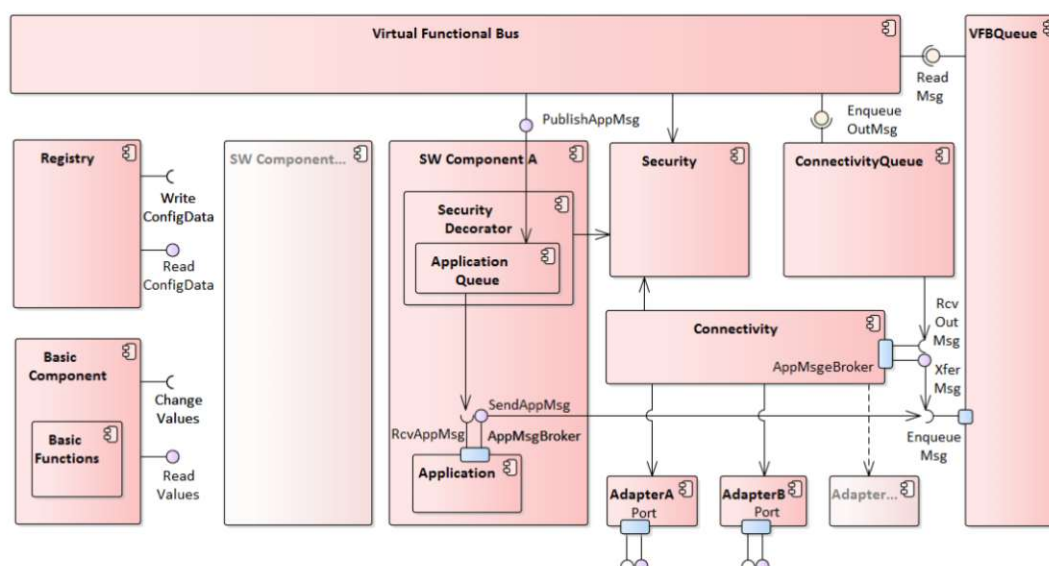


Figure 8. Platform specific implementation of the SmartOS.

### 5.2. Testing Environment

The environment used for testing, simulation and validation is based on a VMware vSphere hypervisor running various CentOS and Debian virtual machines. The hardware consists of a compact laboratory server which has been assembled using then state-of-the-art hardware (4 Core Intel Xeon CPU, 16 GB RAM, 512 GB SSD storage) and is used exclusively for the work at hand.

Testing, simulation and validation have largely been carried out using a number of virtual machines running the various prototypes as sending and receiving elements (representing the various involved devices and stakeholders). In order to provide a more realistic environment, a network infrastructure, including several virtual routers and switches, has been set up. The network topology, including the various virtual machines, is illustrated in Figure 9. This was complemented by temporary virtually private network (VPN) infrastructure allowing external and/or remote devices from another lab to remotely connect to any of the subnets as needed.

vSphere allows the link speed of virtual switches to be defined in 1 KB/s steps, which makes it possible to simulate connections with low bandwidth, such as legacy powerline communication (PLC) networks. Furthermore, OMNeT++ and Mininet were conceptually considered to simulate more complex intermediate transmission paths and connections with high load (denoted Transmission Path Simulator in Figure 9), but extensive simulation and performance evaluation was ultimately deemed beyond the scope of this work.

On the machine denoted credential store, enterprise JavaBeans certificate authority (EJBCA) has been used in combination with open lightweight directory access protocol (OpenLDAP) to provide a public key infrastructure (PKI), as well as the necessary user and rights management capabilities. Both products are widely used and actively maintained state-of-the-art open source solutions that may be used similarly in a production environment.

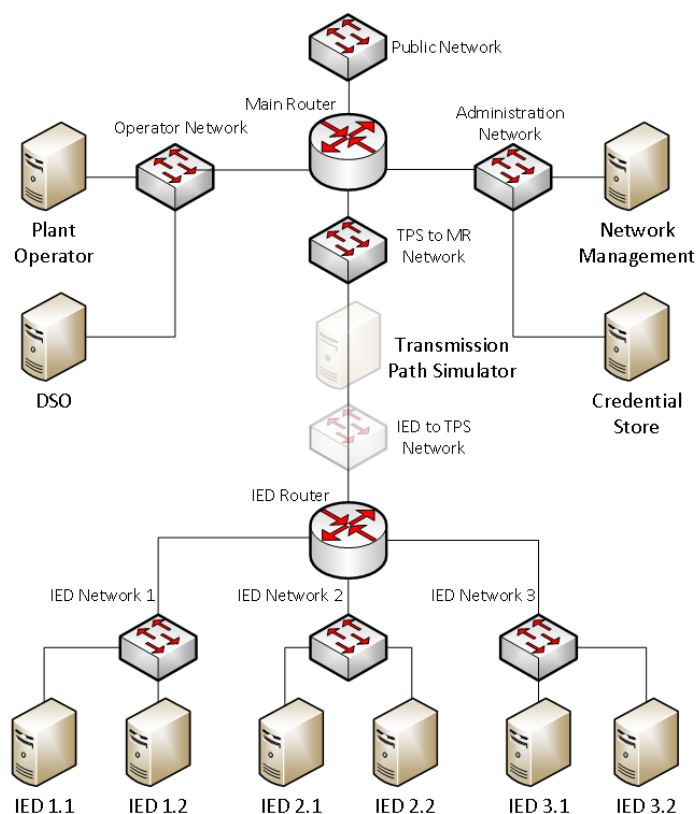


Figure 9. Realised testbed for validation and testing.

### 5.3. Validation Approach

The validity of the approach finally had to be documented by executing appropriate test scenarios, gathering the respective results of the conducted tests, and contrasting these results with the defined requirements, which is discussed in this section.

First, the requirements regarding the proof-of-concept solution have been derived by analysing the process-oriented and operational use cases outlined in Section 3.1. Operational use cases cover typical control tasks, which have to be performed by operational systems; thus, the developed system had to show via a concrete example, that these use cases can actually be performed. Process-oriented use cases cover functionalities to provide easy-to-access engineering tools and methodologies, as well as flexible utilisation of a reliable, secure, and robust communication infrastructure.

Thereafter, the concrete test cases for the system have been defined on basis of a set of validation goals and requirements, including pre- and post-conditions for successful testing. Accordingly, the respective test environments have been provided. The test cases have been conducted as unit tests, e.g., for the Java-based communication infrastructure, appropriate JUnit tests have been performed in the virtualised network environment described in Section 5.2.

Having successfully executed the envisaged functionalities in the respective unit test environments, an integration test has been defined in order to prove, that the participating building blocks are cooperating as intended. This integration test was performed with real hardware (i.e., an inverter-based DER provided by Fronius) at the AIT SmartEST lab [59] and the Fronius system test laboratories to prove the potential of the developed solution for real world scenarios (see Figure 10). This scenario covered the setting and

getting of inverter data points by an application engineered with the developed engineering toolset and utilising the developed communication infrastructure (including the SmartOS middleware and the OpenALP, as well as the certificate authority (CA) and LDAP-based security infrastructure).

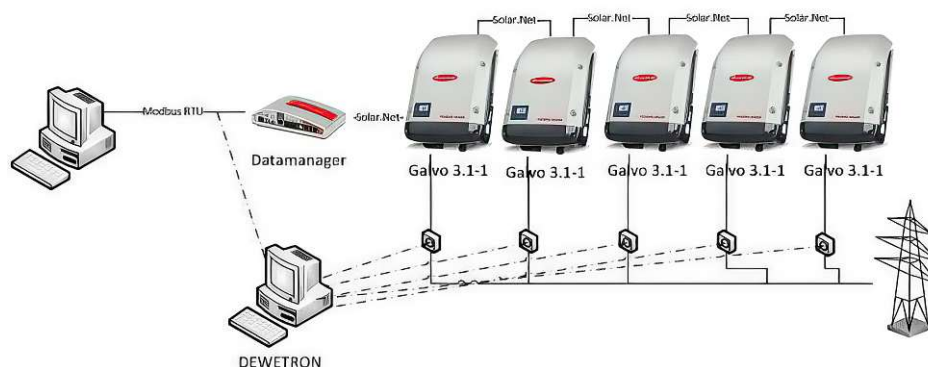


Figure 10. Test setup for remote controllable services.

The high level goal of the proof-of-concept validation was to prove the viability of the overall approach and the developed prototype system via concrete test cases. Prior to the execution of these tests, the test scenarios and the respective test environments to validate the proof-of-concept realisation needed to be defined.

Since the realisation had been split into three major parts, the test cases have accordingly been grouped into three major validation scenarios. An additional validation scenario had been defined in order to test the interoperability of the three parts of the developed system.

In the following sections, the major requirements of these validation scenarios, as well as the scenarios themselves and the results, are outlined. However, the application modelling and engineering aspect of the proof-of-concept validation has already been covered in detail in Reference [7] and is not the focus of this publication; therefore, it is not discussed here.

## 6. Performed Experiments and Achieved Results

To ensure targeted experimentation and meaningful results, a set of subsequent validation requirements (VRs) based on the functional requirements outlined in Section 3.1 was defined for each validation scenario. They are listed at the beginning of each scenario to give an impression of the intended outcomes.

### 6.1. Remote Controllable Services

For the basic DER service part of the proof-of-concept validation, the requirements have been identified as follows:

- VR1-1: Ensure the creation of extended DER services, and
- VR1-2: Ensure the integration of extended DER services in inverter-based DER devices.

The validation of the basic DER services, as well as the remote programmability, was performed in Fronius's system test laboratories. The test system comprises several network routes to connect inverters to each other, as well as to a remote supervisory control and data acquisition (SCADA) system.

The following two main validation scenarios have been realised:

- *PowerLimitChanged*: This scenario was used to validate the ability of the DER to remotely control its AC power output. The inverter itself is remotely controlled via Modbus/TCP command; the power output is rated in steps of 1 percent of the in-



verter's nominal AC output power. The power reduction according to the remote signal was measured via a Dewetron AC network analyser.

- *CosPhiChanged*: This scenario was used to validate the ability of the DER to control its percentage distribution between active and reactive power output. The inverter itself is remotely controlled via Modbus/TCP commands; the power factor  $\cos(\phi)$  is adjusted in steps of  $10^{-2}$  between its lower and upper limits. The  $\cos(\phi)$  delimitation according to the remote signal was again measured via the Dewetron AC network analyser.

Figure 10 provides an overview of the realised test setup for validating the basic DER services using 5 solar inverters connected to the Fronius datamanager box (DM-Box) controller equipped with the SmartOS-compliant prototype.

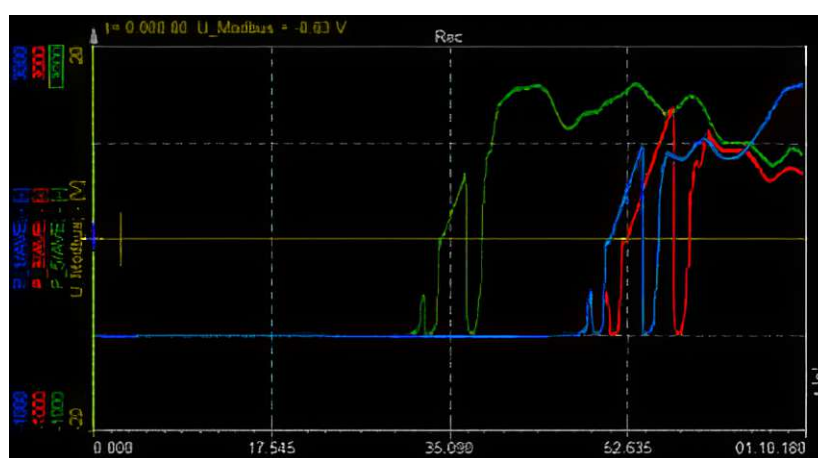
The solar inverters have been remotely controlled via Sunspec/Modbus messages. The time difference between Modbus signal and the reaction of the inverters was measured. Remote commands used for this experiment were startup and shutdown commands, as well as power reduction and  $\cos(\phi)$  variation. In Figures 11–13 and Tables 1 and 2, the startup and shutdown measurements are provided along with the key characteristics.

**Table 1.** Measurement results: startup for 3 phases ( $P = \max$ ,  $\cos(\phi) = 1$ ).

Measurement	Time [s]
t (P_1)	56
t (P_3)	52
t (P_5)	35
Average	48

**Table 2.** Measurement results: shutdown.

Measurement	t (P_1) [ms]	t (P_3) [ms]	t (P_5) [ms]
1	725	677	702
2	839	790	815
3	775	757	778
4	794	745	780
5	832	814	838
Median		780	



**Figure 11.** Startup for 3 phases (I).

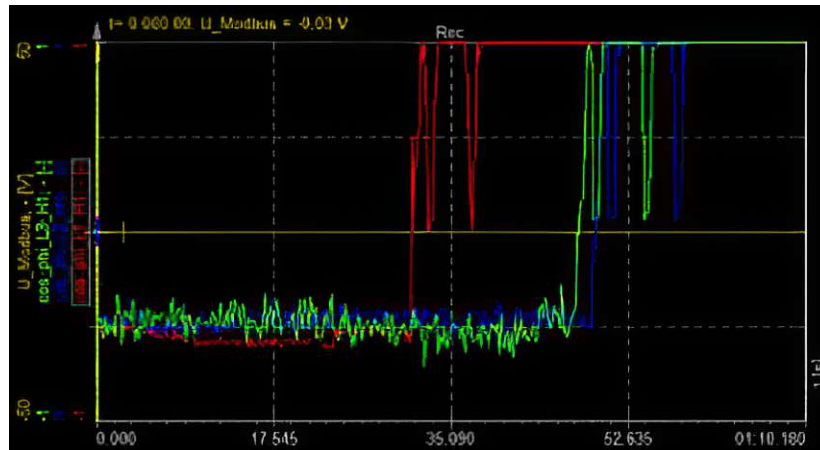


Figure 12. Startup for 3 phases (II).

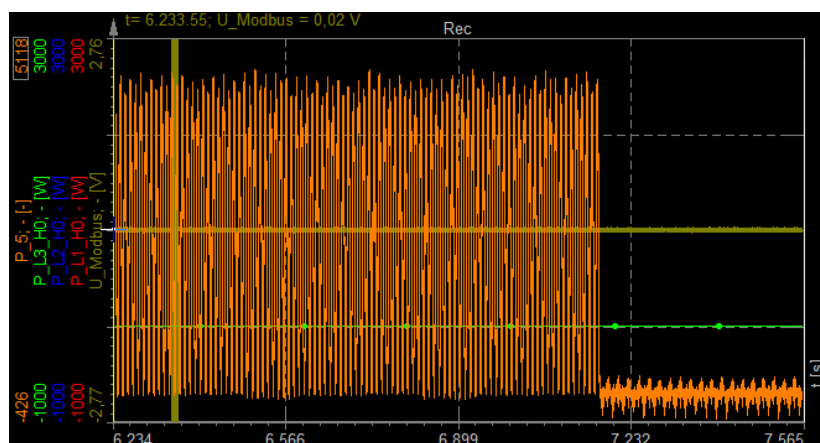


Figure 13. Shutdown scenario.

In summary, it can be said that this validation scenario has successfully proven that the remote control of DER services enables direct influence of the grid environment. It should be noted that inverter-based DER units (in this scenario, conjoined PV inverters) controlled remotely via SCADA can stabilise, but also destabilise, the energy grid. The use of such functionality must, therefore, be strictly controlled and performed with great care in order to avoid disturbances of the energy grid.

### 6.2. Generic and Interoperable Communication

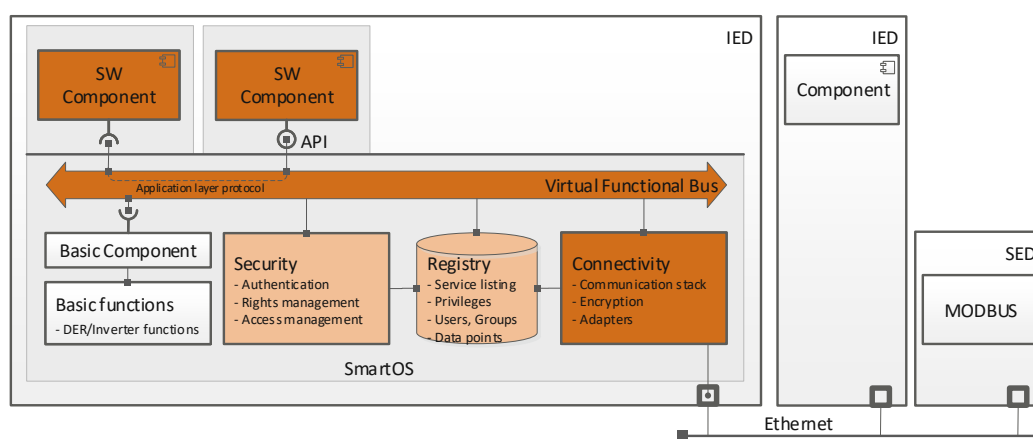
For the communication part of the proof-of-concept validation, the requirements have been identified as follows:

- VR2-1: Ensure internal communication of applications,
- VR2-2: Ensure communication of applications on different systems (i.e., IEDs),
- VR2-3: Ensure security (especially confidentiality and integrity) of external communication,
- VR2-4: Ensure authentication methods and role-based rights management,
- VR2-5: Ensure interoperability with legacy devices (e.g., over Modbus),
- VR2-6: Ensure the seamless cooperation with high level engineering concepts, and
- VR2-7: Ensure flexibility and configurability of the communication subsystem.

The requirements VR2-3 and VR2-4 show that the request for integrating security means has been fulfilled appropriately. The approach here was to integrate security already by design, as described in Reference [5], to allow for the consideration of security related functionalities in the engineering process. As will be demonstrated, an appropriate realisation thereof has been developed as well.

Configuration of all communication related parameters (VR2-7) had to be done manually throughout the proof-of-concept validation. This concerns all sub-scenarios listed below. For obtaining the necessary test results, this is sufficient. For commercial solutions, a more practicable way of configuring the communication subsystem should of course be chosen.

The goal of this validation scenario is to demonstrate the functionality of the modular communication system which has been previously specified and realised. This necessitates a series of tests in order to involve all relevant components of the SmartOS. An overview of the involved components is given in Figure 14, where directly involved components are coloured orange, and indirectly involved components are coloured light orange.



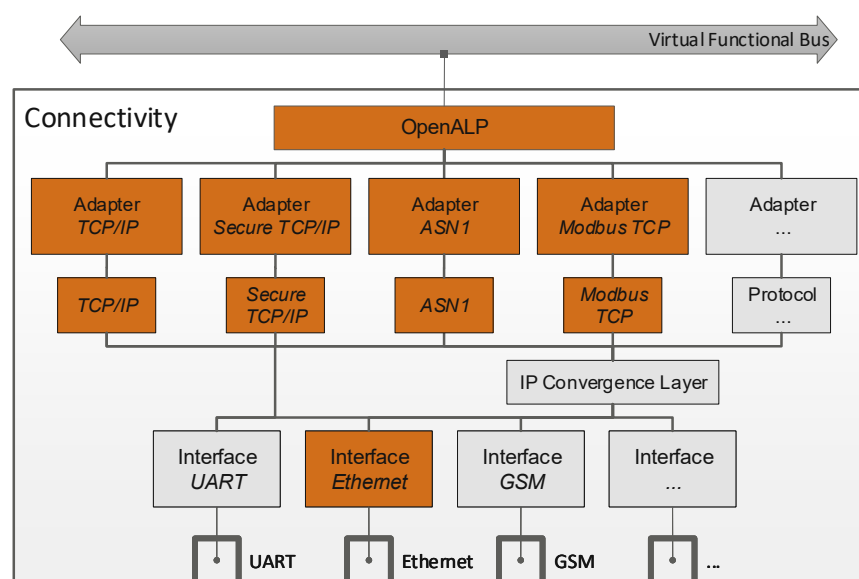
**Figure 14.** SmartOS components validated in scenario “Generic and Interoperable Communication” (adopted from Reference [4]).

Furthermore, an overview of the tested adapters and protocols, which are implemented as a part of the connectivity module, is provided in Figure 15. In order to allow incoming traffic for TCP/IP, secure TCP/IP and abstract syntax notation one (ASN.1), server modules are used. For Modbus, this has been omitted because the SmartOS only acts as a Modbus master in the conducted tests.

For this scenario, a series of increasingly complex tests has been devised, which have been designed to gradually increase and/or broaden the amount of components involved with each successive test. This keeps the amount of (added) complexity for each test case manageable, yet it allows for a systematic test of the entire communication system and a demonstration of its generic and interoperable nature. The scenarios are as follows:

- Two software components communicating locally: This minimal test demonstrates the basic functionality of the VFB and OpenALP, as well as the security and registry subsystems.
- Two software components on separate IEDs communicating over TCP/IP: Extends the previous scenario to include the connectivity module and its ancillary TCP/IP adapter. In addition, the communication pattern is changed from a push message to a request and the accompanying response.
- Two software components on separate IEDs communicating over ASN.1: This shows the transparency and interchangeability of the underlying protocol.

- (d) Two software components on separate IEDs communicating over secure TCP/IP: This extends the scope of scenarios b) and c) to include encryption and authentication (the latter utilising the trust centre). Furthermore, it can be shown that certificate revocation prevents a device from connecting to other devices.
- (e) A software component communicating with a legacy device via Modbus/TCP: Further demonstrates the transparency and interchangeability of the underlying protocol, as well as the support for legacy devices.



**Figure 15.** Connectivity module parts tested in scenario “Generic and Interoperable Communication” (adopted from Reference [4]).

The environment for these scenarios comprises a series of tests spanning one or more projects in the Eclipse integrated development environment (IDE). It is complemented by VPN access to the virtual testing environment, which hosts required services, such as the credential store. For test (e), a software-based Modbus slave has been created containing a process image from which data can be read and to which data can be written by a software component. Tests with a physical Modbus slave device have also been successfully performed and have yielded similar results. For test (a), a JUnit test invokes two software components (VFBAppTalk1 and VFBAppTalk2) running on the same IED. Upon initialisation, VFBAppTalk1 sends a push message to VFBAppTalk2. The successful completion of the test is confirmed via JUnit and console output, as can be seen in Figure 16.

For test (b), a JUnit test invokes 2 software components (VFBAppTalk3 and VFBAppTalk4) on 2 logically separate IEDs (hosted physically on the same machine). Upon initialisation, VFBAppTalk3 sends a request message (a simulated meter reading) to VFBAppTalk4, which then returns the appropriate response (a random integer). The successful completion of the test is confirmed via JUnit and console output, as can be seen in Figure 17.

```

16 public void test2AppsTalkingSameDevice ()
<terminated> Rerun at fhsalzburg.its.opennes.integration.Device2AppsTalking.test2AppsTalkingSameDevice [JUnit]
Sample Software Component VFBAAppTalk1 initialized
Sample Software Component VFBAAppTalk2 initialized
at.fhsalzburg.its.opennes.servers.TCPIPServer waiting for request on port 8080...
at.fhsalzburg.its.opennes.servers.ASN1Server waiting for request on port 8081...
at.fhsalzburg.its.opennes.servers.SecureTCPIPServer waiting for request on port 8443...
Sample Software Component sending message:
Recipient: IED0/at.freenergy.devices.apps.VFBAAppTalk2$DataPoint1
Message content: Shutdown
Communication type: PUSH
Transaction ID: 1

Sample Software Component received message:
Sender: IED0/at.freenergy.devices.apps.VFBAAppTalk1
Transaction ID: 1
Message content: Shutdown
Communication type: PUSH
Sample Software Component received message:
Sender: IED0/at.freenergy.devices.apps.VFBAAppTalk2
Transaction ID: 2
Message content: Shutdown Talk2 added Text
Communication type: PUSH

```

Figure 16. Test of two software components communicating locally.

```

44 public void test2AppsTalkingDifferentDevice() throws InterruptedException
45     final Device IED0 = new Device(new Registry("apps2Test/DeviceIED0/data
46     final Device IED1 = new Device(new Registry("apps2Test/DeviceIED1/data
47
48     (new Thread(new Runnable()
49     {
50     @Override
<terminated> Rerun at fhsalzburg.its.opennes.integration.Device2AppsTalking.test2AppsTalkingDifferentDevice [JUnit]
at.freenergy.devices.apps.VFBAAppTalk4 Sample Software Component initialized
at.fhsalzburg.its.opennes.servers.TCPIPServer waiting for request on port 8081...
at.fhsalzburg.its.opennes.servers.ASN1Server waiting for request on port 9091...
Sample Software Component VFBAAppTalk3 initialized
at.fhsalzburg.its.opennes.servers.TCPIPServer waiting for request on port 8080...
at.fhsalzburg.its.opennes.servers.ASN1Server waiting for request on port 9090...
at.freenergy.devices.apps.VFBAAppTalk3 Sample Software Component sending message:
Recipient: IED1/at.freenergy.devices.apps.VFBAAppTalk4$DataPoint1
Transaction ID: 1
Message content: GET METERREADING
Communication type: REQUEST

at.fhsalzburg.its.opennes.servers.TCPIPServer accepted request at: 58504
at.fhsalzburg.its.opennes.servers.TCPIPServer waiting for request on port 8081...
at.freenergy.devices.apps.VFBAAppTalk4 Sample Software Component received message:
Sender: IED0/at.freenergy.devices.apps.VFBAAppTalk3
Transaction ID: 2
Message content: GET METERREADING
Communication type: REQUEST
Endpoint for ID: 2
at.fhsalzburg.its.opennes.servers.TCPIPServer accepted request at: 58505
at.fhsalzburg.its.opennes.servers.TCPIPServer waiting for request on port 8080...
at.freenergy.devices.apps.VFBAAppTalk3 Sample Software Component received message:
Sender: IED1/at.freenergy.devices.apps.VFBAAppTalk4
Transaction ID: 1
Message content: 301763374kWh
Communication type: RESPONSE

```

Figure 17. Test of two software components communicating remotely.

Test (c) uses an identical setup to test (b), but with a modified device configuration which prioritises ASN.1 instead of TCP/IP. The modified mapping tables are provided in Figures 18 and 19 and the successful completion of the test is again confirmed via JUnit and console output.

openesaddress	priority	adaptername	adapteraddress	datatype	protocol
IED0/at.freenergy.devices.apps.VFBAppTalk3\$DataPoint1	2	SecureTCPIP	localhost,8443	java.lang.String	VFB
IED1/at.freenergy.devices.apps.VFBAppTalk4\$DataPoint1	2	SecureTCPIP	localhost,9443	java.lang.String	VFB
IED0/at.freenergy.devices.apps.VFBAppTalk3\$DataPoint1	1	ASN1	localhost,9090	java.lang.String	ASN1
IED1/at.freenergy.devices.apps.VFBAppTalk4\$DataPoint1	1	ASN1	localhost,9091	java.lang.String	ASN1

Figure 18. Address mapping on intelligent electronic device (IED)0 during Test (c).

openesaddress	priority	adaptername	adapteraddress	datatype	protocol
IED0/at.freenergy.devices.apps.VFBAppTalk3\$DataPoint1	2	SecureTCPIP	localhost,8443	java.lang.String	VFB
IED1/at.freenergy.devices.apps.VFBAppTalk4\$DataPoint1	2	SecureTCPIP	localhost,9443	java.lang.String	VFB
IED0/at.freenergy.devices.apps.VFBAppTalk3	2	SecureTCPIP	localhost,8443	java.lang.String	VFB
IED1/at.freenergy.devices.apps.VFBAppTalk4	2	SecureTCPIP	localhost,9443	java.lang.String	VFB
IED0/at.freenergy.devices.apps.VFBAppTalk3\$DataPoint1	1	ASN1	localhost,9090	java.lang.String	ASN1
IED1/at.freenergy.devices.apps.VFBAppTalk4\$DataPoint1	1	ASN1	localhost,9091	java.lang.String	ASN1
IED0/at.freenergy.devices.apps.VFBAppTalk3	1	ASN1	localhost,9090	java.lang.String	ASN1
IED1/at.freenergy.devices.apps.VFBAppTalk4	1	ASN1	localhost,9091	java.lang.String	ASN1

Figure 19. Address mapping on IED1 during Test (c).

Test (d) again uses an identical setup to tests (b) and (c) and the device configuration has again been modified to prioritise secure TCP/IP. First, valid certificates are used and the test is successfully completed. Thereafter, the certificate used on IED0 is revoked on the trust centre. As a result, the incoming request is refused by IED1, and the unit test fails (see Figure 20), which, in this case, is the expected and desired result.

```

46     final Device IED0 = new Device(new Registry("apps2Test/DeviceIED0/data/regi
47     final Device IED1 = new Device(new Registry("apps2Test/DeviceIED1/data/regi
48
49     (new Thread(new Runnable()
50     {
51     @Override
52     public void run() { IED1.startDevice(); }
53     }).start();

```

```

<terminated> Rerun at fhسالzburg.its.opennes.integration.Device2AppsTalking.test2AppsTalkingDifferentDevice [JUnit]
at.freenergy.devices.apps.VFBAppTalk4 $ Sample Software Component initialized
at.fhsالzburg.its.opennes.servers.ASN1Server waiting for request on port 9091...
at.fhsالzburg.its.opennes.servers.SecureTCPIPServer waiting for request on port 9443...
Sample Software Component VFBAppTalk3 initialized
at.fhsالzburg.its.opennes.servers.ASN1Server waiting for request on port 9090...
at.fhsالzburg.its.opennes.servers.SecureTCPIPServer waiting for request on port 8443...
at.freenergy.devices.apps.VFBAppTalk3 $ Sample Software Component sending message:
Recipient: IED1/at.freenergy.devices.apps.VFBAppTalk4$DataPoint1
Transaction ID: 1
Message content: GET METERREADING
Communication type: REQUEST

at.fhsالzburg.its.opennes.servers.SecureTCPIPServer accepted request at: 58374
Connection refused: Invalid certificate.
at.fhsالzburg.its.opennes.servers.SecureTCPIPServer waiting for request on port 9443...

```

```

Failure Trace
java.lang.AssertionError: expected: <1
at at.fhsالzburg.its.opennes.integrati

```

Figure 20. Refused request due to invalid certificate.

For test (e), the Modbus library Modbus4J has been utilised to provide a Modbus slave containing a process image from which data can be read and to which data can be written. Thereafter, a software component (SimpleVFBApp3) has been created to send a series of messages to the Modbus slave, and the device registry on the IED running SimpleVFBApp3 has been configured to provide the address and data point of the Modbus slave as IED4/sampleNamespace\$SampleModbusValue. Upon initialisation of the IED, SimpleVFBApp3 is invoked and sequentially performs three tasks: First, a request message is sent to read the value of the aforementioned data point. Second, a push message is sent to set a different value for this data point. Last, a request message is sent once again to confirm that the value has indeed been set. The successful completion of the test is again confirmed via JUnit and console output.

Summarising this scenario, it can be said that all tests could be completed successfully, and the goals regarding generic and interoperable communication have been achieved. Although the tests have certainly not been as rigorous as would be required for a market-ready product, they are sufficient to validate the effectiveness of the solutions that have been developed for the critical challenges. Further work towards the creation of a final product would predominantly involve fine-tuning and optimising and is not expected to encounter any major challenges for which no well-established solutions exist.

### 6.3. System Integration

The requirements for the system integration test cover in principle all the above outlined requirements. However, especially for the integration testing of the proof-of-concept validation, the following additional requirements have been identified:

- VR3-1: Ensure the safe and secure execution of DER services, and
- VR3-2: Ensure the safe and secure execution of energy/DER applications.

The previous scenarios all resulted in successful validations of the main parts of the proposed approach: (i) the formal application modelling concept (see Reference [7]), (ii) the remote programmable DER services, and (iii) the generic and interoperable communication. With these validations as a basis, the final scenario shows a system integration validation. In other words, the main intention is to use the application modelling approach to create and use remote DER functions with the help of the generic and modular communication system.

This scenario is also intended to be a validation of the overall idea of the approach, as shown in Figure 4 and as it has also been used in Reference [7] for the validation of the application modelling concept. It is assumed that the overall application has been modelled using the formal modelling approach. Furthermore, it is assumed that the remote programmable DER services are available and that the other components of the SmartOS are operational. The last precondition is, that it is possible to communicate between DERs using the Connectivity component (i.e., the generic and modular communication concept tested in Section 6.2). Consequently, what is left to validate is the integration of these parts with each other. First, it has to be shown, how an application can be deployed, configured, and started on a DER device, once it has been modelled. Second, it has to be shown, how the communication between different devices can be implemented and properly configured.

For this validation, the following test case is used: A voltage change in the grid of a distribution system operator (DSO) causes the voltage spread to increase above its allowed threshold. This causes the control algorithm in VoltVarController seen in Figure 21 to calculate a new Volt-var droop curve for the DERController. With the new curve, the DERController will be more sensitive to voltage deviations. This, in turn, will decrease the voltage spread.

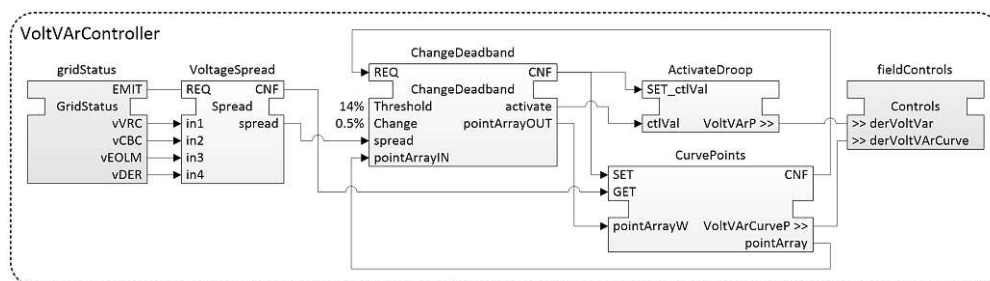


Figure 21. Implementation of the VoltVarController application in IEC 61499 [7].

Compared to the previous scenarios, this test case shows how the envisioned method is used to deploy an application to real devices. Furthermore, it also includes communication

between multiple devices using different communication protocols, which need to be configured. Finally, the goal is also to validate the implementation of the VoltVArController function as depicted in Figure 21.

A scheme of the used parts of the AIT SmartEST lab setup is illustrated in Figure 22. It also shows on which components the different IEC 61499 functions are deployed. The VoltVArController function, as shown in Figure 21, is deployed on the DSOComputer.

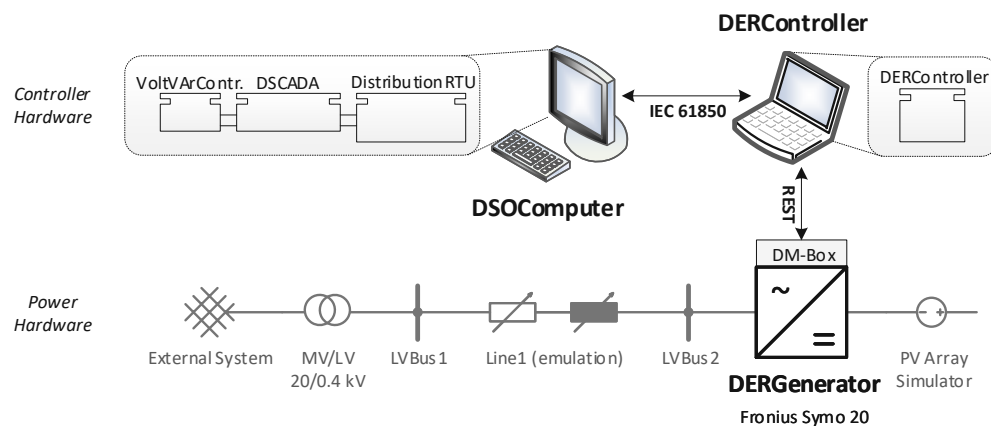
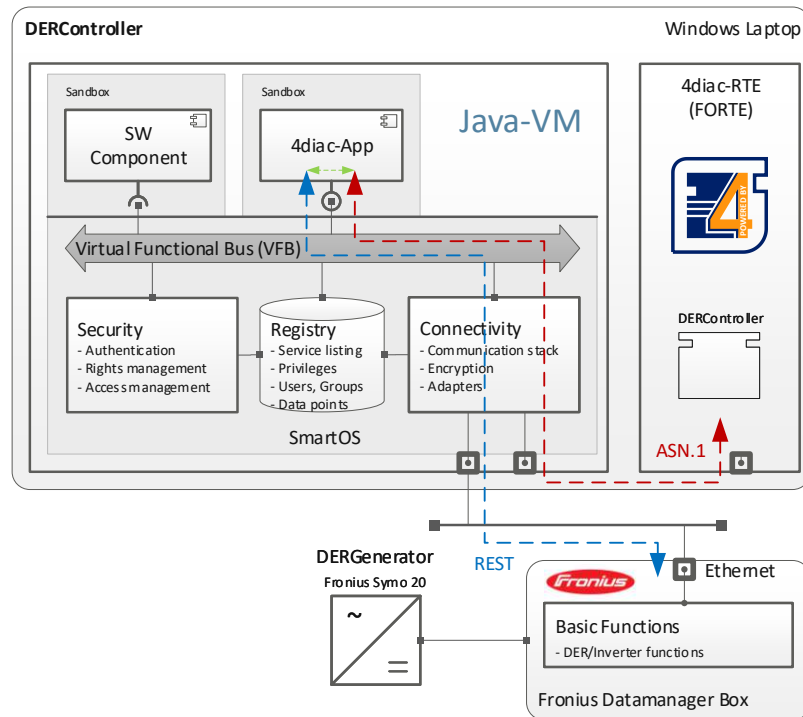


Figure 22. Laboratory setup for the validation use case (adopted from Reference [7]).

The controller hardware used in the laboratory consists of a desktop PC representing the DSOComputer and a laptop PC running the SmartOS and the DERController App. The DERGenerator is a commercial off-the-shelf Fronius Symo 20 PV inverter; on the DC side, it is connected to a PV array simulator, and, on the AC side, to the LV grid. A line impedance had to be emulated by the laboratory equipment in order to create a dependency between the voltage and the inverter output. Natively, the inverter offers a Modbus/TCP control and measurement interface. However, for this test, a representational state transfer (REST) interface was added to facilitate communication between the inverter and the SmartOS App. On the laptop, the SmartOS was executed together with 4diac [60]. The setup of the SmartOS is shown in Figure 23.

In order to use the programmability of 4diac, it was integrated with the VFB of the SmartOS. First, an ASN.1 adapter was created for the Connectivity component. This adapter allows communication between software components and the 4diac runtime environment (FORTE). This communication is shown in Figure 23 as a dashed red line. However, in order to change the reactive power, a command must be sent to the basic functions component running on the Fronius DM-Box. For this, the next step was to create a dedicated software component that forwards messages from the FORTE to the DM-Box. This means that, if the DERController function in the FORTE wants to send a new reactive power setpoint to the inverter, it is first sent to the 4diac-App using ASN.1 prior to being forwarded to the basic functions component using the REST protocol (represented as a dashed blue line in Figure 23). Finally, the DM-Box makes sure that the reactive power setting is applied by the inverter-based DER.

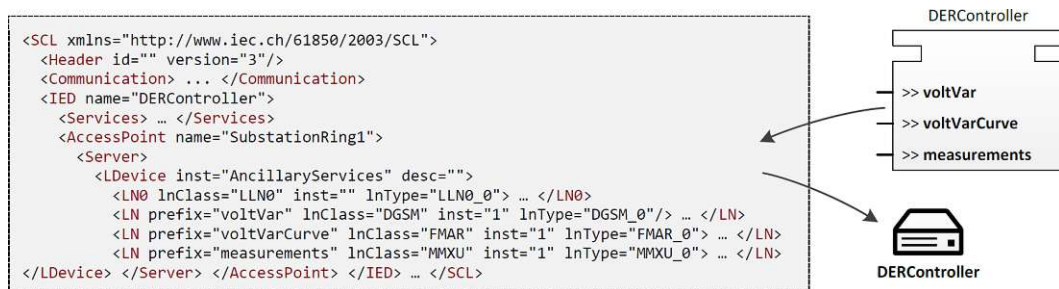




**Figure 23.** Setup of the SmartOS and 4diac on the distributed energy resources (DER) Controller laptop.

Before the laboratory validation could be started, the application had to be deployed. The first precondition for this is that the inverter is connected and feeding power to the grid. Second, the FORTE must be executed on all the components and awaiting a deployment from the 4diac-IDE. The next step was to deploy the communication configurations.

Next, the communication infrastructure was generated for connections between IEC 61499 functions running on different devices. Generated were both communication FBs, as well as communication configurations. For this test case, all the interfaces are derived from IEC 61850, which results in generation of substation configuration description language (SCL) files for the communication configuration. The resulting IEC 61850 configuration for the DERController is seen in Figure 24. As is seen, the IEC 61850 SCL configuration adopts information from the IEC 61499 application (e.g., the IEC 61850 IED is named DERController after the IEC 61499 device).



**Figure 24.** Generated substation configuration description language (SCL) file for configuration of the IEC 61850 server in DER Controller [7].

After this, the **SCL** files and the IEC 61499 applications are deployed to their components according to Figure 22. This is a built-in feature of 4diac and utilises standard IEC 61499 methods; thus, no new code has to be generated from the IEC 61499 model. After deployment completion, the application is automatically started; hereby, the **SCL** files are loaded and the IEC 61850 communication is initialised. For example, the IEC 61850 client in DistributionRTU connects to the IEC 61850 server on the **DERController**.

Subsequently, the VoltVArController algorithm can be validated. For this validation, the inverter is configured to an active power output of 18 kW and a reactive power output of 0 kVAr. The voltage measured by the inverter (i.e., the **DERGenerator**) is forwarded to the VoltVArController at the **vDER** data output of the gridStatus **FB** (see Figure 21). The other voltage measurements (i.e., **vVRC**, **vCBC**, and **cEOLM**) are all emulated and fixed to 0.9 per unit (p.u.). After stabilisation, this results in a **vDER** voltage of around 1.004 p.u., and no extra reactive power. This is seen at the beginning of the time series in the top graph of Figure 25, where **Q** is the reactive power of the inverter, and **U** is the measured voltage by the inverter (i.e., **vDER**).

An increased voltage spread was emulated to trigger the VoltVArController algorithm, seen as the first event (i) in Figure 25. At this event, the fixed voltage of **vVRC** was changed from 0.9 p.u. to 0.86 p.u. This increases the voltage spread ( $U_{\max} - U_{\min}$ , see Figure 25) above the allowed threshold. This is detected by the VoltVArController's algorithm. New Volt/var curve parameters are calculated by the ChangeDeadband **FB** in Figure 21. The new curve parameters are sent to the **DERController**, which results in a new reactive power set point for the **DERGenerator**. With the new Volt/var parameters, the inverter starts producing reactive power. This is shown as event (ii) in Figure 25. However, since the voltage spread is still too high, a second correction of the Volt/var parameters is performed by the VoltVArController after 185 s, which is visualised in Figure 25 as event (iii).

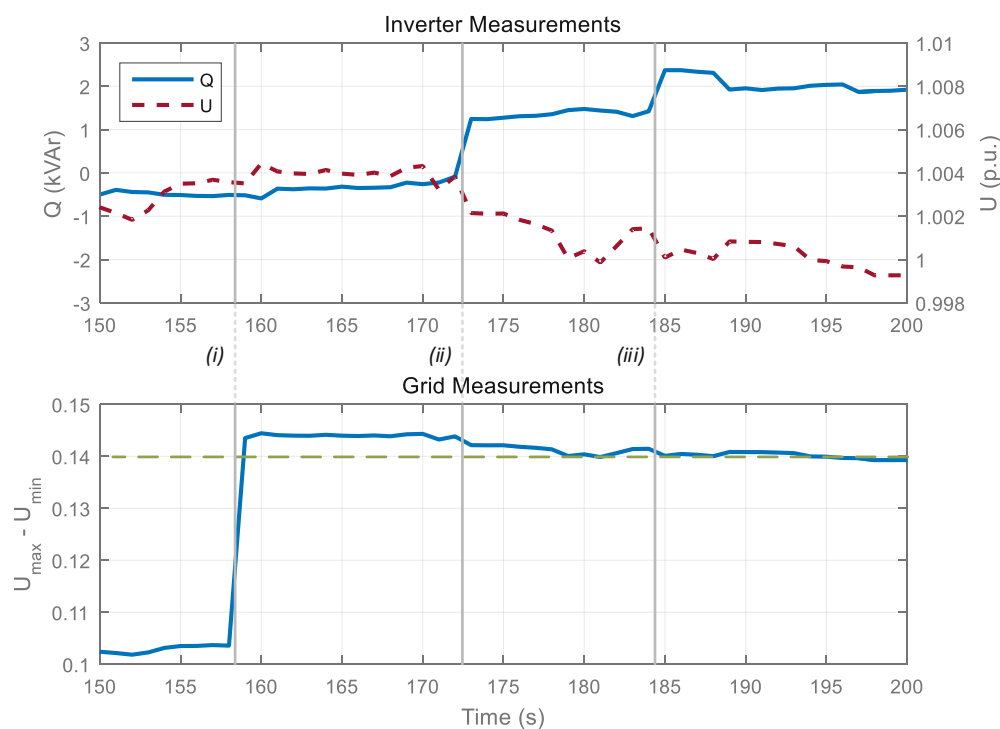


Figure 25. Measurements from the laboratory validation [7].

In general, the results of this experiment indicate a successful integration of the different parts of the proposed approach. This scenario shows that even with the prototypical implementation, it is possible to achieve an open and interoperable ICT solution for the integration of DERs. For productive use, many adjustments would of course have to be made to the setup shown in Figure 22 (such as running the SmartOS on the DM-Box and integrating the FORTE into the 4diac App).

Regarding the results of the test, especially the measurements shown in Figure 25, more evidence of the prototypical nature of the implementation is seen. In an industrial implementation, these changes would have to be in the range of milliseconds, whereas they are in the range of seconds in this case. This can be explained by the suboptimal test setup shown in Figure 22, as well as a complete absence of code optimisation in that regard. Nevertheless, the results demonstrate the basic usefulness of the presented concept. For industrial adoption, the reaction time has yet to be improved to the required level.

#### 6.4. Reflection of Results

The achieved results show that the proposed architecture helps to fill the gaps of current solutions and approaches, as identified in Section 2.4, in the following way:

- As outlined in Section 3.4 and further explained in Reference [7], this work provides an approach for a common application modelling concept for power and energy systems automation applications.
- By supporting a wide range of communication protocols, the developed architecture provides a unifying method of integrating renewable energy sources into smart grids. It allows to build open and scalable smart grid automation applications in a hardware and platform independent way.
- The introduction of RCSs makes it possible to reuse common functionalities and to update and extend DER services across the entire lifecycle of DER components.

The specific system performance and level of compliance to timing requirements will depend on code optimisation, as well as the underlying protocols. It will, therefore, be need to be analysed on a case-by-case basis, if given requirements can be met using given protocols.

For real-world solutions, safe handling of potential controller conflicts is an essential requirement. An approach to a potential solution is provided in Reference [61].

Although cyber-security issues have not been the primary focus of the work at hand, security-by-design principles as outlined in Reference [5] have been applied during the development of the architecture, resulting in a fundamentally robust system. As our solution aims for a maximum of protocol independence, the integration of a more comprehensive security architecture is easily possible if needed.

The proposed architecture and proof-of-concept prototype is currently being further developed into an industry grade solution. This entails, among other things, the addition of support for additional protocols and the realisation of a fully-fledged trust centre/credential store, as well as code optimisation. Prior to its roll-out and use in real-world scenarios, it will further have to undergo extensive testing (conformity, security/penetration, deployment and acceptance tests, etc.).

## 7. Conclusions

As a summary, it could be demonstrated that the presented approach of an open ICT infrastructure for the integration of renewable energy sources, especially DER, into a smart power grid may be used for producing feasible working prototypes. The presented solution is able to provide necessary means of communication to smart grid applications. The central aspects here include the definition of a unique addressing scheme, as well as the provision of an application layer protocol, for control services (OpenALP), which may again be used by smart grid applications. Hereby, legacy protocols are used as much as possible; only functionality necessary for the correct working of OpenALP is additionally provided in the form of protocol wrappers. Furthermore, a tool (PSAL) to facilitate the

engineering of remote programmable services to be used in the presented ecosystem has also been provided. To test the approach, the developed prototype has been validated in a number of concrete validation cases. It was not a goal of the validation to provide a fully-fledged communication middleware—thus, a prototype containing the minimum required functions has been realised.

As has been shown, with the open definition of the address scheme in combination with [OpenALP](#), it is easily possible for vendors of grid related middleware frameworks to integrate the demonstrated functionalities. Thus, the solution at hand does not intend to provide yet another framework but to suggest an open ecosystem which may be used by several vendors. Consequently, the realised testbed was used in order to prove the concept at hand, rather than to provide an already optimised solution. Nevertheless, the results demonstrate the feasibility of the approach, even when not optimised regarding performance. Thus, it can be expected that a fully engineered version of the presented functionalities would clearly contribute to an efficient and easy to use tooling for integrating renewables into smart grids, especially dedicated to [LV](#) grids. Even more, as the concept has been set up in a very generic way, without clear bindings on the power and energy systems domain, it can be expected that the presented solution may provide benefits to other sectors, as well, especially for those industry domains, where massively distributed control will play an important role in the near future.

**Funding:** This research work received funding by the Austrian Ministry for Transport, Innovation and Technology (bmvit) and the Austrian Research Promotion Agency (FFG) under the “ICT of the Future” program in the OpenNES project (FFG No. 845632). Contributions have also been performed within the project InterGrid, which is funded by the State of Upper Austria via the FFG under the contract number 881296.

**Author Contributions:** All authors jointly developed the overall approach, including the prototype implementation and the validation of it. Moreover, A.V. defined the overall paper structure and provided the main parts of Sections 1–4, as well as the abstract and the conclusion. O.L. wrote the main parts of Sections 5 and 6 and contributed to the readability of the entire paper. F.P.A. edited the description of the engineering part of the prototype, especially but not only, in Section 3. C.K. brought in his experience on practical uses of [ICT](#) infrastructures and integrated information on the relevant requirements. T.I.S. contributed to the overall structure, validated the content of the relevant sections, and provided a proof reading of the total paper. All authors have read and agreed to the published version of the manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

### Abbreviations

The following abbreviations are used in this manuscript:

AMI	advanced metering infrastructures
API	application programming interface
ARQ	automatic repeat request
ASN.1	abstract syntax notation one
CA	certificate authority
CAN	controller area network
CIM	common information model
CORBA	common object request broker architecture
COSEM	companion specification for energy metering
DDS	data distribution service
DER	distributed energy resources

DLMS	device language message specification
DM-Box	datamanager box
DSL	domain specific language
DSO	distribution system operator
EBNF	extended Backus Naur form
EJBCA	enterprise JavaBeans certificate authority
FB	function block
FIFO	first in first out
FORTE	4diac runtime environment
GWAC	GridWise architecture council
HA/BA	home and building automation
ICT	information and communication technology
IDE	integrated development environment
IEC	International Electrotechnical Commission
IED	intelligent electronic device
IMAP	Internet message access protocol
IoT	Internet of things
IP	Internet protocol
ISO	International Organisation for Standardisation
LD	logical device
LDAP	lightweight directory access protocol
LV	low voltage
MBSE	model-based system engineering
MDA	model-driven architecture
MIME	multipurpose Internet mail extensions
MOM	message oriented middleware
OPC UA	open process control unified architecture
OpenALP	open application layer protocol
OpenLDAP	open lightweight directory access protocol
OS	operating system
OSGi	Open Services Gateway initiative
OSI	open systems interconnection
PDP	policy decision point
PEP	policy enforcement point
PIM	platform independent model
PKI	public key infrastructure

PLC	powerline communication
POP3	post office protocol 3
PSAL	power system automation language
PV	photovoltaics
RBAC	role-based access control
RCS	remote controllable service
REST	representational state transfer
RTU	remote terminal unit
SAP	service access point
SCADA	supervisory control and data acquisition
SCL	substation configuration description language
SGAM	smart grid architecture model
SIFB	service interface function block
SIP	session initiation protocol
SMTP	simple mail transfer protocol
TCP	transmission control protocol
TSN	time sensitive network
UDP	user datagram protocol
UML	unified modelling language
VFB	virtual functional bus
VPN	virtually private network
VR	validation requirement
XMPP	extensible messaging and presence protocol

## References

1. The European Parliament and the Council of the European Union. Directive 2009/28/EC. Available online: <https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32009L0028> (accessed on 25 January 2021).
2. Kupzog, F.; Dimitriou, P.; Faschang, M.; Mosshammer, R.; Stifter, M.; Andr n, F. Co-Simulation of Power- and Communication-Networks for Low Voltage Smart Grid Control. In Proceedings of the 1st D-A-CH Energieinformatik 2012, Oldenburg, Germany, 5–6 July 2012.
3. Budka, K.C.; Deshpande, J.G.; Thottan, M. *Communication Networks for Smart Grids*; Springer: London, UK, 2016.
4. Pr stl Andr n, F.; Strasser, T.; Langthaler, O.; Veichtlbauer, A.; Kasberger, C.; Felbauer, G. Open and Interoperable ICT Solution for Integrating Distributed Energy Resources into Smart Grids. In Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Germany, 6–9 September 2016.
5. Veichtlbauer, A.; Langthaler, O.; Engel, D.; Kasberger, C.; Pr stl Andr n, F.; Strasser, T. Towards Applied Security-by-Design for DER Units. In Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Germany, 6–9 September 2016.
6. Veichtlbauer, A.; Parfant, M.; Langthaler, O.; Pr stl Andr n, F.; Strasser, T. Evaluating XMPP Communication in IEC 61499-based Distributed Energy Applications. In Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Germany, 6–9 September 2016.
7. Pr stl Andr n, F.; Strasser, T.I.; Kastner, W. Engineering Smart Grids: Applying Model-driven Development from Use Case Design to Deployment. *Energies* **2017**, *10*, 374.

8. Bründlinger, R.; Strasser, T.; Lauss, G.; Hoke, A.; Chakraborty, S.; Martin, G.; Kroposki, B.; Johnson, J.; de Jong, E. Lab Tests: Verifying That Smart Grid Power Converters Are Truly Smart. *IEEE Power Energy Mag.* **2015**, *13*, 30–42, doi:10.1109/MPE.2014.2379935.
9. Kupzog, F.; Veichtlbauer, A.; Heinisch, A.; von Tüllenbur, F.; Langthaler, O.; Pache, U.; Jung, O.; Frank, R.; Dorfinger, P. The Impact of Virtualisation Techniques on Power System Control Networks. *Electronics* **2020**, *9*, 1433–1454, doi:10.3390/electronics9091433.
10. SMB Smart Grid Strategic Group (SG3). *IEC Smart Grid Standardization Roadmap*; Technical Report; International Electrotechnical Commission (IEC): Geneva, Switzerland, 2010.
11. Huang, A.Q.; Crow, M.L.; Heydt, G.T.; Zheng, J.P.; Dale, S.J. The Future Renewable Electric Energy Delivery and Management (FREEDM) System: The Energy Internet. *Proc. IEEE* **2011**, *99*, 133–148, doi:10.1109/JPROC.2010.2081330.
12. IqtiyaniIham, N.; Hasanuzzaman, M.; Hosenuzzaman, M. European smart grid prospects, policies, and challenges. *Renew. Sustain. Energy Rev.* **2017**, *67*, 776–790.
13. Romero Aguero, J.; Khodaei, A. Grid Modernization, DER Integration & Utility Business Models—Trends Challenges. *IEEE Power Energy Mag.* **2018**, *16*, 112–121.
14. Rohjans, S.; Danekas, C.; Uslar, M. Requirements for Smart Grid ICT-architectures. In Proceedings of the 2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe), Berlin, Germany, 14–17 October 2012; pp. 1–8, doi:10.1109/ISGTEurope.2012.6465617.
15. Kanabar, M.G.; Voloh, I.; McGinn, D. Reviewing smart grid standards for protection, control, and monitoring applications. In Proceedings of the 2012 IEEE PES Innovative Smart Grid Technologies (ISGT), Washington, DC, USA, 16–20 January 2012; pp. 1–8.
16. Nafi, N.S.; Ahmed, K.; Gregory, M.A.; Datta, M. A survey of smart grid architectures, applications, benefits and standardization. *J. Netw. Comput. Appl.* **2016**, *76*, 23–36.
17. International Electrotechnical Commission. *IEC 61970-301: Energy Management System Application Program Interface (EMS-API)—Part 301*; IEC: Geneva, Switzerland, 2016.
18. International Electrotechnical Commission. *IEC 61850-1: Communication Networks and Systems for Power Utility Automation—Part 1*; IEC: Geneva, Switzerland, 2013.
19. Gungor, V.C.; Sahin, D.; Kocak, T.; Ergut, S.; Buccella, C.; Cecati, C.; Hancke, G.P. Smart Grid Technologies: Communication Technologies and Standards. *IEEE Trans. Ind. Inform.* **2011**, *7*, 529–539, doi:10.1109/TII.2011.2166794.
20. IEC Central Office. *The Smart Grid Standards Map*; IEC: Geneva, Switzerland, 2019.
21. Institute of Electrical and Electronics Engineers (IEEE). *IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads*; Institute of Electrical and Electronics Engineers (IEEE): New York, NY, USA, 2011.
22. Smart Grid Coordination Group. *Smart Grid Reference Architecture*; Technical Report; CEN/Cenelec/ETSI Smart Grid Coordination Group: Brussels, Belgium, 2012.
23. The GridWise Architecture Council. *GridWise Interoperability Context-Setting Framework*; Technical Report; The GridWise Architecture Council: Richland, WA, USA, 2008.
24. International Organization for Standardization. *ISO/IEC 7498-1:1994 Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model*; ISO: Geneva, Switzerland, 1994.
25. International Electrotechnical Commission. *IEC 61131-3: Programmable Controllers—Part 3: Programming Languages*; IEC: Geneva, Switzerland, 2012.
26. International Electrotechnical Commission. *IEC 61499: Function Blocks*; IEC: Geneva, Switzerland, 2012.
27. Strasser, T.; Zoitl, A.; Christensen, J.; Sünder, C. Design and Execution Issues in IEC 61499 Distributed Automation and Control Systems. *IEEE Trans. Syst. Man, Cybern. Part C Appl. Rev.* **2010**, *41*, 41–51, doi:10.1109/TSMCC.2010.2067210.
28. Veichtlbauer, A.; Pfeiffenberger, T. Generic Middleware for User-friendly Control Systems in Home and Building Automation. *Int. J. Adv. Networks Serv.* **2013**, *6*, 51–67.
29. Fraunhofer. *OGEMA: Open Gateway Energy Management*; Technical Report; Fraunhofer IWES: Bremerhaven, Germany, 2012.
30. Broy, M.; Gleirscher, M.; Merenda, S.; Wild, D.; Kluge, P.; Krenzer, W. Toward a Holistic and Standardized Automotive Architecture Description. *Computer* **2009**, *42*, 98–101, doi:10.1109/MC.2009.413.
31. International Electrotechnical Commission. *IEC 62056: Electricity Metering Data Exchange—The DLMS/COSEM Suite*; IEC: Geneva, Switzerland, 2014.
32. Veichtlbauer, A.; Engel, D.; Knirsch, F.; Langthaler, O.; Moser, F. Advanced Metering and Data Access Infrastructures in Smart Grid Environments. In Proceedings of the 7th International Conference on Sensor Technologies and Applications (SensorComm 2013), Barcelona, Spain, 25–31 August 2013.
33. The openHAB Community and the openHAB Foundation e.V. *Welcome to openHAB*; Technical Report; The openHAB Community and the openHAB Foundation e.V.: Ober-Ramstadt, Germany, 2019.
34. The OSGi Alliance. *OSGi Core Release 5*; OSGi Alliance: San Ramon, CA, USA, 2012.
35. Pichler, M.; Veichtlbauer, A.; Engel, D. Evaluation of OSGi-based Architectures for Customer Energy Management Systems. In Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT2015), Seville, Spain, 17–19 March 2015.
36. Zeadally, S.; Kubher, P. Internet access to heterogeneous home area network devices with an OSGi-based residential gateway. *Int. J. Ad Hoc Ubiquitous Comput.* **2008**, *3*, 48–56.

37. Object Management Group. *Common Object Request Broker Architecture (CORBA) Specification*; Technical Report; Object Management Group: Milford, MA, USA, 2015.
38. Object Management Group. *Data Distribution Service (DDS)*; Technical Report; Object Management Group: Milford, MA, USA, 2015.
39. OPC Foundation. *OPC—The Interoperability Standard for Industrial Automation & Other*; OPC Foundation: Scottsdale, AZ, USA, 2012.
40. Sermersheim, J. Lightweight Directory Access Protocol (LDAP): The Protocol, RFC4511. Available online: <https://www.rfc-editor.org/info/rfc4511> (accessed on 19 February 2021).
41. Freed, N.; Borenstein, N. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, RFC 2045. Available online: <https://www.rfc-editor.org/info/rfc2045> (accessed on 19 February 2021).
42. Desruisseaux, B. Internet Calendaring and Scheduling Core Object Specification (iCalendar), RFC5545. Available online: <https://www.rfc-editor.org/info/rfc5545> (accessed on 19 February 2021).
43. International Electrotechnical Commission. *IEC 60870-1: Telecontrol Equipment and Systems—Part 1: General Considerations*; IEC: Geneva, Switzerland, 1988.
44. Rohjans, S. (S2)In—Semantic Service Integration for Smart Grids. Ph.D. Thesis, Carl von Ossietzky University, Oldenburg, Germany, 2012.
45. Alkhawaja, A.R.; Ferreira, L.L.; Albano, M. Message Oriented Middleware with QoS Support for Smart Grids. In Proceedings of the INForum 2012 Conference on Embedded Systems and Real Time, Caparica, Portugal, 6–7 October 2012.
46. Rosenberg, J.; Schulzrinne, H.; Camarillo, G.; Johnston, A.; Peterson, J.; Sparks, R.; Handley, M.; Schooler, E. SIP: Session Initiation Protocol, RFC3261. Available online: <https://www.rfc-editor.org/info/rfc3261> (accessed on 19 February 2021)
47. Saint-Andre, P. Extensible Messaging and Presence Protocol (XMPP): Core, RFC6120. Available online: <https://www.rfc-editor.org/info/rfc6120> (accessed on 19 February 2021).
48. Postel, J. User Datagram Protocol, RFC 768. Available online: <https://www.rfc-editor.org/info/rfc768> (accessed on 19 February 2021).
49. Postel, J. Transmission Control Protocol—DARPA Internet Program Protocol Specification, RFC 793. Available online: <https://www.rfc-editor.org/info/rfc793> (accessed on 19 February 2021).
50. Deering, S.; Hinden, R. Internet Protocol, Version 6 (IPv6) Specification, RFC8200. Available online: <https://www.rfc-editor.org/info/rfc8200> (accessed on 19 February 2021).
51. Postel, J. Internet Protocol—DARPA Internet Program Protocol Specification, RFC 791. Available online: <https://www.rfc-editor.org/info/rfc791> (accessed on 19 February 2021).
52. Morello, R.; Capua, C.D.; Fulco, G.; Mukhopadhyay, S.C. A Smart Power Meter to Monitor Energy Flow in Smart Grids: The Role of Advanced Sensing and IoT in the Electric Grid of the Future. *IEEE Sens. J.* **2017**, *17*, 7828–7837.
53. Shelby, Z.; Bormann, C. *6LoWPAN: The Wireless Embedded Internet*; John Wiley & Sons: Hoboken, NJ, USA, 2011; Volume 43.
54. Schneider Automation. *MODBUS Messaging on TCP/IP Implementation Guide V1.0b*; Technical Report; Modbus Organization, Inc.: Hopkinton, MA, USA, 2006.
55. Ethernet POWERLINK Standardisation Group. *Ethernet POWERLINK Communication Profile Specification DS301*; EPSG: Fredersdorf, Germany, 2016.
56. International Electrotechnical Commission. *IEC PAS62559: IntelliGrid Methodology for Developing Requirements for Energy Systems*; Technical Report; International Electrotechnical Commission (IEC): Geneva, Switzerland, 2008.
57. Gottschalk, M.; Göring, A.; Uslar, M. Applying the Use Case Methodology to Smart Cities. In Proceedings of the VDE-Kongress, Frankfurt am Main, Germany, 20–21 October 2014.
58. Smart Grid Coordination Group. *Sustainable Processes*; Technical Report; CEN/Cenelec/ETSI Smart Grid Coordination Group: Brussels, Belgium, 2012.
59. Strasser, T.; Lauss, G.; Andren, F.; Stifter, M.; Bründlinger, R.; Fechner, H.; Knöbl, K. Smart Grid Research Infrastructures in Austria—Examples of available laboratories and their possibilities. In Proceedings of the 13th IEEE International Conference on Industrial Informatics (INDIN 2015), Cambridge, UK, 22–24 July 2015; pp. 1539–1545.
60. Zoitl, A.; Strasser, T.; Valentini, A. Open source initiatives as basis for the establishment of new technologies in industrial automation: 4DIAC a case study. In Proceedings of the 2010 IEEE International Symposium on Industrial Electronics (ISIE), Bari, Italy, 4–7 July 2010.
61. Znanabria, C.; Tayyebi, A.; Pröbstl Andrén, F.; Kathan, J.; Strasser, T. Engineering support for handling controller conflicts in energy storage systems applications. *Energies* **2017**, *10*, 1595.



## 2.2 Publication 2

A. Veichtlbauer, A. Heinisch, F. v. Tüllenbug, P. Dorfinger, O. Langthaler, U. Pache:  
**Smart Grid Virtualisation for Grid-Based Routing.**

*In: Electronics, Vol. 9, No. 11, pp. 1879, Nov. 2020*


### Own Contribution

Also for this paper, the candidate defined the overall paper structure. He furthermore acted as corresponding author and provided proof-reading and conceptual checks for the whole paper. He provided the main parts for the virtualisation concept and the communication subsystem. He also contributed to the evaluation of the prototypical realisation and the derived conclusions and gave feedback on the use case definition.



Article

## Smart Grid Virtualisation for Grid-Based Routing

Armin Veichtlbauer <sup>1,\*</sup> , Alexander Heinisch <sup>2</sup>, Ferdinand von Tüllenbaur <sup>3</sup>, Peter Dorfinger <sup>3</sup>, Oliver Langthaler <sup>4</sup> and Ulrich Pache <sup>4</sup>

<sup>1</sup> Campus Hagenberg, University of Applied Sciences Upper Austria, 4232 Hagenberg, Austria

<sup>2</sup> Corporate Technology, Siemens AG, 1210 Vienna, Austria; alexander.heinisch@siemens.com

<sup>3</sup> Advanced Networking Center, Salzburg Research Forschungsg.m.b.H., 5020 Salzburg, Austria; ferdinand.tuellenbaur@salzburgresearch.at (F.v.T.); peter.dorfinger@salzburgresearch.at (P.D.)

<sup>4</sup> Center for Secure Energy Informatics, University of Applied Sciences Salzburg, 5412 Puch/Salzburg, Austria; oliver.langthaler@fh-salzburg.ac.at (O.L.); ulrich.pache@fh-salzburg.ac.at (U.P.)

\* Correspondence: armin.veichtlbauer@fh-hagenberg.at; Tel.: +43-50-804-22825

Received: 19 August 2020; Accepted: 31 October 2020; Published: 8 November 2020



**Abstract:** Due to changed power consumption patterns, technological advance and deregulation, the appearance of the power grid in the low and medium voltage segment has changed. The spread of heating and cooling with electrical energy and an increase of electric vehicles as well as the broad rollout of photovoltaic systems has a major impact on the peak power demand of modern households and the volatility smart grids have to face. Thus, besides the load impact of the growing population of electric vehicles, modern households are not only consumers of electrical power, but also power producers, so called prosumers. The rising number of prosumers and the limitations of grid capacities lead to an increasingly distributed system of heterogeneous components, which have to be managed and operated with locality and scalability in mind. Virtualisation technologies, particularly known as state of the art in data centre computing, can lead to a paradigm shift needed to meet the growing demands of this evolution. A key issue here is to forward data to the correct data sinks, where data are required in order to keep the grid balanced. This routing process has to be able to react on grid changes in a timely manner, i.e., it must be based on the instantaneous state of the grid. In this paper, we propose a solution based on virtualising the communication infrastructure in the low and medium voltage grid. We evaluate two different approaches. The first approach is based on SDN; an ONOS SDN controller is used to change the behaviour of the communication infrastructure according to information provided by components of the power grid. The second approach uses Coaty and a Mosquitto MQTT broker to deliver messages to the desired endpoint, again based on information from the power grid.

**Keywords:** smart grid; communication; virtualisation; application layer routing; SDN; MQTT; Coaty

### 1. Introduction

Smart grids rely on at least two kinds of networks: on one hand, the power grid which is used to transfer the energy from producers to consumers, and, on the other hand, the communication network used to transmit data between the various participants within the smart grids. These participants include tap changers, smart circuit breakers, e-car-charging stations, smart buildings, virtual power plants, just to name a few. Additionally, there are utilities which are not directly related to the power grid augment and extend the information available from the grid. Since the number of sources and sinks of information, as well as the interest in or legal restrictions on certain parts of the information, is highly heterogeneous, the management and deployment of components in a smart grid can be quite challenging.

New smart buildings, virtual plants, wind parks and e-car charging stations are commissioned every day, not only participating in but also influencing the distribution network of a smart grid. In case new participants in the grid are connected, or existing ones are disconnected from grid segments, the communication infrastructure has to be reconfigured to guarantee the correct information flow to operate the smart grid segments correctly. The data relevant to a grid node can be determined by the instantaneous state of the grid. Since supervisory control and data acquisition (SCADA) systems do not scale very well to the proposed number of participants in the power grid, the use of such systems to manage the data centrally is not an option. On the other hand, managing this (re-)configuration in a distributed system using traditional networking equipment is complex, time-consuming and prone to errors.

The distribution of control signals in a smart grid is a very critical task. The validity and correctness of such signals depend not only on the correct values but also demand strong guarantees on data being redirected to the desired endpoints, the timeliness of the delivery and much more. This paper sets its scope on the delivery of monitoring data. Nevertheless, the provided proof of concept implementation can also handle control signals, albeit only with a very reduced set of guarantees. As monitoring data are collected from very distributed locations, and has potential relevance to again distributed data processing units, our approach allows communication to take place in a one to many, many to one, and many to many fashion. If the data are used for critical control algorithms, dependability measures have to be taken into account.

Concerning these preconditions, we evaluated contemporary methodologies on virtualisation already known from information and communication technology (ICT) to solve the above mentioned dilemma for smart grids. We chose two promising concepts on how to virtualise the communication layer of a smart grid, namely software-defined networking (SDN) and state-of-the-art cloud technology. For both approaches, we created proof-of-concept implementations, conducted several tests with these prototypes, and compared the results. In this paper, we present and compare the base technologies and our proof-of-concept implementations built on these technologies. Furthermore, we describe the conducted tests and evaluate their results.

In Section 2, we present the state of the art on how communication takes place in smart grids and outline trends and developments we identified to already take place or are expected in the near future. The following Section 3 describes example use cases for smart grid applications which were used to identify the requirements needed for a future proof communication framework. Section 4 elaborates possible approaches how to implement such a framework, followed by a description of the realised prototype given in Section 5. The results of our work are discussed in Section 6, and Section 7 concludes the paper.

## 2. State of the Art

### 2.1. Smart Grid Communication Networks

According to [1], communication networks for the smart grid should provide the following essential properties:

- the required quality of service (QoS) in order to prioritise and assure the delivery of critical traffic,
- reliable communication even if parts of the network fail, data security and privacy as well as resiliency against attacks, and
- scalability and availability even in remote locations.

As pointed out by [2], smart grids provide four general functions, specifically, advanced monitoring and control (ACM) to monitor and control the whole electrical system, demand-side management (DSM) to switch loads on or off to reduce the costs for consumers as well as for grid operation, generation and storage management to lower power generation costs and decide where

and how much excess energy should be stored, and finally system protection (SP) to provide resilience against faults and enable the smart grid self-healing.

From a communication infrastructure point of view, smart grids can be divided into three different network segments. home area networks (HANs) are small networks which span a customer site and provide communication among appliances located in that site. neighbourhood area networks (NANs) are part of the advanced metering infrastructure (AMI) and connect multiple HANs. wide area networks (WANs) connect NANs and all other parts of the smart grid, like substations, monitoring and control systems such as SCADAs, the utilities enterprise network and provide a connection to the Internet. A large number of devices is connected to the HAN and NAN part of the smart grid. Therefore, cost is an important factor in these areas of the network, and wireless and power line communication (PLC) are appropriate technologies to keep the costs at an acceptable level. Considering the bandwidth requirements and the great distances that are spanned by WAN networks, fibre-optic cables are the best suited medium for WAN communication, yet the high costs of this technology are a concern [2].

Different types of wireless networks are proposed for use in smart grids. Which technology is chosen depends on the bandwidth demand and the area that has to be covered by the network. For HANs and NANs, Zigbee and Wireless LAN are appropriate choices, while cellular communication networks can also span larger distances and satellite communication can be used where no other options are available [1]. PLC is a technology that uses the existing power lines, so that it can provide great coverage [3] and be deployed with costs comparable to wireless networks [4]. Depending on bandwidth and used frequency, PLC can be divided into broadband PLC and narrow band PLC [5]. Narrow band PLC is a solid choice for smart metering because metering does not require high data rates [6]. Because of the nature of power lines, they provide a noisy channel which results in a high bit error rate (BER). In addition, security concerns were expressed concerning PLC because it causes electromagnetic interference which can be received by radio receivers [1].

Smart grids produce various types of traffic, which require different levels of service from the network infrastructure, ranging from media access control (MAC) to end-to-end application layer protocols. Furthermore, many of the protocols used in smart grids, especially on the substation level, rely heavily on Layer 2 multicast. This further increases the complexity of the network configuration for smart grids. Technologies like virtual local area network (VLAN), multicast filtering, generic attribute registration protocol multi registration protocol (GRMRP), multiple MAC registration protocol (MMRP) and multiple VLAN registration protocol (MVRP) are used to satisfy the requirements the smart grid imposes on its communication infrastructure [7]. In addition to the aforementioned technologies, multiprotocol label switching (MPLS) is used to provide a network that satisfies the various different needs of smart grid communication. While MPLS provides means to enhance security and ways to implement efficient overlay networks, it falls short when it comes to innovation and the testing of new ideas because companies are limited to the feature set of their network hardware. In [8], it is shown that relatively cheap SDN switches are capable of providing the same level of performance as MPLS switches. Open-Flow, which was used as communication protocol between an SDN controller and SDN switches, supports all features provided by MPLS and can thus coexist with MPLS or even replace it.

SDN uses a central controller, or a cluster of controllers, as a centralised control plane for the whole network. The controller uses its southbound interface to push forwarding rules to its SDN switches. These switches only implement the data plane, which is responsible for packet forwarding. In most cases, OpenFlow is the protocol used for southbound communication. Traditional networks rely on network devices which implement the control plane and the data plane on every single network device. A big disadvantage of the traditional approach is that the control planes on the network devices need special protocols to interact with the control plane of other network devices to share different kinds of information, like routing information, while still having only a very limited view of the whole network. The SDN controller on the other hand has a complete view of the network and can

therefore create rules for traffic forwarding which can take into account the state of the whole network. In addition to this, the centralisation of the control plane increases flexibility and programmability of the network. Programs developed by third parties or by the network owner can communicate with the controller through its northbound application programming interfaces (APIs). By using these APIs, the behaviour of the network can be adapted to the needs of the network owner. Northbound APIs can also be used for network automation tasks [9].

A lot of research has been conducted on the use of SDN for smart grid communication. A comprehensive survey about this research is given by Rehmani et al. [10]. According to them, some of the main motivations for the use of SDN in smart grids are for example separation of traffic of different traffic types, Quality of Service, virtual network slicing, enhancing the resilience of smart grid communication, fast failure detection and recovery, and timely load shifting to prevent voltage drops.

SDN enables faster development of new routing algorithms and customer tailored traffic forwarding behaviour. In [11], a double constrained shortest path first algorithm is introduced that takes into account the current state of the whole network before making its routing decision. The information needed for this kind of algorithm is delivered by the controller which has a complete view of the network. In addition, Montazerolghaem et al. [12] showed that the complete view of the network can be used to forward traffic on optimal paths. They proposed a special routing scheme (OpenAMI) to find optimal routes and to provide load balancing through an AMI network. Through the use of OpenAMI, low end-to-end delay and higher throughput could be achieved. Concerning multicast, Pfeifferberger et al. [13] used OpenFlow's fast failover groups to increase reliability in substation multicast communication by reducing the number of lost packets between link failure and link failure recovery. Cahn et al. [7] proposed a software defined energy communication network (SDECN). This auto-configuring substation network architecture delivers the same functionality as traditional networks, but without the need for complicated and tedious network configuration as is the case with e.g., multicast filtering. This approach should be adaptable to other areas of the smart grid and reduce configuration time, costs, and errors.

Another interesting protocol in the smart grid area is message queuing telemetry transport (MQTT). It is a highly scalable publish/subscribe system that can deliver a published message to thousands of subscribers. MQTT has become an ISO Standard in 2016 and is proposed as WAN transport mechanism for distributed energy resource (DER) applications like automated demand response (ADR) systems in [14]. MQTT uses connection oriented communication over transport control protocol (TCP), which enables the system to detect lost publishers and inform subscribers. Because MQTT uses TCP, it can be easily secured using transport-layer security (TLS) [14]. Furthermore, MQTT offers three different quality of service levels and an authorisation system with topic level granularity. MQTT has been adopted for the Internet of things (IoT) because of its lightweight nature and simplicity. [15] used MQTT over general packet radio service (GPRS) to transport measurements from smart meters to a database. [16] proposes MQTT and MQTT for Sensor Networks (MQTT-SN) as communication protocols for user energy management systems (UEMS). The QoS features of MQTT were used to enhance communication reliability by decreasing the number of lost messages. The downside of this approach is that the amount of network traffic rises significantly because of the large number of retransmissions in unreliable, lossy networks. In [17], an IoT based architecture was proposed to enable real-time monitoring and management of large scale photovoltaic systems. MQTT has been found to be an excellent candidate for communication in such an architecture because of its efficient communication and low resource usage. In addition, distributed middleware frameworks like Coaty like to use MQTT to enable loosely coupled bidirectional communication.

### 2.2. Developments and Trends in Smart Grids

Today, SCADA systems for the power grid comprise data processing systems, human-machine-interfaces (HMIs), master terminal units (MTUs) and remote terminal units (RTUs). The data processors are in charge of evaluating operational data of the power grid and presenting this

data to the operators via the **HMIs**. The data required for that purpose comes from field-deployed sensors and actuators. These are connected to the **RTUs**, which, in turn, are connected to **MTUs**. **RTUs** and **MTUs** are responsible for pre-processing and aggregating the collected data and reporting it to the processing systems. The backbone for the operation is a communication network based on standardised networking technologies. Except for data preprocessing and aggregation, the main portion of data processing, system monitoring and control happens at the control centres of a power system operator. Thus, **SCADA** systems nowadays combine central monitoring and control functions with data acquisition in the field.

However, for centralised **SCADA** system architectures, both scalability (steadily increasing numbers of connected devices) and adaptability (to changing environmental conditions) have been recognised as weaknesses [18]. Both are experiencing growing importance: Scalability issues arise due to the continuous evolution of power grid control systems and the current developments regarding the Internet protocol (**IP**)-based connectivity of small devices (Internet of Things, **IoT**). System operators are confronted with the massive deployment of several thousands of off-the-shelf **IoT** devices, which are attached to the power grid infrastructure. For the future, it is predicted that the number of data points in power grids will rise massively. Consequently, the number and diversity of control and monitoring services (e.g., integrating mobile devices of maintenance work forces and providing them with real-time grid information [19]) will increase as well. Here, adaptability also comes into play, as the new services are paving the way for new operating paradigms such as micro-grids or islanding [20].

Several architectural approaches have been proposed in the past to overcome the shortcomings of centralised **SCADA** control structures. In particular, these are hierarchical approaches [21], multi-agent systems [22], as well as cloud and fog/edge computing approaches particularly in context of **IoT** [23,24]. Additionally, the mentioned operating paradigms for power grids (such as micro-grids or islanding) are enabled by decentralised control and monitoring structures. All these developments have in common that the relations of the subsystems and actors within the power systems will not be as stable as before. This also changes how information and data are distributed between the actors. The paths on which the data are exchanged will not be static, but will depend on the particular requirements of the services (e.g., real-time delivery of data), external conditions like the grid state (e.g., islanded operation), faults (in case of degraded operation), or environmental factors (e.g., in preparation of or as a reaction to adverse weather conditions).

A flexible data communication system can support these re-arrangements and minimise the need for reconfiguration at the component level. For instance, it is not necessary to inform a particular sensor device to send its information to another recipient, as this can be handled directly by the communication system ensuring the required communication quality while maintaining adaptability and scalability of the **SCADA** system. The previously described developments in the smart grid let us identify the following particular trends for smart grids:

- **Edge Computing:** Real-time data or data with highly sensitive information is processed and distributed only locally (e.g., within a closed/islanded group of communication nodes, determined by the instantaneous grid state) while other data are transmitted to the cloud for further processing and operation, not demanding the criteria stated above.
- **Data follows device:** Telemetry data are displayed on a mobile device, handheld or augmented reality device. For example: According to the geographic location of the worker, the device displays useful information dependent on the task (e.g., maintenance, commissioning, analysis) and based on the field devices nearby.
- To circumvent the shortcomings of **SCADA** systems stated above, computations and decisions will be made decentrally. Thus, data has to be distributed to components that are allowed to receive the data to which it has relevance.
- **Mass rollout of IoT devices:** With the proliferation of **IoT** and industrial internet of things (**IIoT**) devices for metering, the number of these devices will challenge network operators as well as conventional **SCADA** systems. On one hand, these devices have to communicate with other

devices in a secure way without revealing information to the outside world, on the other hand, the configuration effort has to be minimised. Thus, frameworks or systems which can manage a high number of communication endpoints are highly demanded.

- Self-adaptivity of future power systems (such as automated changes of the power grid topology regarding which loads are connected to which generators) might require dynamic interactions between the components of the electrical infrastructure and the ICT infrastructure. This leads to the requirement of programmable networking.
- Agent-based information distribution: Agent-based approaches are commonly described as self-adaptive, flexible, and scalable, which includes the ability to dynamically adapt the interactions (data exchanges) between individual agents. A flexible communication system could support this by directly controlling data distribution corresponding to current needs. For instance, in a multi-agent-system (MAS), an observer agent might be in the position to control the distribution of information according to the gathered system state (which might be influenced by system-internal or environmental conditions).

### 3. Example Use Cases

In this section, we present two sample use cases for grid-based routing, both being based on the same topological testbed shown in Figure 1. The letters *A*, *B*, *C*, and *D* denote field stations; hereby, *A* and *D* are primary substations, whereas *B* and *C* are switching stations. Dependent on the station type, different components can be combined within a station: on-load tap changers (OLTCs)  $T_i$ , voltage meter  $U_j$  and circuit breakers  $S_k$ . All stations use bus bars to connect their electrical devices. Neighbouring stations are connected via respective net segments controlled by these stations; hereby, the stations form a linear topology.

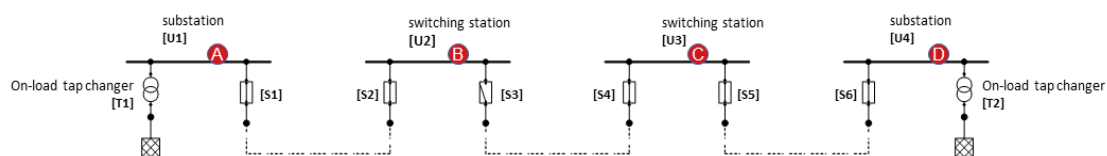


Figure 1. Testbed architecture.

In our case, the primary substations are compositions of an on-load tap changer, a voltage meter on the bus bar and a circuit breaker for each output. The switching stations are also housing a voltage meter  $U_j$  and multiple circuit breakers  $S_k$ . Additionally, all components are connected to a common ICT infrastructure, i.e., an IP based backbone, such that packets from one station can technically be forwarded to all others. In the base setup, all the network segments between  $T_1$  and  $S_3$  are driven by  $T_1$ . All other segments are driven by  $T_2$ . When closing  $S_3$ , both on-load tap changers  $T_1$  and  $T_2$  are driving the whole net. Conversely, when using the base setup and additionally opening  $S_4$ , the net segment between  $S_3$  and  $S_4$  is not powered.

#### 3.1. On-Load Tap Changing Based on Grid State

In a traditional power grid, an OLTC has a set of pre-configured data sources that provide the voltage measurements for its controller. These data sources are statically linked with the controller via a SCADA system. The algorithm might yet not only be interested in the measurements nearby, but also wants to be informed on rises and drops of voltages in certain other net segments. As an example, assume a high load at station *B* which leads to a voltage drop at  $U_2$ . Depending on the net segment between *A* and *B*, it is not possible to reliably measure this drop at  $U_1$ . Thus, the measurements from station *B* ( $U_2$ ) have to be reported to the OLTC  $T_1$  at substation *A* to build a more accurate foundation for decisions taken within the controller's algorithm.

In case a new meter is installed, the SCADA has to be reconfigured to accept the new measurements. Usually, all data sources have a fixed configuration where to send their data and the SCADA has knowledge about the meaning of the reported measurements. As soon as the grid's topology changes, these predefined links have to be adapted and the whole information flow between data sources (measurements) and the controllers (OLTC) needs to be redirected. In a smart grid, such changes are assumed to occur quite frequently (e.g., new smart measurement stations are rolled out, e-car-charging terminals are installed, etc.) and therefore, the number of measurement units in the topology is increasing considerably. Whether a measurement unit  $U_j$  is relevant for the algorithm of tap changer  $T_i$  is dependent on the state of the power grid and the current position of the switches  $S_k$ . Thus, the reconfiguration of components and the redirection of information flows have to be done automatically to meet the demands of future smart grids.

### 3.2. Delimiting Outages Spatially

Assume a broken cable between  $S_3$  and  $S_4$  in the topology given in Figure 1. Using current solutions, the network operator receives an alarm that a short circuit was detected and the voltage drops in all network segments between  $S_3$  and  $T_2$ . In the medium voltage grid, only the phases are distributed by wire. The reference potential is found by an earth connection. If a line breaks and falls to the ground, the potential (voltage) on this line drops immediately to ground and the current increases to infinity, thus a short-circuit occurs. The operator has then to investigate manually where the problem originates based on information like nearby construction sites, routes of overhead lines through forests and the like. To do this, operators have to physically visit one station after another to check the direction of the fault and identify the faulty region. Afterwards, the outage can be fixed by bridging affected net segments with other parts of the grid. Since this approach is very time-consuming, an ICT based solution could improve the situation and provide a big gain in efficiency. In a first step, data could be transmitted from the substation protection devices to a monitoring application on the operator's handheld. Thus, the operator would be able to remotely check the direction of the fault. In a second step, the whole process could be automated.

Since automated delimiting of faults requires a dense mesh of monitoring devices, potential scalability issues exist with traditional, centralised SCADA systems and large numbers of IoT devices. Referring to Section 2, modern distributed SCADA architectures might provide a solution. Agents are used here as logical monitoring and controller units, i.e., to monitor and control the behaviour of the grid. These agents do not necessarily need to be executed within the grid's ICT infrastructure (on premises) or within the stations. If their ICT connectivity allows for a timely delivery of necessary control data and security considerations do not stand against it, they could also be executed in a cloud environment. Furthermore, agents could also be implemented as decentralised controller applications which are automatically (re-)deployed in the system depending on the grid segmentation—thus, they can be part of a distributed SCADA system. In our example, two cases have to be distinguished:

- Detection of the outage and receiving information about the fault direction from the protection devices: The information about the fault direction from  $A, B, C, D$  is automatically forwarded to the agents in charge for the affected net segment (i.e., the agent controlling the respective switch  $S_4$ ). According to the instantaneous grid state (e.g., positions of the switches  $S_{1,\dots,6}$ ) received from other agents in the same grid segment, the agent in charge can delimit and isolate the origin of the fault. In our example, since the direction of the fault at  $C$  points to the same direction than at  $D$ , it is already clear that the problem is not between  $C$  and  $D$ ; hence, the origin of the problem has to be between  $C$  and the open switch  $S_3$  at  $B$ . Isolation can be performed by opening  $S_4$ .
- Detection of the outage without information from the protection devices (but with remote controllable circuit breakers): Since no information about the faulty network segment is available in this case, the agents can delimit the fault region by sequentially disabling net segments and rechecking if there is still a short circuit to be detected at the secondary substation. Since  $S_3$  is open, the fault must have occurred somewhere between  $T_2$  and  $S_3$ . Applying a binary search



scheme, we start splitting the network segments at  $S_5$ . Since the short circuit detection does not report a problem any more, we can delimit the error to be located between  $S_3$  and  $S_5$ . Moreover, by opening circuit breaker  $S_4$  and again closing  $S_5$ , no failure is detected at  $U_4$ , so we can deduce that the problem must be located somewhere between  $S_3$  and  $S_4$ .

This use case is widely identical to the M/490 high level generic use case fault location, isolation and restoration (FLIR) [25], identified by the Sustainable Processes document originated from the European Union mandate M/490. However, this use case collection makes no definitions on how to implement automatic fault management in distribution grids; thus, the approach at hand provides a possible solution.

### 3.3. Requirements

As can be seen from the described use cases, the current developments in the area of power systems include a range of challenges. On closer inspection, it becomes apparent that solutions for more flexible data distribution can be important milestones on the way to more automation and improved performance of future power systems. Summarised from the two use cases, the main goal of future power systems can be identified as situation-aware automated grid operations. This means that the operation of the power grid tends to become more automated and closer related to the current grid state. The rise of distributed automation is yet enabled by a rapid increase of ICT equipment installed within the grid. Altogether, this leads to a variety of challenges with respect to information distribution in the future grid:

- First, the required information needs to be available in the right place and at the right time, which becomes a particularly tough challenge when we assume that communicating elements such as intelligent electronic devices (IEDs), end devices used by service technicians like smart phones, backend servers, or even services built in software only, may continually change their location and need to be supplied with the required information while in transit.
- Second, not only the locations change but also the group of communicating elements change constantly through autonomously interacting agents forming coalitions to solve grid problems with or without the involvement of human operators.
- Third, the ongoing augmentation of the grid with ICT and the continuously developing operational modes require that new applications including software and hardware elements can be easily integrated into the information distribution system without interfering with already established applications.

Based on those challenges, a variety of requirements on communication infrastructure can be derived:

- R1 Forwarding of data required for monitoring and control needs to be adapted according to the instantaneous grid state.
- R2 Flexible and scalable m:n communication is required in order to take into account the potentially large number of communicating elements and changing interconnections.
- R3 Application-aware traffic separation has to be provided in order to guarantee required communication quality with respect to application requirements and it needs to be agnostic to the currently used communication links and insensitive to brief interruptions.
- R4 The communication solution has to support a differentiation between critical and non-critical traffic and to avoid interference between different traffic and application types.
- R5 The communication system should provide information forwarding with particular support for mobile agents and autonomously acting agent coalitions.
- R6 Information forwarding needs to be highly reliable and must not be affected by changes of the grid state and/or grid topology configuration.
- R7 A flexible communication system is required to contribute to fast innovation cycles and to make the integration of new devices safe and simple.

R8 The communication solution has to support degraded services in case of overload or failure situations.

In general, the concept of programmable networks may provide a basis on which future power grid communication systems can be built on in order to fulfil the above mentioned challenges and requirements. In this article, we elaborate on different types of programmable networks based on virtualisation-based communication in order to compare the technological capabilities in context of the stated use cases and their requirements.

#### 4. Virtualisation of the Communication Subsystem

Virtualisation is the provision of infrastructure functionalities by a software abstraction instead of direct access to dedicated hardware. The deployment of logical machines to real hardware can be handled by an appropriate management system, allowing the deployment of logical resources on physical devices on demand. Thus, the functionalities of several useful services and their impact to grid applications can be provided by an abstraction instance, which is independent of the underlying real (software or hardware) components. In addition, applications and users of infrastructural resources can share these components without interfering with each other (sandboxing).

##### 4.1. Virtualisation Benefits in the Smart Grid

Changing users and applications (and consequently their requirements) can be met by managing the deployment of the (changing) logical machines to the same physical infrastructure, as long as the capacities thereof are sufficient. Conversely, changes of the physical infrastructure do not affect users and applications, as long as their abstracted services can be provided sufficiently. With respect to the energy domain, virtualisation is used mainly for the following reasons [26]:

1. Effort for configuration or re-configuration of system components shall be minimised.
2. Control tasks shall be redistributed in case of a system downtime (either due to maintenance or outage).
3. Situational awareness in case of failures, overload, or deliberate attacks shall be improved.

As pointed out in Section 3, the use cases mentioned induce demanding requirements especially on the communication subsystem. Furthermore, these requirements are quite volatile. Virtualisation of the communication subsystem has the potential to support the control applications in the grid by detecting malfunctions or overload (no. 3), activating functional components (no. 1) and redistributing control data to the newly activated components (no. 2). However, communication virtualisation is a very vague term, covering a wide range of virtualisation technologies: **VLAN**, virtual extensible LAN (**VxLAN**), **MPLS**, **SDN**, software-defined WAN (**SD-WAN**), programming protocol-independent packet processors (**P4**), Cloud Computing, and Edge Computing.

##### 4.2. Traditional Virtualisation Technologies

Traditionally, virtualisation is seen as an abstraction of physical infrastructure. Thus, a logical infrastructure can be defined, which is deployed upon the underlying physical infrastructure in an appropriate way. In local-area network surroundings, the easiest way to achieve this abstraction is the using of **VLAN**. With **VxLAN**, **VLAN** frames using IEEE 802.1Q tagging can be tunnelled through the Internet as “underlay” network; thus, the **VLAN** concept can be extended to wide-area surroundings. Both **VLAN** and **VxLAN** allow for a separation of different data flows and their association to different logical networks (the “overlay” networks). This is especially important for critical infrastructure such as the power grid, as it allows for separate critical control data from less critical billing data and again from any other kind of traffic present in the underlay. Nevertheless, this kind of separation is considered too lightweight in real-world surroundings, as it does not take into consideration any performance means and is not adaptable to changing application requirements.

Here, either a dedicated communication infrastructure or a higher level separation technology is used: **MPLS**. With **MPLS**, traffic is organised into so-called flows along basic attributes such as sending and receiving addresses and ports, as well as **QoS** attributes. Thus, data packets belonging to the same flow (i.e., sharing common attributes) can be treated the same way. By doing so, an effective separation of data can be achieved, while at the same time common forwarding procedures (including same **QoS** treatment) for associated data packets can be provided (**MPLS-Traffic Engineering**). Hereby, logical overlay connections (**MPLS** paths) over public underlay can be provided, which also takes care of performance means. Still, one of the main intentions of network virtualisation (and also one of the basic requirements of the use cases at hand) is the flexibility to react to current situations, e.g., overload or malfunctions of the grid and/or of the communication infrastructure. **MPLS**, however, is not capable of providing this flexibility, as **MPLS** paths have to be defined and booked from the provider in advance.

#### 4.3. Software-Defined Virtualisation Technologies

As solution for this issue, **SDN** is widely accepted [27]. As mentioned in Section 2, **SDN** allows for configuring the forwarding scheme dynamically via software, where forwarding decisions are made in **SDN** capable switches based on a comprehensive and adaptable rule set provided by central **SDN** controllers. Furthermore, it also enables the integration of packet metadata into the rule set, including sending addresses and ports. As **SDN** provides the desired flexibility in its forwarding behaviour, while keeping the properties of traffic separation and **QoS** consideration known from **MPLS**, it seems to be an appropriate choice for use in smart grid environments, and especially for the use cases at hand. However, there are still two issues to be mentioned.

First, the **SDN** controller is very flexible in its forwarding logic and can include application layer information via its northbound interface, yet the rules communicated to the switches via the southbound interface must be broken down to simpler criteria including open systems interconnection (**OSI**) [28] layer 2 and 3 information, but not higher. This way, real application layer routing is not possible, as **SDN** switches are able to route for example based on **MAC**, **IP**, **TCP** or user datagram protocol (**UDP**) header fields or some link information, but not based on states of end systems. One possible solution to overcome this issue is to use **P4** [29] instead of **SDN**, which provides full flexibility on the control plane, as well as the possibility of adding data plane operations (e.g., for data aggregation). However, this approach currently has an important drawback: the lack of devices in the smart grid domain supporting this standard. As it is a rather new approach, it will take several more years for widespread adoption in the power industry.

Second, for massively distributed systems, as are common in the smart grid domain, a purely switched system is not viable. **SDN** may have proven to scale well, but in this case a complete distributed **SDN** infrastructure would have to be provided by the network operator. As soon as traffic has to be routed over a public underlay infrastructure, other approaches are necessary. This is provided by **SD-WAN** [30], which allows for the definition of a private overlay over a public underlay infrastructure. **SD-WAN** solutions are already quite common in the smart grid domain [10]. However, for the given use cases, which are located in the distribution domain and thus usually operated by a single network operator, an **SD-WAN** based solution would cause unnecessary dependencies on out-of-premise infrastructure parts, thus generating availability and integrity issues (see Section 4.4), such that an on-premise **SDN** solution appears to be the better solution here.

The controller used for the **SDN** version of the prototype at hand (as described in Section 5.7) is named open network operating system (**ONOS**). **ONOS** is developed by the open networking foundation (**ONF**) with resilience, performance and scalability in mind to deliver carrier grade **SDN** solutions. **ONOS** offers support for different southbound protocols like OpenFlow, NETCONF and T1, just to name a few, and it keeps expanding its list of southbound protocols as can be seen by the recently added support for **P4** runtime. The northbound **APIs** offer the option to run applications on the controller-hardware through native interfaces as well as off-box using representational state

transfer (REST) and grpc remote procedure calls (gRPC). As described in Section 5.7, the REST API and OpenFlow were used to implement the SDN version of grid based routing.

#### 4.4. Cloud and Edge Computing

Alternatively to infrastructure virtualisation, network functions can also be virtualised. These network functions (e.g., routing, anomaly detection) are then consumed by distributed applications (e.g., grid control) as services (“network as a service”). In case the location of the services is considered irrelevant, this is called cloud computing. However, this requires trust in the capability of the cloud provider to ensure privacy and security. Furthermore, the access to (public) cloud services is usually given via public infrastructure. This raises additional questions regarding the efficiency of the underlay infrastructure (which is important to guarantee service availability), as well as regarding the effectivity of the data flow separation (which is important to guarantee service integrity). As mentioned, one possible solution for those issues is provided by SD-WAN, a wide-area derivative of SDN. SD-WAN is an overlay technology which provides the same level of configurability as SDN, but additionally controls the flow separation over an insecure underlay, as well as the adherence to agreed upon service performance (QoS monitoring).

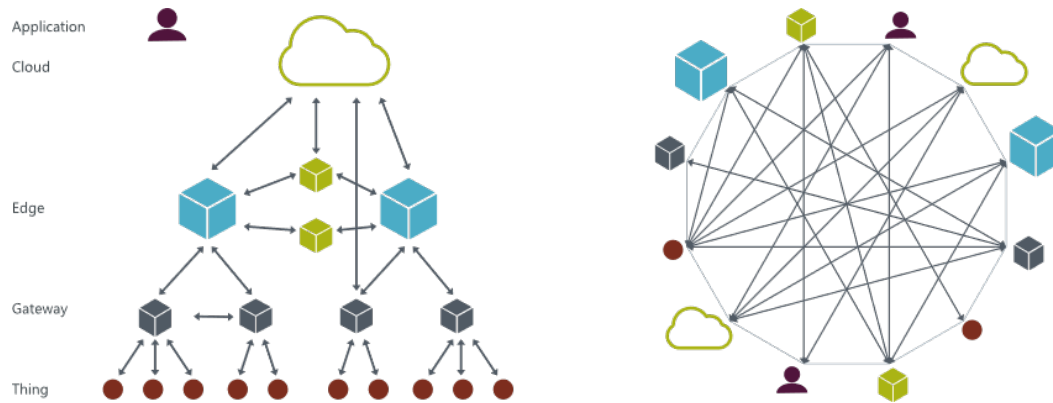
For distributed applications in critical infrastructure, it is for the stated reasons commonly preferred to keep the location of the service provision known—ideally, on premises of the grid operator, but at least in a secured space where others do not have access. This is called private cloud. However, when availability guarantees are desired, not only the service provision, but also the data exchange has to be performed under controllable circumstances. This means that it is not sufficient to run applications on premises, but it is also necessary to exchange data via tunnels when using public infrastructure, since it can usually not be guaranteed that the public infrastructure provides the desired performance (at least not without additional technologies such as SD-WAN). Integrity on the other side is usually provided via data flow separation; this can be realised using the overlay approach (secure tunnels over public infrastructure). One solution to the named performance issues is to apply distributed control logics, which for instance avoid real-time communication over public infrastructure and make decisions locally. This is often referred to as edge computing, as the real-time tasks are then executed locally on the network “edge”, i.e., using on-premise components.

#### 4.5. Message Queue Solutions and Distributed Middleware Frameworks

For the use cases at hand, a re-distribution of grid services in case of malfunctions or overload is desired. One solution for this is that the necessary control data can be delivered to all potential data sinks (which then perform the actual control tasks) simultaneously. This can for instance be achieved by using IP multicast, or by appropriate message queues (with or without message brokers). In this case, situational awareness of malfunctions and overloads is not required anymore for the network infrastructure; in addition, network re-configuration (e.g., re-routing) can be omitted. Every station has the same control data available, and the voting for the leading station, which decides for the control actions, is done at the end devices then. However, this solution pulls back the management effort of the voting process from the network infrastructure to the end devices, i.e., it is done again at application level rather than at infrastructure level.

A possible way out of this dilemma is the introduction of distributed middleware frameworks, which operate between the infrastructure and the application level. They may take over routing decisions on the basis of application related information, thus providing application layer routing within an overlay network, which is operated above an underlying infrastructure, but below the actual applications. As shown on the left side in Figure 2, typical edge and cloud applications are organised following a hierarchical top down approach using multiple gateways aggregating information and forwarding it to their northbound or southbound components, respectively. These gateways may also hide information from the applications if needed, and provide thus a functional abstraction, i.e., virtualisation. However, the hierarchical approach seemed not appropriate for our purposes,

as changes in the topology would lead to complete reorganisations of the established communication tree. Thus, another data distribution technology is sought for use in smart grid ICT infrastructure (regardless of it being cloud/edge based or dedicated).



**Figure 2.** Conventional cloud/edge architecture and Coaty architecture (Source: [31]).

Coaty, as shown on the right side of Figure 2, is an open source framework for collaborative IoT, which is used to build decentralised applications in an autonomous, distributed environment. This allows for a more flexible and adaptable way of communication and collaboration of the participating components. While many communication platforms only provide publish–subscribe, push–pull or request–response patterns in either one to many, one to one, or many to many fashion, Coaty allows a mixture of these. Technically, Coaty is designed to run completely decentralised; nevertheless, it uses an MQTT broker to exchange metadata about its peers. This metadata are used to associate the matching endpoints to each other, such that they can autonomously communicate with each other. In contrast to a plain MQTT broker, it also possible to modify this association by applying additional dynamically computed rules to decide which devices are allowed to communicate to each other.

#### 4.6. Overall Assessment

After all, these technologies can be assessed regarding their potential usefulness for the use cases depicted in Section 3, with a special focus on the fulfilment of the listed requirements, as shown in a +/o/- scheme in Table 1:

**Table 1.** Technology Assessment.

Technology	Flexibility (R1, R2, R5, R7)	Flow Separation (R3, R4)	QoS Provision (R5, R6, R8)
VLAN	-	+	o
VxLAN	-	+	o
MPLS	-	+	o
SDN	o	+	+
SD-WAN	o	+	+
P4	+	+	+
Coaty	+	o	o

In total, P4 is the most promising candidate from a pure technological point of view; however, the missing maturity leads to practical issues like a small number of supporting devices. This made the much more mature SDN a candidate of choice for the project at hand. The other candidate Coaty is somewhat orthogonal to the other mentioned technologies. Like all cloud and edge solutions, it has to rely on existing underlay technologies and is thus incorporating some dependencies especially concerning QoS. Flow separation can be performed by appropriate tunnelling technologies,

additionally encrypting the original data. However, this is dependent on the effectiveness of the used security approach.

## 5. Prototype

A common element to smart grids is the application of digital processing and communications to a power grid. The project VirtueGrid (<https://projekte.ffg.at/projekt/1822052>) analysed modern communication paradigms like SDN, P4 and broker-based communication approaches to decrease the complexity of the applications used on the devices in a smart grid. This was achieved by a clear separation of tasks concerning the communication infrastructure from the power grid itself and the integration of additional subsystems managing the dependencies between communication endpoints transparently.

### 5.1. Architecture

Figure 3 shows how these subsystems can be directly mapped to the corresponding layers in smart grid architecture model (SGAM) [32].

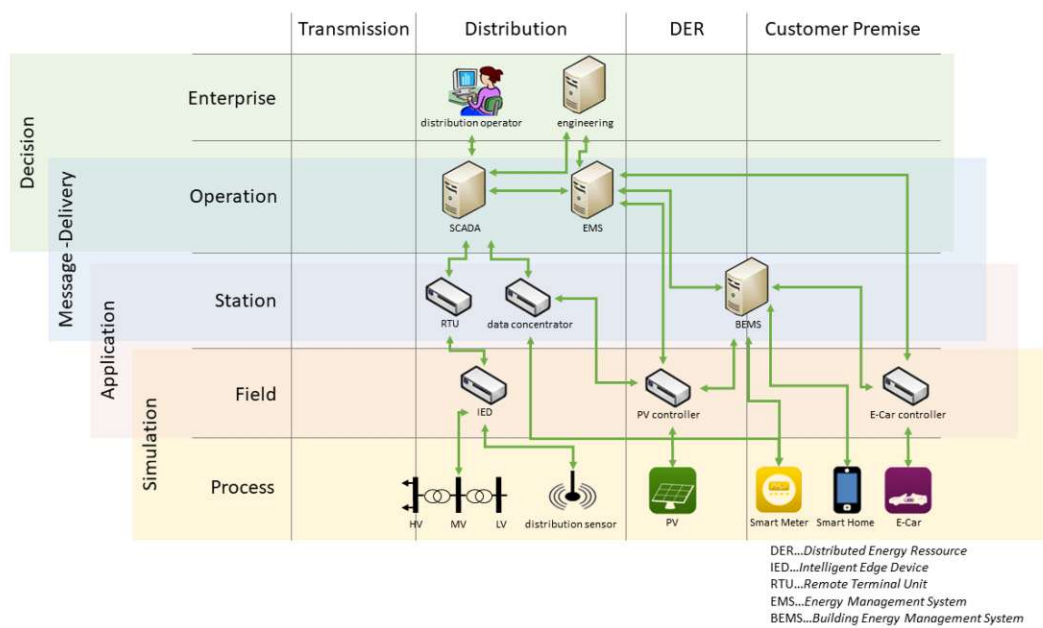


Figure 3. Mapping to SGAM.

- **Decision Subsystem:** The Decision Subsystem represents the entity deciding on routing decisions in the Message-Delivery Subsystem based on topological changes in the grid topology. Therefore, this subsystem has components facing two orthogonal responsibilities. First, the mapping of relevant paths in the grid from sources to sinks based on the grid's state and second, the mapping of these paths to the communication topology.
- **Message-Delivery Subsystem:** The Message-Delivery Subsystem is the part of the system responsible for the rerouting, duplication and deduplication of the messages sent by components to the desired receivers dependent on the mapping of the grid state onto the communication topology computed in the Decision Subsystem.
- **Application Subsystem:** The Application Subsystem represents the software applications running on the physical endpoints of the smart grid. This can either be the logic for switches, meters (power and voltage sensors) as well as transformers (including an OLT) or power and voltage sensors. Since these sensors are physically connected within the transformer station, the power and voltage sensors are also part of the transformer application and can be considered a single

unit. We further assume that the applications don't need any knowledge about their position in the grid and the grid's topology by themselves. In terms of communication, they only know about their dedicated endpoint in the communication topology. The data they receive from or provide to the environment (either simulation or the smart grid itself) is specified via an a priori configuration applied during setup and deployment.

- Simulation Subsystem: The Simulation Subsystem is the link between the applications running on the endpoints and the real or a simulated environment.

In general, the task of these additional subsystems is to automatically reroute data packets to the required endpoints in the communication topology when changes in the grid topology are made. Since the communication topology does not know anything about the grid topology and vice versa, intelligent components have to be introduced controlling the behaviour of the message delivery dependent on the grid state. Therefore, we represented the current state of the power grid as a directional graph  $G = (V, E)$ , where  $V$  represents the devices (e.g.,: meter, power switch, transformer) and  $E$  represents the connections between them. Dependent on the states of the power switches, the graph is partitioned according to the power supply situation in the power grid. All metering data measured in one partition is forwarded to all transformer applications in that partition.

### 5.2. Lab and Field Setup

As shown in Figure 4, our prototype has been implemented and proved in the scope of the project using several lab setups using different communication protocols (e.g.,: MQTT, IEC 60870-5-104), broker based (Coaty) and SDN based (OpenFlow/ONOS) communications and various simulations of low and medium voltage grids culminating in a field trial:

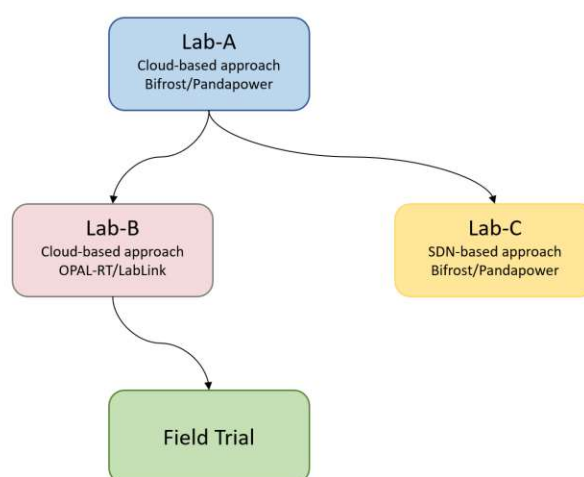


Figure 4. Lab setups.

- Lab-A: In the Lab-A setup, we used power system simulation based on Bifrost and Pandapower to simulate the dynamics and the state of the smart grid. As communication framework, we used a broker based version of Coaty with custom defined messages. For application deployment, docker-compose was used on a local machine.
- Lab-B: In this setup, we used the OPAL-RT/Lab-Link realtime power simulator representing a medium voltage grid. The communication and deployment strategy remained identical to Lab-A.
- Lab-C: In the Lab-C setup, we again used Bifrost/Pandapower to simulate the dynamics and the state of the smart grid. Instead of the cloud or broker based approach, we used an SDN based approach using IEC 60870-5-104 for the communication infrastructure. Furthermore, we used

Mininet to simulate the network characteristics based on the network we utilised during the field trials and ONOS as the SDN controller (see Figure 5). The deployment of the applications still took place on a single host using docker-compose.

- Field Trial: Lastly, our system has been tested in a field trial. Therefore, the applications for the meters and the transformers have been installed on real hardware. Since medium voltage grids are considered critical infrastructure, the possibilities for R&D activities directly within the grid are limited. Thus, we also implemented a connector injecting data from the medium voltage grid simulator into the applications in the field instead of using real power grid data. The communication throughout the field trial took place via a broker based implementation based on Coaty, tunnelling IEC 60870-5-104 packets, located in the on-premises cloud of the grid provider.

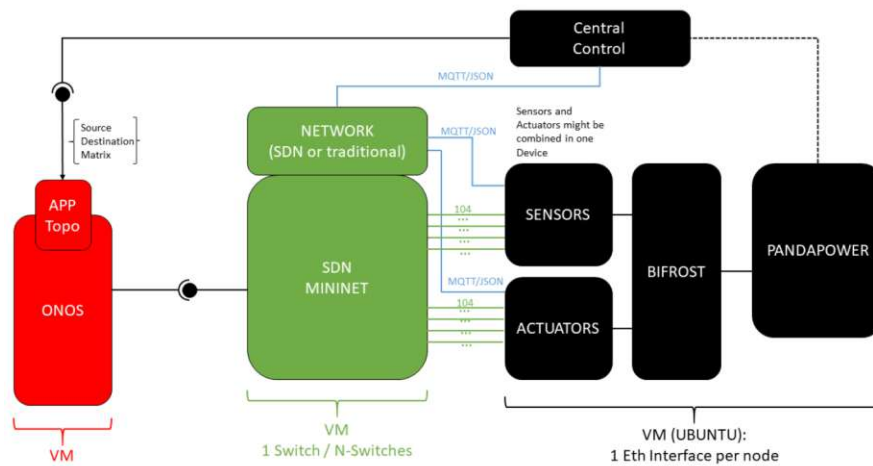


Figure 5. Lab C Virtual Machine Setup.

Given the fact that our solution supports multiple different simulation frameworks, the integration into the real hardware tested, and different communication systems and protocols, the architecture of our solution has been split into four parts, representing the subsystems described above. While Figure 6 shows a top level overview of the interactions between the subsystems, Figure 7 outlines the interfaces between these described in the remainder of this section.

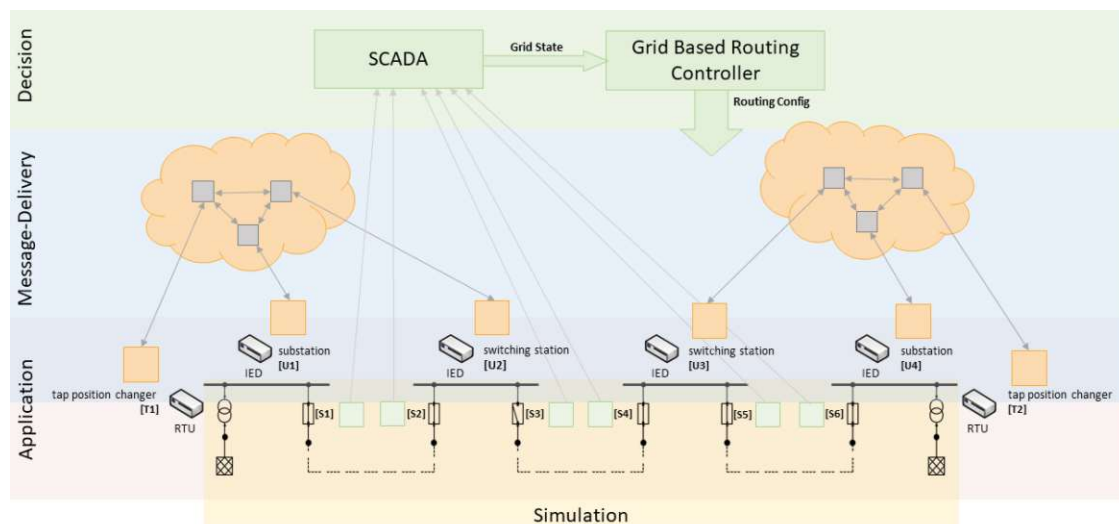


Figure 6. Architecture: Mapping of the top level components.



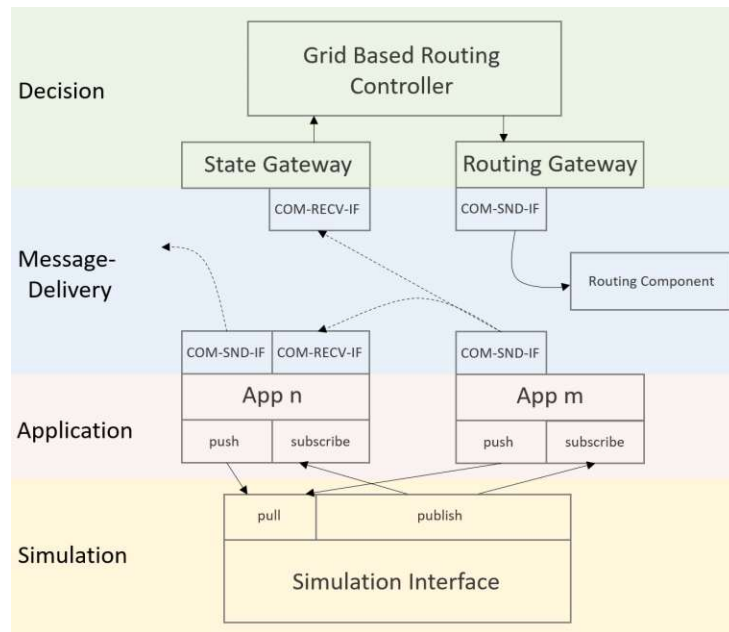


Figure 7. Architecture: Overview of the top level components.

### 5.3. Component Integration: Message-Delivery Subsystem

Dependent on the setup of our deployment, different communication variants could be used. Their integration was detached from the application logic using a REST based interface as shown in Figure 6. The interface of the *COM-SND-IF* is the bridge between the Application Subsystem and the Message-Delivery Subsystem. The *COM-RECV-IF* is its counterpart and thus serves as bridge between the Message-Delivery Subsystem and the Application Subsystem.

Both interfaces have been implemented as:

- interface for the cloud/broker based solution via Coaty,
- interface for a real communication network using custom **UDP** messages,
- interface for a real communication network using IEC 60870-5-104 and
- a bridge to translate between IEC 60870-5-104 and the broker based format.

For the application, the communication stack used is transparent. The application always interfaces an *COM-SND-IF*. Due to this design approach, applications written once can be combined and used with either of them. In contrast to the *COM-SND-IF*, the *COM-RECV-IF* is connected to the application (providing the interface for incoming data). Thus, the Application Subsystem has no knowledge of the Message-Delivery Subsystem itself.

### 5.4. Component Integration: Decision Subsystem

*State Gateway:* The State Gateway receives its data from the Message-Delivery Subsystem via the *COM-RECV-IF*. While the *COM-RECV-IF* provides the endpoints *tappos/*, *meter/* and *switch/*, the state gateway only uses data received via the *switch/* endpoint. Since the *Grid Based Routing Controller* has no information about the communication topology, a mapping of the communication endpoints to the grid-topology endpoints has to be done in the state gateway. The mapped data are forwarded to the *Grid Based Routing Controller* via a REST API, namely the *State Gateway Client*.

*Routing Gateway:* The Routing Gateway receives the updated connections in the grid topology from the *Grid Based Routing Controller*, transforms it to communication links in the communication system and then forwards the updated routing configuration to a dedicated routing component (namely the *Routing Component*) which either configures Coaty or instructs the **SDN** controller to implement the correct information flows in the network.

The *Routing Gateway* uses a [REST API](#) served by the *Routing Component* providing the endpoint `/api/v1/channel`. The GET request on `/api/v1/channel/` returns a full connectivity update for the whole grid topology and the POST request updates the full connectivity for the whole grid topology. An example payload representing the grid state shown in Figure 1 is shown in Listing 1:

**Listing 1.** Payload for the grid state shown in Figure 1.

```
[
  {
    "epa": "U1s_communication_id",
    "epb": "T1s_communication_id",
    "active": true
  },
  {
    "epa": "U2s_communication_id",
    "epb": "T1s_communication_id",
    "active": true
  },
  {
    "epa": "U3s_communication_id",
    "epb": "T2s_communication_id",
    "active": true
  },
  {
    "epa": "U4s_communication_id",
    "epb": "T2s_communication_id",
    "active": true
  }
]
```

*Grid Based Routing Controller:* The *Grid Based Routing Controller* is the only component with knowledge about the effects of switch states on the topology. It receives the switch updates from the *State Gateway* via a REST API. The controller stores the current state of the grid in the graph  $G = (V, E)$  and activates or deactivates the edges in  $E$  according to the state of the switches. Afterwards, it calculates all strongly connected components and forwards them to the *Routing Gateway*.

##### 5.5. Component Integration: Simulation Subsystem

On the southbound side, the applications are connected to a simulation gateway, abstracting the real simulation environment (e.g.,: Bifrost, OPAL-RT/Lablink). The Simulation Subsystem is connected via a custom ZeroMQ interface component. It provides an application and a simulation endpoint. The data transfer from the application endpoint to the simulation endpoint is performed via *push/pull*, while the transfer between simulation and application environment is performed via the *publish/subscribe* pattern. It is assumed that the simulation environment is omnipresent and thus fully available prior to startup of the applications.

##### 5.6. Coaty Based Message Delivery

In the broker based approach, the communication takes place via Coaty backed by a *Mosquitto MQTT* broker. Besides many very common communication patterns, like request–response (two-way-communication) and publish–subscribe (one-way-communication) as well as combinations of these (two-way-communication), the framework provides contextual routing capabilities.

Each source (*IoSourceController*, e.g., Meter, Switch) publishes the data it produces (as *IoSource*) to its own topic on the *MQTT* broker. An actor (*IoActorController*, e.g., Trafo) interested in this data or

subsets of it can be associated with the source by sending association events. These association events are used to match source topics to subscriptions based on meta information in association events. Consequently, this association can be terminated by a disassociation event.

After startup, all devices advertise their *IoSource* and *IoActor* capabilities to the system. The communication interface provides the following types of *IoSource* and *IoActor*:

- *SENSOR\_IO\_VALUE\_TYPE*: This type is used to transmit values for metering (e.g., Meter or Transformer applications) and contains measurements of voltage and power for each phase in [V] and [W], respectively.
- *TAPPOS\_IO\_VALUE\_TYPE*: This type is used to monitor decisions about the current tap changer made in the transformer application. It has no operational need in our implementation.
- *SWITCH\_IO\_VALUE\_TYPE*: This type is used to exchange the states of circuit breakers (e.g., Switch application) with the *State Gateway*.

As shown in Figure 8, in our implementation, the association and dissociation of the actors and sources is done in the *Routing Component (RuleBasedIoRouter)*. Therefore, the channel information received from the *Routing Gateway* is translated to a set of *IoAssociationRules* which are further published to all the device clients by the framework.

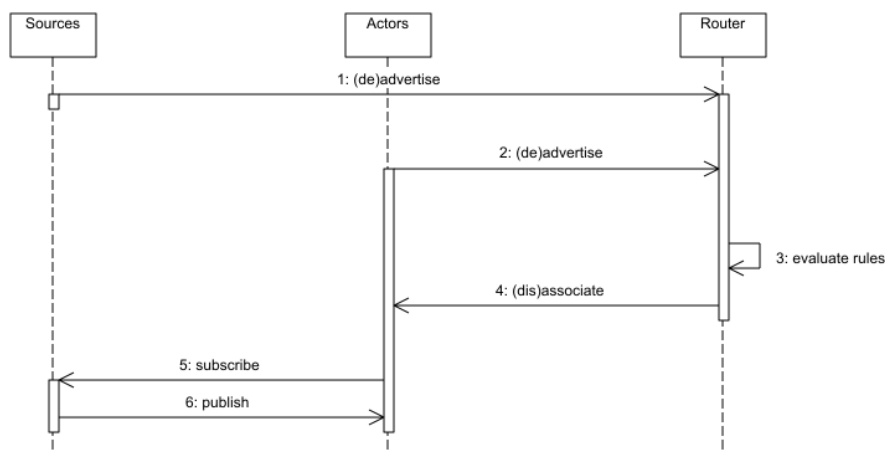


Figure 8. Coaty: Interactions of Sensors and Actors with the Router.

### 5.7. ONOS Based Message Delivery

The *SDN* prototype is implemented within a virtualisation environment with three virtual machines. One machine contains the simulation of the electrical grid, consisting of models for the power line topology, transformers and sensors. Three of these devices (one voltage sensor and two transformers) are picked for demonstration. Figure 9 depicts the simulation used for development and evaluation. The values of the voltage sensor U2 are required to be received either at transformer T1, T2 or both at the same time in accordance with the current switch state of the underlying power grid (which is also simulated by the power grid simulation environment, but not explicitly shown in Figure 9).

The *SDN* simulation represents a real-world inspired operation level *ICT* network. This is a pure *SDN* network, consisting of several *SDN*-capable switches interconnected via fibre-optical links. With the exception of switch SW-1, all switches have a fixed data plane configuration defining the forwarding behaviour of packets along forwarding path A, B, or both. The switches send the *UDP* datagrams, destined to T1 along path A, and datagrams destined to T2 along path B. The forwarding decision is taken using the addressing information in the *IP* header of the datagrams.

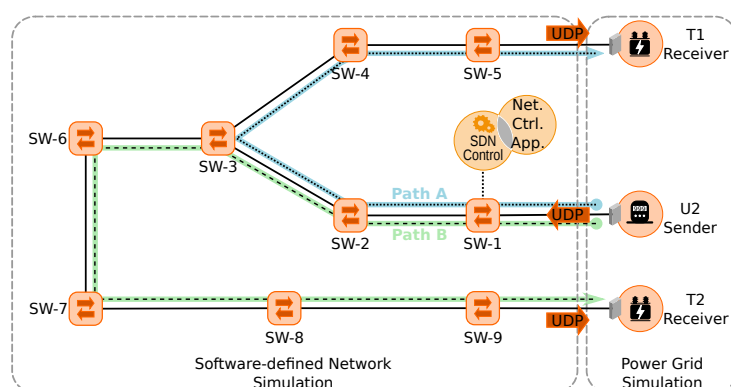


Figure 9. The SDN prototype as used within the project.

The network simulation is based on Mininet, and the switches are open Vswitch instances. In this demonstrator, UDP/IP is used for the transmission of the sensor values from the voltage sensor U2 to the receivers at T1 and T2, respectively. Although it is common to use TCP for 104 communication, we chose to use UDP for this proof-of-concept in order to omit the complexity of TCP connection handling when switching between receivers. We see this approach as valid as receiving up-to-date voltage sensor values is considered substantially more critical than avoiding packet loss. The reason is that out-of-date sensor values lose their relevance for the control algorithms and furthermore, as the 104 protocol maintains a session state, data loss becomes obvious to the control algorithms. An implementation based on TCP/IP including the necessary TCP session handling is possible future work.

To achieve the goal of relieving the sensor device of the need for awareness regarding its communication counterpart needed in the present grid configuration (T1, T2, or both), the required logic has been outsourced to the Routing Component. The Routing Component, in turn, controls the network control application running on the SDN controller. Thus, for decision-making on how to forward packets according to the current situation, the same logic as described in Sections 5.3 and 5.4 is applied. In contrast to the approach described in Section 5.6, the network control application installs SDN rules for the reconfiguration of the network's forwarding behaviour by controlling the way that switch SW-1 treats packets coming from U1. To enable U1 to send its packets regardless of the current grid situation, a placeholder destination address (IP and MAC) is used for the UDP packets that are sent out, which is then altered to the currently desired destination by SW-1.

The forwarding rules applied at switch SW-1 are as follows: in situations where the sensor values of U1 are needed by transformer T1, for any packet incoming from sensor U1, the destination MAC and IP addresses are changed to the IP and MAC addresses of transformer T1. The other switches within the network contain static forwarding rules, ensuring that packets with destination T1 are forwarded along path A. Respectively, if the sensor values of U1 are needed at transformer T2, the destination MAC and IP are changed to those of T2. Again, the static configuration of the other switches ensures that the packets are forwarded along path B. In case both paths are active simultaneously, switch SW-1 is configured with a rule which duplicates the incoming packets from U1 and modifies each of the packets according to the rules above, which ensures that one of the duplicate is forwarded along path A, while the other is forwarded along path B.

The SDN controller consists of an ONOS instance running software version 2.0.0. Regarding ONOS applications, only "Default Drivers", "OpenFlow Base Provider" and "Optical Network Model" are enabled. The network control application which performs the flow switching operations consists of a Java application providing a REST interface to the Routing Component. The information exchanged via this REST interface consists of two endpoints (designated "A" and "B" and identified via IP addresses) and a boolean (designated "active") which is used to indicate whether the connection between these two endpoints should be active or not. Upon receipt of information via this REST

interface, basic plausibility testing is performed and if the desired connections correspond to a known and valid network state, the application alters the SDN flows accordingly, in turn using the REST interface provided by ONOS.

## 6. Evaluation

In this section, we compare the state-of-the-art SCADA communication with the two data distribution approaches using SDN or Coaty. First, we give a general qualitative comparison of the three approaches followed by a quantitative estimation of the complexity to reconfigure the information exchange within the three approaches. The reconfiguration becomes necessary in order to fulfil the challenges and requirements derived from the example use cases in Section 3. For the quantitative comparison, we refer by way of example to the electricity grid model introduced together with the application cases.

### 6.1. Differences in the Information Exchange

The communication systems under consideration differ in the way that the information is forwarded from source to destination. In state-of-the-art SCADA systems, standard network technology as described in Section 2 along with some virtualisation techniques as described in Section 4 is employed. As a result, the information forwarding follows the conditions given by the commonly applied forwarding and routing protocols such as IP. This changes for the other two approaches: The middleware approach (Coaty) uses meta information of published messages such as topic titles, data formats, etc. in order to store messages within message queues (or topics) and message receivers pull these messages on demand. SDN forwards packets throughout the network based on information encoded into the packet headers such as addressing information and protocol types. The difference between the classical SCADA approach and SDN is that, with SDN, the forwarding decision for a particular packet at each node uses programmed packet processing rules instead of classical forwarding or routing protocols.

This general distinction implies differences in several characteristics between the given approaches which affect their suitability regarding challenges and requirements of the example use cases. In the following, the approaches are compared referring to the requirement groups (as depicted in Table 1) flexibility, flow separation, and QoS provision.

### 6.2. Flexibility

The flexibility of the communication system refers to the ability of a communication system to adapt connections between communication endpoints with respect to the current grid state, the addition and removal of endpoints, as well as to the integration of new device types and services. For the first two aspects, the adaption should happen particularly quick while the latter is considered a more long-term development.

For classical SCADA systems, the idea is to rely on a relatively static communication system with relatively rare adaptations. Thus, when it comes to the reconfiguration of the connections between communication endpoints, the endpoints have to be modified directly. For every change, each controller of an affected field device needs to be reconfigured. Furthermore, whenever new devices and services are integrated, they are required to follow the conditions given by the existing network.

SDN, in contrast, provides the possibility to reconfigure the forwarding behaviour of the network directly without any modifications at communication endpoints in order to ensure that traffic flows arrive at the correct endpoints. In this case, the forwarding behaviour of affected switches needs to be adapted to fit to a new grid situation and establish the required end-to-end paths. Even for the integration of new device types and applications, SDN is less limited regarding the conditions of the network. In case of a change, modifications are necessary at all affected network components.

In comparison to SDN, where end-to-end paths must be established, for Coaty, grid-state-aware message forwarding and m:n-communication can be reduced to inserting published messages into

the correct message queues by the broker. Furthermore, the diverse communication patterns such as synchronous/asynchronous, client–server, or publish–subscribe provided by middleware solutions give more options for different interaction patterns between communicating agents. Necessary tasks are limited to modifications of the message broker configuration.

### 6.3. Flow Separation

Flow separation refers to the ability of a communication system to distinguish between different data flows of particular applications or classes of applications and, if necessary, to avoid interference between the different flows.

In classical **SCADA** systems, flow separation is not directly possible as **SCADA** systems only have control over the field devices. Flow-separation, however, requires the re-configuration of the layer 2 and layer 3 protocols of the network and is typically an involved and risky task as configuration changes often interfere with other protocols. In critical infrastructure such as power grids, such changes require detailed planning and a careful implementation.

Regarding the requirements for traffic quality enforcement and traffic interference avoidance (R3 and R4), **SDN** provides some benefits here as the state of a network can be monitored globally and modifications are done based on that global state. Sophisticated tools help to establish a highly automated and risk-reduced implementation. Fast reconfiguration is possible. The operation of **SDN** on the network layers 2 and 3 allows the separation of traffic flows either by applying distinct traffic handling (similar to quality of service technologies such as DiffServ) or the deployment of particular flows on completely separated paths through the network in order to avoid mutual interference of traffic flows. The establishment of distinct paths across the network additionally helps to improve the reliability of the communication, for instance, by circumventing problematic areas of the network.

The Coaty-based approach, in contrast, does not provide capabilities for low level control of the forwarding behaviour of the network. Separation of traffic and interference avoidance cannot be directly implemented. Therefore, network reconfiguration similar to classical **SCADA** is necessary.

### 6.4. QoS Provision

QoS provision allows message delivery in compliance with the performance requirements (such as latency) of a given application. It can also be brought together with flow separation, referring to the capability of the communication system to comply with the particular requirements different applications have regarding the data they exchange. The reliability of the communication system is particularly important for requirements R6 and R8.

Classical **SCADA** systems have almost no means of adjusting QoS parameters and, as in case of flow separation, a direct reconfiguration of the network would be required.

**SDN** gives in-depth control of traffic distribution and forwarding within the network and particularly allows for fine-grained adaptations focusing on aspects of communication quality such as end-to-end latency of certain paths which may change according to the current grid configuration. Basically, all affected network components need to be reconfigured. As this can be done particularly fast, **SDN** is suitable to support degraded power grid operation. The capability of **SDN** to control information forwarding on lower layers of the network stack also allows for a quick reaction to failures within the network, or for load balancing by forwarding packets along disjunct paths.

The Coaty approach does not provide direct QoS control within the network; however, prioritisation can be implemented at the message broker by applying priorities for certain messages or message queues. Usually, high-layer meta-data and content information is used as a basis for application-specific handling of messages. Furthermore, middleware-based solutions provide persistence of messages. Thereby, message delivery can be guaranteed even if the receiving communication end point is not available at the time of message transmission. Message sender and receiver are decoupled and are not required to be active at the same time. Furthermore, an individual receiver instance can be replaced by another without notifying the sender. Lastly, in case of a

failed receiving component, messages can be repeatedly received after restoration. Middleware systems usually provide multiple message delivery services such as at-least-once, at-most-once and exactly-once.

### 6.5. Flexibility Assessment

As increasing flexibility is the main benefit of the presented solutions, we followed a strategy to also compare the flexibility on a simple quantitative level (counting rather than measuring), yet limited it to the concrete testbed described in Section 5. Hereby, we assess the effort necessary to switch between different supply states in the given power grid. Based on the explanations above and the example power grid from Section 3, we count the number of atomic change operations for a given state change in each of the three given approaches. To do so, the power grid model given in Figure 1 is extended with a dedicated communication network consisting of router devices (R0, ..., R4), interconnected in a star topology, as visualised in Figure 10.

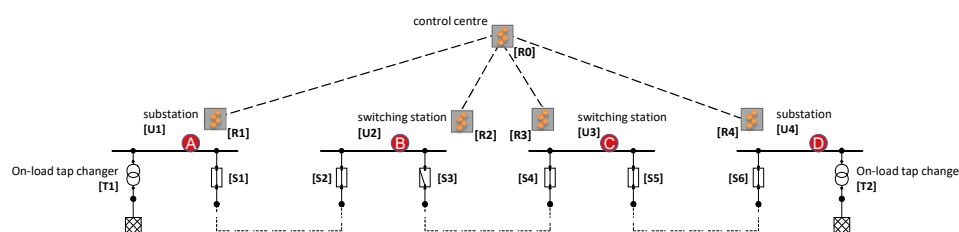


Figure 10. Evaluation scenario.

In this scenario, the supply states describe which power grid segments are supplied by which transformers. This, in turn, corresponds to the states of the switches S1 to S6. A supply state change is related to the opening or closing of one or more of these switches. In order to estimate the effort required to adapt the communication systems to a new supply situation, we measure the number of devices that need to be reconfigured. However, we ignore the effort (such as computation and data transfer time) it takes to reconfigure the individual devices in the three different communication systems, assuming a constant effort for such an atomic operation. This is viable due to three reasons:

- First, at least from a general view, the necessary steps for all three systems are similar.
- Second, a detailed evaluation of these effects would only hold for specific implementations, leading to a lack of generality.
- Third, we assume that an automated reconfiguration system is replacing human operators in the SCADA systems in order to ensure comparability to a certain degree.

With regard to the first aspect, the steps for reconfiguration in the three systems after the connectivity information has been received from the routing gateway component (see Section 5) can be summarised as:

1. Computation of the reconfiguration information for topic associations (Coaty), network switches (SDN), or field devices (classical SCADA)
2. Establishing the network connection to Broker instances (Coaty), network switches (SDN), or field device controllers (SCADA)
3. Installation of the necessary message forwarding or addressing information

In the given scenario, there are seven different supply states depending on which of the power grid segments are supplied by which of the transformers. In six of these states, the power grid is separated at one or more locations leading to the situation that the two transformers are supplying a subset of the grid regions. These situations particularly include those where some grid regions are disconnected from any transformer and, thus, are in a state of blackout. In the seventh state, both transformers are supplying the grid concurrently in so-called mixed operation, where all of the

six switches are closed. The latter is considered the initial state for the evaluation. The seven states including their particular supply situations are summarised in Table 2. Starting from the initial state, we extract the distinct switch operations which lead to a change of the supply situation. The transition table is given in Table 3. It is noteworthy that not all possible switching actions result in a change of the supply situation and in turn require a reconfiguration of the communication system.

**Table 2.** Supply states defined by the supply situation in the example grid.

State	T1	T2	Open Switches
1	A,B,C,D	A,B,C,D	none
2	A	B,C,D	$S1 \vee S2$
3	A	C,D	$(S1 \vee S2) \wedge (S3 \vee S4)$
4	A	D	$(S1 \vee S2) \wedge (S5 \vee S6)$
5	A,B	C,D	$S3 \vee S4$
6	A,B,C	D	$S5 \vee S6$
7	A,B	D	$(S3 \vee S4) \wedge (S5 \vee S6)$

**Table 3.** State transition table including a reconfiguration action count for state transitions within the different approaches.

Opening Switch	Initial State	Next, State	Open Switches Out	Classic	Coaty	SDN
$S1 \vee S2$	1	2	$S1 \vee S2$	4	1	1 / 4
$S3 \vee S4$	1	5	$S3 \vee S4$	4	1	1 / 4
$S5 \vee S6$	1	6	$S5 \vee S6$	4	1	1 / 4
$S3 \vee S4$	2	3	$(S1 \vee S2) \wedge (S3 \vee S4)$	1	1	1 / 1
$S5 \vee S6$	2	4	$(S1 \vee S2) \wedge (S5 \vee S6)$	2	1	1 / 2
$S1 \vee S2$	5	3	$(S1 \vee S2) \wedge (S3 \vee S4)$	1	1	1 / 1
$S5 \vee S6$	5	7	$(S5 \vee S6) \wedge (S3 \vee S4)$	1	1	1 / 1
$S1 \vee S2$	6	4	$(S1 \vee S2) \wedge (S3 \vee S4)$	2	1	1 / 2
$S3 \vee S4$	6	7	$(S5 \vee S6) \wedge (S3 \vee S4)$	1	1	1 / 1

To assess the necessary effort to reconfigure the communication system, we evaluate how many entities need to be reconfigured in order to ensure correct data delivery corresponding to the current supply situation. Table 3 contains the state transitions and shows a reconfiguration count for the three different approaches.

Considering the classical **SCADA** approach, transitions from the initial state 1 to states 2, 5, and 6 require the reconfiguration of the sensor devices  $u1$ ,  $u2$ ,  $u3$ , and  $u4$ . The state transitions from 2 to 3 and 5 to 3 require the reconfiguration of sensor device  $u2$  while the transitions from 2 to 4 and 6 to 4 additionally require the reconfiguration of sensor  $u3$  and  $u2$ . The transitions from states 5 to 7 and states 6 to 7 require the reconfiguration of sensor  $u3$ . Summing up, depending on the concrete state transition, up to four atomic sensor reconfiguration operations are needed for the opening of just one switch.

Compared to this, independent of the concrete state transition, in the case of the Coaty approach, every reconfiguration concerns the broker itself, and thus only one single reconfiguration action is required. Due to this, scalability issues are less probable for the Coaty approach when compared to the classical **SCADA** approach, where dedicated network connections to multiple switches are necessary for configuring the network.

In case of **SDN**, the reconfiguration effort depends on the location where packet manipulation happens. In the given scenario, for example, packet manipulation could either happen at the individual **SDN** switches located at each field station or they might happen at the central **SDN** switch located at the control centre (R0). The decision where to execute packet manipulation depends on overall system automation and several other considerations. If, for instance, a large number of field stations with a large amount of sensors is deployed in a given system configuration, it might be wise to deploy packet manipulation functions directly to the field stations for scalability reasons. When packet manipulation



needs to be done in a high frequency at a central location in the network, the central device could be overloaded. In the first case (reconfiguration at Switch R0), the effort is equal to the effort for Coaty, in the second case (reconfiguration at field stations), the effort is equal to the effort for SCADA. The number of operations is given in the SDN column of Table 3 for both cases.

Particularly, the SDN solution may exhibit growing delay depending on the bandwidth required by sending nodes and the number of flow entries within the flow table of a switch. The middleware-based approach additionally may exhibit an increasing message processing overhead proportional to the number of distributed messages per time interval and the number of sending and receiving nodes. While most recent middleware-based messaging solutions are capable of dealing with large amounts of messages, the potential impact of latency due to message brokering may limit the applicability of middleware solutions when strict latency requirements exist (such as real-time-communication).

While SDN and Coaty provide some advantages regarding flexible reconfiguration of communication links compared to the classical SCADA approach, SDN does not perform very well if the communication between potentially changing endpoints is session-based and maintains the state of a connection. In such a case, the SDN solution needs to be extended with deep-packet-inspection and packet modification capabilities. The middleware-based approach is easier to handle in that aspect, as the communication end points maintain a stable connection to the broker, which assumes the role of the expected counterpart of the end-to-end connection. As we could show in the field tests, this approach works in principle. However, the introduction of the broker as an intermediary instance in what would otherwise be direct end-to-end connections between devices introduces some overhead that might have a negative impact on performance.

An additional complication is imposed by TLS. Regarding long-term adaptability to future innovations of the power grid, both new solutions provide a good starting point. Both approaches have been proven to be applicable and provide flexibility within a wide range of applications. One shortcoming of the SDN solution is that it generally requires SDN capable hardware, which might at some point be replaced by another (incompatible) technology.

## 7. Conclusions

The rising number of prosumers and the limitations of grid capacities lead to an increasingly distributed system of heterogeneous components. The smart grid communication network should be able to deal with dynamic changes in the power grid. Existing smart grid communication networks are not designed for highly dynamic conditions and managing the associated (re-)configuration is a complex, time-consuming and error prone task. We presented a solution based on software-defined virtualisation technologies and a broker based approach. Our use case driven evaluation shows that both solutions are able to handle the increased dynamics expected in future smart grid networks, and are consequently potential candidates for smart grid communication networks.

When comparing both proposed solutions, both can be said to have advantages as well as disadvantages. The broker based solution is easier to implement, as no specific hardware is needed. For the SDN approach, which was based on ONOS, SDN capable hardware is needed. Conversely, the middleware approach increases delay compared to SDN, and is thus not feasible when real-time constraints are of importance. Ultimately, the appropriate virtualisation technology depends on the use case and its requirements.

**Author Contributions:** A.V. defined the overall paper structure and is corresponding author. He furthermore edited the main part of Section 4, and provided a proof reading of the whole paper. A.H. contributed to the abstract the example use case definitions and the derived requirements. Furthermore, he also implemented and described the proposed prototypes. F.v.T. contributed to the overall structure of the paper and contributed to the state of the art description (developments and trends in smart grids), the example use cases and the requirements section, to the SDN prototype and the results section. P.D. contributed to the abstract, to the SDN prototype, and performed proofreading of the entire document. O.L. contributed to Section 5.7 as well as to the SDN prototype and performed proof reading of the entire document. U.P. contributed to the abstract and to the state of the art (Smart Grid

Communication Networks) and performed proofreading of all parts concerned with IT-network technologies. All authors have read and agreed to the published version of the manuscript.

**Funding:** The presented work has mainly been conducted in the research project VirtueGrid, which was funded by the Austrian Climate and Energy Fund (KLIEN) within the program e!MISSION under the project number 858873. Contributions have also been performed within the project InterGrid, which is funded by the State of Upper Austria via the Austrian Research Promotion Agency (FFG) under the contract number 881296.

**Acknowledgments:** The authors acknowledge the contributions made by Tobias Gawron-Deutsch in the definition of the Lab setup.

**Conflicts of Interest:** The authors declare no conflict of interest.

### Abbreviations

SCADA	supervisory control and data acquisition
MAS	multi-agent-system
RTU	remote terminal unit
IED	intelligent electronic device
VM	virtual machine
SDN	software-defined networking
ICT	information and communication technology
IoT	Internet of things
IIoT	industrial internet of things
ACM	advanced monitoring and control
DSM	demand-side management
SP	system protection
HAN	home area network
NAN	neighbourhood area network
AMI	advanced metering infrastructure
WAN	wide area network
PLC	power line communication
BER	bit error rate
VLAN	virtual local area network
GRMRP	generic attribute registration protocol multi registration protocol
MMRP	multiple MAC registration protocol
MVRP	multiple VLAN registration protocol
MPLS	multiprotocol label switching
QoS	quality of service
VxLAN	virtual extensible LAN
LAN	local area network
API	application programming interface
AMI	advanced metering infrastructure
SDECN	software defined energy communication network
MQTT	message queuing telemetry transport
DER	distributed energy resource
ADR	automated demand response
TLS	transport-layer security
GPRS	general packet radio service
HMI	human-machine-interface
MTU	master terminal unit
OLTC	on-load tap changer
FLIR	fault location, isolation and restoration
IP	Internet protocol
SD-WAN	software-defined WAN
P4	programming protocol-independent packet processors
OSI	open systems interconnection
ONOS	open network operating system

REST	representational state transfer
gRPC	grpc remote procedure calls
SGAM	smart grid architecture model
UDP	user datagram protocol
TCP	transport control protocol
MAC	media access control

## References

1. Khan, F.; Rehman, A.U.; Arif, M.; Aftab, M.; Jadoon, B.K. A survey of communication technologies for smart grid connectivity. In Proceedings of the 2016 International Conference on Computing, Electronic and Electrical Engineering, ICE Cube 2016, Quetta, Pakistan, 11–12 April 2016; pp. 256–261.
2. Wen, M.H.; Leung, K.C.; Li, V.O.; He, X.; Kuo, C.C. A survey on smart grid communication system. *APSIPA Trans. Signal Inf. Process.* **2015**, *4*, [CrossRef]
3. Gungor, V.C.; Lambert, F.C. A survey on communication networks for electric system automation. *Comput. Netw.* **2006**, *50*, 877–897, [CrossRef]
4. Galli, S.; Scaglione, A.; Wang, Z. Power Line Communications and the Smart Grid. In Proceedings of the 2010 First, IEEE International Conference on Smart Grid Communications, Gaithersburg, MD, USA, 4–6 October 2010; pp. 303–308, [CrossRef]
5. Katayama, M.; Yamazato, T.; Okada, H. A mathematical model of noise in narrowband power line communication systems. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 1267–1276, [CrossRef]
6. Rieken, D.W.; Walker, M.R. Ultra low frequency power-line communications using a resonator circuit. *IEEE Trans. Smart Grid* **2011**, *2*, 29–38, [CrossRef]
7. Cahn, A.; Hoyos, J.; Hulse, M.; Keller, E. Software-defined energy communication networks: From substation automation to future smart grids. In Proceedings of the 2013 IEEE International Conference on Smart Grid Communications, Vancouver, BC, Canada, 21–24 October 2013; pp. 558–563, [CrossRef]
8. Sydney, A.; Nutaro, J.; Scoglio, C.; Gruenbacher, D.; Schulz, N. Simulative Comparison of Multiprotocol Label Switching and OpenFlow Network Technologies for Transmission Operations. *IEEE Trans. Smart Grid* **2013**, *4*, 763–770, [CrossRef]
9. Sezer, S.; Scott-Hayward, S.; Chouhan, P.; Fraser, B.; Lake, D.; Finnegan, J.; Viljoen, N.; Miller, M.; Rao, N. Are we ready for SDN? Implementation challenges for software-defined networks. *IEEE Commun. Mag.* **2013**, *51*, 36–43, [CrossRef]
10. Rehmani, M.H.; Davy, A.; Jennings, B.; Assi, C. Software Defined Networks-Based Smart Grid Communication: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2637–2670, doi:10.1109/COMST.2019.2908266. [CrossRef]
11. Zhao, J.; Hammad, E.; Farraj, A.; Kundur, D. Network-aware QoS routing for smart grids using software defined networks. In *Smart City 360*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 166, pp. 384–394, doi:10.1007/978-3-319-33681-7\_32. [CrossRef]
12. Montazerolghaem, A.; Moghaddam, M.H.Y.; Leon-Garcia, A. OpenAMI: Software-Defined AMI Load Balancing. *IEEE Internet Things J.* **2018**, *5*, 206–218, [CrossRef]
13. Pfeifferberger, T.; Du, J.L.; Arruda, P.B.; Anzaloni, A. Reliable and flexible communications for power systems: Fault-tolerant multicast with SDN/OpenFlow. In Proceedings of the 2015 7th International Conference on New Technologies, Mobility and Security, Paris, France, 27–29 July 2015; pp. 1–6, [CrossRef]
14. Hastings, J.C.; Lavery, D.M. Modernizing wide-area grid communications for distributed energy resource applications using MQTT publish-subscribe protocol. In Proceedings of the IEEE Power and Energy Society General Meeting. IEEE Computer Society, Portland, OR, USA, 5–9 August 2018; pp. 1–5, [CrossRef]
15. Sacoto-Cabrera, E.; Rodriguez-Bustamante, J.; Gallegos-Segovia, P.; Arevalo-Quishpi, G.; León-Paredes, G. Internet of things: Informatic system for metering with communications MQTT over GPRS for smart meters. In Proceedings of the 2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies, Pucon, Chile, 18–20 October 2017; pp. 1–6, [CrossRef]
16. Jia, K.; Xiao, J.; Fan, S.; He, G. A MQTT/MQTT-SN-based user energy management system for automated residential demand response: Formal verification and cyber-physical performance evaluation. *Appl. Sci.* **2018**, *8*, [CrossRef]

17. Shapsough, S.; Takroui, M.; Dhaouadi, R.; Zualkernan, I. An MQTT-Based Scalable Architecture for Remote Monitoring and Control of Large-Scale Solar Photovoltaic Systems. In *International Conference on Smart Grid and Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2019; Volume 256, pp. 57–67.
18. Abbas, H.; Shaheen, S.; Amin, M. Simple, Flexible, and Interoperable SCADA System Based on Agent Technology. *Intell. Control. Autom.* **2015**, *6*, 184–199. [[CrossRef](#)]
19. Karnouskos, S.; Colombo, A.W. Architecting the next generation of service-based SCADA/DCS system of systems. In Proceedings of the IECON 2011-37th Annual Conference of the IEEE Industrial Electronics Society, Melbourne, Australia, 7–10 November 2011; pp. 359–364.
20. Radhakrishnan, B.M.; Srinivasan, D. A Multi-Agent Based Distributed Energy Management Scheme for Smart Grid Applications. *Energy* **2016**, *103*, 192–204. [[CrossRef](#)]
21. Von Tullenburg, F.; Panholzer, G.; Du, J.L.; Soares, A.; Spiessens, F.; Geysen, D.; Ectors, D. An Agent-Based Flexibility Trading Architecture with Scalable and Robust Messaging. In Proceedings of the 2017 IEEE International Conference on Smart Grid Communications (SmartGridComm), Dresden, Germany, 23–27 October 2017; pp. 509–514. [[CrossRef](#)]
22. Kantamneni, A.; Brown, L.E.; Parker, G.; Weaver, W.W. Survey of Multi-Agent Systems for Microgrid Control. *Eng. Appl. Artif. Intell.* **2015**, *45*, 192–203. [[CrossRef](#)]
23. Dubey, A.; Karsai, G.; Pradhan, S. Resilience at the Edge in Cyber-Physical Systems. In Proceedings of the 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), Valencia, Spain, 8–11 May 2017; pp. 139–146. [[CrossRef](#)]
24. Sanislav, T.; Zeadally, S.; Mois, G.D. A Cloud-Integrated, Multilayered, Agent-Based Cyber-Physical System Architecture. *Computer* **2017**, *50*, 27–37. [[CrossRef](#)]
25. CEN-CENELEC-ETSI Smart Grid Coordination Group—Sustainable Processes, 2012. Available online: [https://ec.europa.eu/energy/sites/ener/files/documents/xpert\\_group1\\_sustainable\\_processes.pdf](https://ec.europa.eu/energy/sites/ener/files/documents/xpert_group1_sustainable_processes.pdf) (accessed on 4 October 2020).
26. Veichtlbauer, A.; Heinisch, A.; von Tullenburg, F. Virtualisierung für Energienetze. In *Newsletter 2020-03—Virtualisierungstechnologien für das Energienetz*; OVE—Österreichischer Verband für Elektrotechnik: Vienna, Austria, 2020. (In German)
27. Aydeger, A.; Akkaya, K.; Cintuglu, M.H.; Uluagac, A.S.; Mohammed, O. Software defined networking for resilient communications in Smart Grid active distribution networks. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 23–27 May 2016; pp. 1–6.
28. ISO/IEC 7498-1:1994 Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model. 1994. Available online: <https://www.iso.org/standard/20269.html> (accessed on 4 October 2020).
29. TheP4LanguageConsortium. The P4 Language Specification. Available online: <https://p4.org/p4-spec/p4-14/v1.0.5/tex/p4.pdf> (accessed on 19 March 2020).
30. Yang, Z.; Cui, Y.; Li, B.; Liu, Y.; Xu, Y. Software-defined wide area network (SD-WAN): Architecture, advances and opportunities. In Proceedings of the 2019 28th International Conference on Computer Communication and Networks (ICCCN), Valencia, Spain, 29 July–1 August 2019; pp. 1–9.
31. Siemens. Coaty—The lightweight Open-Source Framework for Collaborative IoT. Available online: <https://coaty.io/> (accessed on 14 October 2020).
32. Smart Grid Coordination Group. CEN-CENELEC-ETSI Smart Grid Coordination Group—Smart Grid Reference Architecture. 2012. Available online: [https://ec.europa.eu/energy/sites/ener/files/documents/xpert\\_group1\\_reference\\_architecture.pdf](https://ec.europa.eu/energy/sites/ener/files/documents/xpert_group1_reference_architecture.pdf) (accessed on 4 October 2020).

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

## 2.3 Publication 3

A. Veichtlbauer, C. Praschl, L. Gaisberger, G. Steinmaurer, T. I. Strasser:  
**Toward an Effective Community Energy Management by Using a Cluster Storage.**

*In: IEEE Access, Vol. 10, pp. 112286-112306, Oct. 2022*

### Own Contribution

Again, the candidate defined the overall paper structure and acted as corresponding author. He provided the main parts of the introduction, including research questions and methodology, as well as of the related work section. He also contributed to the algorithm and architecture parts. He also provided the multi-household simulation part, and contributed to the field tests. Finally, he contributed to the evaluation and the conclusion sections.

Received 3 October 2022, accepted 14 October 2022, date of publication 21 October 2022, date of current version 28 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3216298



## Toward an Effective Community Energy Management by Using a Cluster Storage

ARMIN VEICHTLBAUER<sup>1</sup>, CHRISTOPH PRASCHL<sup>1</sup>, LUKAS GAISBERGER<sup>2</sup>,  
GERALD STEINMAURER<sup>2</sup>, (Member, IEEE),  
AND THOMAS I. STRASSER<sup>3,4</sup>, (Senior Member, IEEE)

<sup>1</sup>Faculty for Informatics, Communication, and Media, Center of Excellence Energy, University of Applied Sciences Upper Austria, 4232 Hagenberg, Austria

<sup>2</sup>Faculty for Engineering, Center of Excellence Energy, University of Applied Sciences Upper Austria, 4600 Wels, Austria

<sup>3</sup>Electric Energy Systems—Center for Energy, AIT Austrian Institute of Technology, 1210 Vienna, Austria

<sup>4</sup>Faculty of Mechanical and Industrial Engineering, Institute of Mechanics and Mechatronics, TU Wien, 1060 Vienna, Austria

Corresponding author: Armin Veichtlbauer (armin.veichtlbauer@fh-hagenberg.at)

This research work received funding from the Austrian Ministry for Transport, Innovation and Technology (bmvit) and the Austrian Research Promotion Agency (FFG) under the “Smart Cities Demo” program in the Storage Cluster South Burgenland project (FFG No. 858896). Contributions have also been performed within the project InterGrid, which is funded by the State of Upper Austria via the FFG under contract number 881296.

**ABSTRACT** The integration of renewable local energy generation in single households – turning the household into a “prosumer” – is an important way to support an ecological transition of the electric power system. However, due to the volatile and distributed nature of most renewable energy sources, the power system may face stability problems when integrating a large number of renewables. The paper at hand describes an approach to overcome these shortages in a two-fold manner: First, the effects of the installed renewables shall be limited locally to a group of households – a so-called “energy community”. To do so, all the participating households are using existing self-consumption optimization tools. However, when a household has excess energy which can not be consumed locally, this energy is shared among the other participating households by using a cluster storage device, thus enabling a community self-consumption before feeding into the low-voltage distribution grid. Second, the connected operator may request flexibility from the participating households. For that, additional loads or load sheds are triggered by the requesting grid operator, depending on the current situation in the grid. The households decide autonomously about the amount of granted flexibility, receiving respective financial incentives. This work introduces an energy management concept and a prototypical control infrastructure used for the aforementioned functionalities. In a number of simulations and field tests, the proposed approach was successfully evaluated. The article provides a comprehensive overview of the gained results and the conclusions derived from them.

**INDEX TERMS** Cluster storage, energy community, energy management system, flexibility, optimization, renewables, self-consumption, volatility.

### I. INTRODUCTION

The electric power system is currently undergoing a phase of transition towards a so-called smart grid. An important driver hereby is the integration of renewable energy sources [1], like Photovoltaic Systems (PVs), which support decarbonization of the power system and thus allow for an ecologic “energy transition”, i.e., the change of the power grid to a “green” and sustainable infrastructure. However, most renewable energy sources face two important disadvantages: First, they are very

often used with small-scale plants (e.g., roof-top installations on single houses) where many small and highly distributed installations have to be coordinated to ensure the security of supply, as well as the stable and efficient operation of the power grid [2]. Second, they are not controllable, and sometimes even hardly predictable; therefore, means have to be taken to use the energy when it is present, or to store energy (electrically or maybe thermally) for later use [3].

#### A. CUSTOMER ENERGY MANAGEMENT SYSTEMS

At single household level, Customer Energy Management Systems (CEMSs) [4] perform this by either using

The associate editor coordinating the review of this manuscript and approving it for publication was Akin Tascikaraoglu.

controllable loads (electric vehicles, heat pumps, air conditioning, etc.) at appropriate times – thus adhering to a “load follows generation” paradigm – or by utilizing energy storage devices to decouple production and consumption times. In both cases, the goal is to maximize self-consumption and thus minimize the energy exchange with the providing Distribution System Operator (DSO). This triggers economic benefits for households, as sales prices usually are much lower than purchase prices. For DSOs, a minimal exchange with prosumer households eases the control in the Low Voltage (LV) grid, as the power shares of volatile sources in the LV grid are then reduced due to less feed-in.

Beyond that self-consumption maximization, CEMSS additionally allow for providing flexibility [5] to the associated DSO. In the case of a lack of energy in the LV grid which is connected to households equipped with CEMSSs, the DSO operating the LV grid may request a load shedding from the respective households. Conversely, when excess energy is present in the LV grid, additional consumption may be requested by the DSO. In both cases, the several households' CEMSSs decide autonomously if and to which extent these requests are fulfilled. The amount of shifted load can be measured, and financial incentives can thus be given by the DSO to the households dependent on the amount of shifted energy – a strategy called demand side management (DSM) [6]. More details about that can be found in Section III.

### B. ENERGY COMMUNITIES

The power optimization possibilities in LV distribution grids are even higher, when not just considering independently controllable households (i.e., utilizing CEMSSs), but coordinated energy communities [7]. Without coordination, the exchange of energy between the households and the associated LV grid equals the sum of the single exchanges of all households. The more participants are considered, the smoother the consumption curve will be due to the simultaneity factor; i.e., consumption peaks of single households will not have that big effects anymore.

For the generation side, things turn out to be more complex: Usually, power generation is done with roof-top PV installations; in a local community, it can be expected that weather and shadowing conditions are almost similar and thus also the production curves will not differ widely. A difference is yet to emerge in the business model. In some cases, some of the households will feed into the LV grid, whereas others are consuming from the grid. In such a case, the producing and consuming households can be (logically) balanced, before the exchange with the LV grid is done (even when having multiple real connections).

When considering a much higher purchase price than selling price for the single households in their respective contracts with the DSO operating the LV grid, an internal balancing [8] using an internal transfer price in the middle of the DSO prices would let both selling and purchasing households benefit. However, this effect may reduce the amount of energy

exchanged between the LV grid and the energy community, but does not change the total consumption and/or generation of the community at a certain point in time.

When additionally adding community storage(s), this may be changed. Producing households may feed into the storage up to a certain technically given upper limit, whereas consuming households may use stored energy up to a certain technically given lower limit. In this case, all households may feed in or use stored energy at the same time, as long as the upper and lower limits are kept. This may lead to an additional time shift in the usage of self-produced energy, which can not be accomplished just with a local storage system. Such effects are described in Section V-C.

### C. RESEARCH QUESTIONS AND METHODOLOGY

In this work, results of a simulative and practical assessment of energy optimization in an energy community are presented. An appropriate prototypical implementation was realized to answer several Research Questions (RQs) in the context of managing energy communities with a high share of renewable Distributed Energy Resources (DERs), which are listed in the following:

- RQ1: To what extent can energy communities of multiple consumer/prosumer households contribute to the stability of LV grids? Which metrics can be used to assess these flexibility offers also quantitatively?
- RQ2: How realistic are the measurements of these contributions, considering the natural volatility of both electric consumers and renewable energy sources? Can a realistic baseline be defined to assess the effects of the control algorithm, and can this baseline be validated against real environments?
- RQ3: Which additional stability and economic effects can be gained by using a cluster storage in the multi-household community?
- RQ4: Which performance of the underlying Information and Communication Technology (ICT) infrastructure is needed to allow a smooth operation of the intended flexibility system? What timing requirements are needed for the controller? Which hardware resources are required? How can a scalable architecture be set up?
- RQ5: Which coupling strategies can be used in that architecture? What effects do these strategies have on the installation effort of new appliances? How can necessary state information be handled in the controller?

Some preliminary simulation results on these RQs have already been published in [9]. In opposition to that paper, in this work, a more realistic simulation environment is being used. First, a Matlab/Simulink [10] model has been utilized which contains close-to-reality physics for the simulations presented here, in opposite to very basic linear models as used in the work-in-progress paper (e.g., for environmental parameters as room temperature). Second, the simulation scenarios have been adopted to more realistic input patterns for the simulation; this regards usage patterns for the considered

appliances and other input parameters like weather conditions.

After successfully finishing the simulations, several field tests have been conducted to complete the validation of the realized Proof of Concept (PoC) implementation. For this purpose, a real-world testbed has been set up in a mixed commercial and private building in the village of Stegersbach, South Burgenland, Austria, as described in Section IV-A. More details on the simulations and field tests can be found in Sections V and VI respectively.

The remaining parts of this article are structured as follows: Section II gives an overview of the current state of the art in the areas of electric power grids and related ICT infrastructures. The algorithmic basis of the controller application is given in Section III. In Section IV, the system architecture of the testbed in Stegersbach, South Burgenland, Austria, is briefly described, accompanied by a short depiction of the controller application. The validation tests of the controller and its system context are explicated in Sections V and VI respectively; also, the scenarios for the validation and evaluation of the realized controller are defined here. In Section VII, a discussion of the gained results and an evaluation of the system behavior is done; finally, conclusions and outlooks to further possible research and development activities are given here.

## II. RELATED WORK

As the work at hand combines ICT infrastructures with smart grid applications, the discussion of relevant academic and industrial research and development undertakings can be split into (i) an “energy part”, dealing with the necessary functionalities for distributed control logic for several appliances of interest, and (ii) an “ICT part”, dealing with the calculatory power of participating nodes (which is to a great extent depending on the used software stack on the respective devices), as well as with connectivity issues of the underlay and overlay network infrastructures. These two aspects are considered in the following subsections; followed by a particular consideration of Quality of Service (QoS) issues, as these are crucial for many grid-related applications. Thus, the actual benefit and challenge of the current work is to utilize interdependences between these fields. For instance, timely and secure transport of control commands to actuators allows for distributed control strategies. Much existing research is done in one of these fields, but lacks the consideration of interdependences, especially in practical environments – as will be discussed in the following.

### A. CHALLENGES FOR FUTURE POWER SYSTEMS

The integration of volatile and distributed renewable energy sources is one of the big challenges to the evolving smart grid [11]. Control of these DER is explored in [12]. Reference [13] shows how renewable energy forms can be integrated on a local level. In [14], the integration of renewable energy sources into a cooling, heating, and power grid in urban areas is explored in order to be able to derive statements

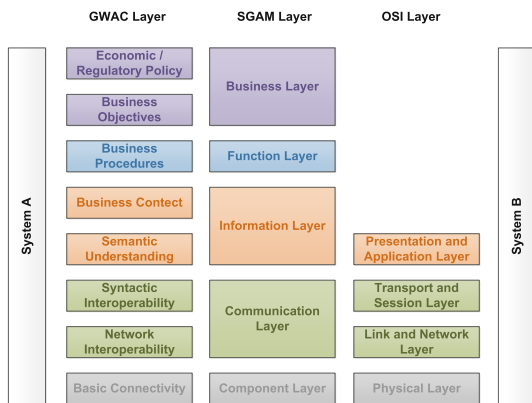
about how and under which conditions the share of renewable energies can be increased. The volatility of renewable energy sources such as PV and wind presents particular challenges to DSOs. To keep the LV distribution grid stable, it is common to rely on Demand Side Management (DSM), i.e., influencing consumption depending on the energy currently being generated [15]. In addition to direct control of the devices by the DSO, there is also the possibility to leverage energy flexibilities by utilizing local control facilities of suitable devices in individual households [16].

In this context, a basic prerequisite is to avoid loss of comfort, for example by optimizing charging times of electric cars while ensuring timely charging [17]. Due to decreasing feed-in tariffs, optimization of self-consumption is an essential topic [18]. The concept of renewable energy communities is considered to be of immense importance. This concept is not very new – as early as 2015, more than 1000 projects were dealing with the topic of energy communities [19]. The constellations investigated are primarily concerned with bilateral exchange and the storage of energy in order to increase the self-consumption rate of the community [20]. In addition, energy communities allow DSOs additional room for maneuvering. Further projects deal with the optimization in an energy community taking into account the current energy price [21].

Besides control and flexibility issues for DSOs, another important aspect of the integration of renewables is self-consumption optimization. This is a central motivation for private households as well as energy communities, as the feed-in tariffs to sell energy to a DSO are usually much lower than purchase tariffs. This has been an important research topic over the last decade (e.g., [22]), but also commercial developments (e.g., from PV vendors) have incorporated appropriate technologies as offers for their customers. Machine Learning (ML) technologies [23] play an important role in this context. In many cases, demand and/or supply prediction is utilized for these optimization purposes (e.g., [24]), allowing for time-shifting the use of appliances that provide sufficient flexibility.

All of these smart grid applications turn the grid into a complex and multi-vendor system of systems, which needs appropriate mechanisms for interoperability; especially, communication between different subsystems has to be ensured [25]. For that purpose, a plethora of standards have been developed for different use cases and applications, as shown in [26]. Interoperability hierarchies have been developed to structure the collaboration issues and standards, like the GridWise Architecture Council (GWAC) stack [27]. Mappings to existing layered abstraction models like the Open Systems Interconnection (OSI) protocol stack [28] defined by the International Organization for Standardization (ISO) are thus possible, as depicted in Figure 1. Yet, the GWAC model extends the OSI model to even higher abstraction levels regarding business and application interoperability, which go beyond the possibilities of technical communication protocols as covered by the OSI stack.





**FIGURE 1.** The mapping between GWAC and SGAM layers can already be found in [29]; additionally, a mapping of the “network interoperability” GWAC layer to the lower OSI layers can be identified, as well as a mapping of the “syntactic interoperability” GWAC layer to the higher (end-to-end) OSI layers.

Based on that, meta-architectures like the Smart Grid Architecture Model (SGAM) [29] have been defined; the SGAM complements the interoperability hierarchy with domains along the power distribution chain, as well as zones representing different roles in the automation pyramid [30]. To ease the interoperability on a practical level, collaboration platforms as Common Object Request Broker Architecture (CORBA) [31] and automation protocols as Open Process Control Unified Architecture (OPC UA) [32] have been developed. Yet, they do not allow for an application layer-specific course of actions – e.g., OPC UA implementations just allow for simple request/response communication patterns. On local level, frameworks such as OpenHAB [33] enable the integration of several hardware and software components, but do not enable cross-platform communication.

### B. GRID-RELATED ICT INFRASTRUCTURES

The focus of the aforementioned work is mainly set on algorithmics and control logic. Other projects aim at a flexible and expandable infrastructure, e.g. to acquire the necessary sensor data. Also, this topic has been researched for many years, e.g. in [34]; however, still, no generic infrastructure suitable for the whole variety of grid applications has been developed so far [35]. Rather, tailor-made approaches have been developed for several application domains such as grid control, metering, etc. Some of these approaches are discussed in the following.

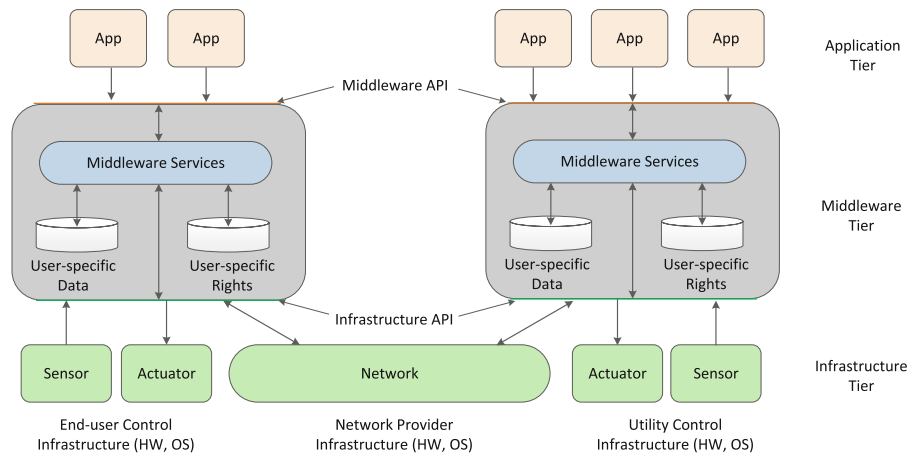
The first and basic requirement on the ICT infrastructure is to provide connectivity between participating nodes (the Intelligent Energy Devices (IEDs)). Traditionally, dedicated lines have been installed for that purpose, i.e., copper or (in newer installations) fiber cables along the high and medium voltage lines. On top of that physical infrastructure, often protocols like International Electrotechnical Commission (IEC)

60870-5-104 [36] are used for coordination at operation level (as depicted by the SGAM model [37]) of operating DSOs or Transmission System Operators (TSOs). Newer installations (especially with DSOs) often use IEC 61850 [38] for substation communication. Wide area coordination, even with dedicated systems, is usually based on Internet Protocol (IP) stacks, i.e., Transmission Control Protocol (TCP) [39] or User Datagram Protocol (UDP) [40] over IP [41], whereas on local levels (e.g., at station level) also field buses may be used for communication, using other address formats than IP to address IEDs.

For the LV grid, usually, no communication lines have been installed. This makes the coordination in the LV grid particularly challenging, as the demand for distributed control increases with the number of participants (especially due to the volatile and highly distributed nature of home installed PV systems). Also, applications like smart metering require appropriate infrastructure to be able to deliver meter data in timely and secure manner. Smart grid applications do not only need an exchange protocol; furthermore, the data definition itself has to be specified in advance. Standards are important here obviously; however, the plethora of different applications results in a variety of associated standards, as outlined by [25]. In the case of metering, the Device Language Message Specification (DLMS)/Companion Specification for Energy Metering (COSEM) suite [42] provides a very widespread and common solution for data definition as well as protocol actions.

For ensuring connectivity in the LV grid, three approaches are common: First, infrastructure may be newly set up to meet the communication requirements – in most cases, this is done via cellular mobile radio. Second, the existing power line infrastructure is used for communication purposes also (Power Line Communication (PLC)). Third, other existing connections (home Internet) are used for power applications such as metering. In the latter case, a public communication infrastructure like the Internet is used as an underlay network, which hosts a private overlay network. The traffic of this overlay network has to be encrypted and strictly separated from other kinds of traffic in the underlay to ensure the required security and QoS.

In Local Area Networks (LANs), the simplest approach for traffic separation is the use of Virtual LANs (VLANs), as described in [43]. As VLANs are working only locally (in a switched infrastructure), this is feasible for instance in intra-substation communication. However, for use in Wide Area Networks (WANs), extensions are necessary. Virtual eXtensible LANs (VXLANS) [44] provide an overlay approach as mentioned above; i.e., LAN frames are encapsulated and tunneled over the underlying public Internet. This is an easy way to provide traffic separation in a wide area network, and with encrypted payload also security can be provided to a sufficient level (at least for the transport – the end systems have of course to be secured additionally). However, the big downside of this lightweight approach is the lack of QoS support, especially concerning real-time data.



**FIGURE 2.** X-Architecture model depicts a common middleware layer that provides standardized services over a dedicated API to diverse applications; moreover, infrastructure elements such as communication means and hardware access are utilized by the convergent middleware layer.

### C. QUALITY OF SERVICE ISSUES

The usual way to handle QoS requirements in Internet-based environments is the use of Multi-Protocol Label Switching (MPLS) [45]. Thereby, MPLS paths are defined in advance and can be associated according to QoS requirements. These requirements are encoded in different “Differentiated Services Codepoints (DSCPs)”, representing different traffic classes. Routers can thus implement their respective strategies (Per Hop Behavior (PHB)) to forward packets depending on the DSCP. The setup of MPLS paths can also be used for Traffic Engineering (TE), allowing grid operators to have better control of their respective traffic flows (using a technology called MPLS-TE). However, as MPLS paths have to be ordered from service providers in advance, a lack of flexibility remains the biggest issue in using that technology for future grids.

Software Defined Networking (SDN) [46] is seen as a solution to overcome this issue, while providing the same potentials regarding security and QoS. SDN comes along with a separation of forwarding plane (being responsible for forwarding packets, as determined by a set of forwarding rules) and control plane (being responsible for defining the forwarding rules and sending them to the forwarding plane devices, i.e., the SDN switches). For the control plane, central devices (SDN controllers) are used; often, only one controller is servicing a whole SDN network. Yet, if availability is an issue, at least one backup controller should be used in order to minimize downtimes.

The rules are based on frame and packet headers; i.e., header fields like Medium Access Control (MAC) addresses, IP addresses, TCP port numbers, etc. can be used as distinguishing parameters by the rule set. Application-specific data (payload) however is not evaluated by SDN. To allow for forwarding based on payload data, “Programming Protocol-independent Packet Processors (P4)” [47] could provide an

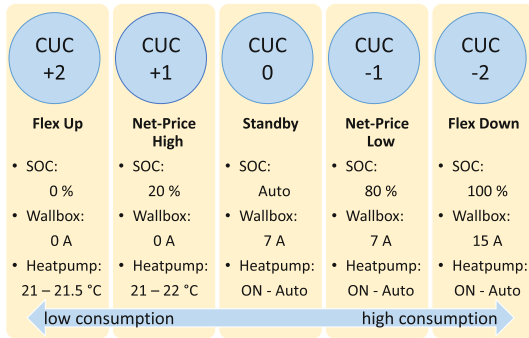
alternative; yet, the number of devices supporting that technology is still exiguous. In summarizing, SDN provides sufficient support of QoS for power ICT infrastructures, but still lacks some desired flexibility features. Additionally, as usually public underlay infrastructure is used, the security topic has to be taken into account. In this context, much literature already exists (e.g., [48]); this is not in the focus of the work at hand, yet best practice solutions such as encryption (e.g., [49]) are of course considered.

From an architectural point of view, many existing frameworks and approaches are based on an X-Architecture model (e.g., [50]) as depicted in Figure 2. These approaches define a convergence layer in the middle, which may be used by a number of applications over well-defined interfaces [51]. Often, these applications are sandboxed, i.e., they are running in a closed environment and can then communicate only via these system interfaces, allowing for a controlled environment and thus assuring security and privacy guarantees. A more detailed overview of infrastructural research can be found in [52] (focusing on connectivity issues) and in [35] (focusing on interoperability frameworks).

The work at hand now tries to combine these algorithmic and infrastructural approaches to a comprehensive PoC solution and to validate this PoC in simulations as well as in field trials. The innovation of this work lies in the usage of a general purpose infrastructure (Internet as a public underlay technology, Message Queuing Telemetry Transport (MQTT) as messaging protocol) for specifically distributed control logic, which can be tested in real-world scenarios including long-term tests, stress tests, and scalability tests.

### III. ENERGY MANAGEMENT ALGORITHM

The proposed algorithm for the envisaged solution is planned to act on different scenarios and their respective requirements, which are set by the cluster operator. This cluster operator



**FIGURE 3.** Cluster Use Cases are defined in the basic control algorithm, having a defined state of charge of the battery (SOC), maximum allowed charging current at the wallbox and room temperature levels to be maintained by the heat pump.

can be a human, organizational or technical actor that acts independently from the households and usually pursues maximum profitability in cluster operation.

#### A. CLUSTER USE CASES

The different scenarios in the cluster and the adherent LV distribution grid are called Cluster Use Cases (CUCs) throughout the concerned research project and represent different control strategies for all the devices within all households of the concerned cluster community. Triggers of the DSO are used to either increase the power consumption and/or decrease the power generation because of an energy surplus in the LV grid, or to decrease the consumption and/or increase the generation because of an energy lack. In addition to that, there is also the standby level, which represents an equilibrium in the adherent LV distribution grid. This CUC triggers no action on the cluster participants, as the DSO has no flexibility requirements for the time being. Hence, it may be used for self-consumption optimization of every participating household in the cluster. Depending on the situation in the LV grid, the CUCs have been defined (sorted from energy lack in the LV grid to energy surplus) as shown in Figure 3.

The CUCs are defined in such a way that in CUCs +1 and +2 the energy demand of the household should be reduced as much as possible, whereas in CUCs -1 and -2 the opposite should be the case. The main difference between the price-controlled CUCs (+1, -1) and the flexibility-controlled CUCs (+2, -2) is the capacity provided by the energy storages including the batteries of Electric Vehicles (EVs), as well as the possible temperature band for space heating.

By means of presetting the “Flex” CUCs, the flexibility potential is allowed to be used for as long and as intense as possible. Also, more profound interventions to counteract the imbalances are possible (e.g., direct load shedding with technical means for devices allowing this technology, triggered by the DSO). However, in order to prevent loss of comfort, the used control algorithm is free to not or only partially implement the requested flexibilities. Of course, this will lower the incentives given by the DSO to cooperative customers.

**Algorithm 1** The Base Algorithm Consists of an Endless Loop, Which Is Executed Once per Minute. The Actual Logic Is Implemented in the *calculate* Part. It Is Dependent on the Strategy, Which Is Again Defined by the Current CUC. The CUC Changes and Wallet Calculations Are Done Only in the First of 15 Loops Within a 15 min Timeslot

```

1: while true do
2:   for all datapoint ∈ sensors do
3:     dv ← read(value(datapoint))from broker
4:   end for
5:   for all datapoint ∈ config do
6:     sv ← read(value(datapoint))from file
7:   end for
8:   for all datapoint ∈ actuators do
9:     value(datapoint) ← calculate(strategy, dv, sv)
10:    write(value(datapoint))to broker
11:  end for
12:  if first loop in timeslot(n) then
13:    strategy ← read(value(CUC))from broker
14:    Pzero ← Pcurr
15:    Pcurr ← Average(Ptotal ∈ timeslot(n - 1))
16:    update(wallet(Pzero, Pcurr))
17:  end if
18:  wait(1min)
19: end while

```

#### B. BASE ALGORITHM

The base algorithm cyclically (once in a minute) checks the sensor inputs from the respective devices in a (sub-) household, calculates the set values for the actuators and sends them back. The calculation strategy depends on the current CUC. CUCs are updated in 15 min intervals, so the strategy has to remain the same within this 15 min timeslot. That does not mean, however, that the set values for actuators have to keep their values. If other inputs change (e.g., the room temperature exceeds a limit), the controller will react on that, even when using the same strategy. As the controller has no notion of the exact time, when input values are arriving, these values are stored locally and used only when the controller is active. Thus, it is possible that values are overwritten within the 1 min waiting time, such that only the latest measurements per datapoint are used. This has no influence on the controller’s basic functionality; however, as values are also persisted in a database for later validation (as described in Section IV), it may cause performance issues when too many sensor values are transmitted.

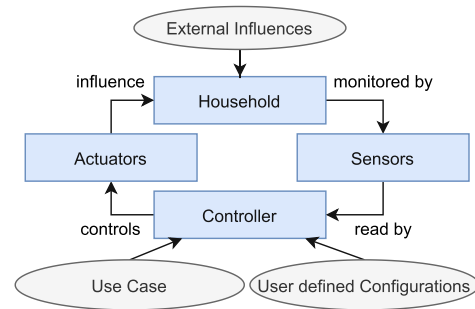
In Algorithm 1, an overview of the basic loop is depicted in pseudocode. Hereby, the *broker* is the central communication instance for the MQTT data exchange with the controller (as described in Section IV), the *calculate* function derives the set values for actuators based on measurements and the current CUC (as described above).  $P_{zero}$  and  $P_{curr}$  are used for the “wallet” calculation, as described in Section III-C,  $P_{total}$  is the total measured power at a given measurement time,  $dv$  are the locally stored dynamic measurement values, and  $sv$  are the

static configuration values, which are also stored locally in a *config file*.

The depicted algorithm is simplified for ease of understanding. Especially, the reading of measurement values is done independently from the main loop (as an own thread) in reality. As MQTT is working on a publish/subscribe basis, the reading occurs whenever the broker sends out values in subscribed MQTT channels. Yet, due to the limitations on the broker side mentioned in Section IV-A, these sensor measurement messages are sent exactly once per 1 min cycle. The CUCs are specified by the DSO in 15 min intervals; the new CUCs are sent out at the beginning of each timeslot. The control strategies for each of the named CUCs are defined as follows:

- 1) CUC +2 (“*Flex Up*”): The energy supply network is in a strong imbalance; in this case, the demand is much higher than calculated. Load shedding should be performed wherever possible.
- 2) CUC +1 (“*Net Price High*”): In this case, the power grid is only slightly skewed. When this CUC is sent to the households, the participating customers should decrease consumption in order to purchase less energy from the DSO or increase the feed-in. Balance is to be created based on the current electricity price.
- 3) CUC 0 (“*Standby*”): No requirements are transmitted to the individual households. Accordingly, the specific subscriber systems do not adapt their mode of operation to a higher-level requirement. Self-consumption optimization, initiated by devices like inverters or energy storages, may take place; otherwise, the devices are operating under their usual parameters.
- 4) CUC –1 (“*Net Price Low*”): This is the opposite use case to CUC +1 and means that there is a slight energy surplus in the LV grid. Controllable consumers should be turned on, or operated with higher power (e.g., by increasing the charging power of EVs).
- 5) CUC –2 (“*Flex Down*”): Here, the production is much too high; i.e., consumption in the LV grid should be drastically increased. Every flexible energy sink (such as batteries and thermal storages) should be operated at maximum power. Additionally, a lowering of production in DER sites could be considered.

Changing DER production is considered to be the ultimate mechanism. In the tests with our prototype, we did not implement this control strategy, as connected PV systems have to provide such control by default (such that no additional value can be derived for the prototype at hand). Furthermore, production could only be limited, but the natural value of power production could not be further exceeded; thus, the treatment would be asymmetric. However, in this work, we used equal treatment for both imbalance directions, by delaying or advancing power consumption, as far as possible. Nevertheless, some appliances (especially EV charging) can not be advanced easily, as they usually start consumption right after plugging. Thus, the behavior may be not completely identical for power shortage and power overflow.



**FIGURE 4.** Control algorithm reads the current state of a household using sensors and adapts it using different actuators based on cluster commands and user-defined configurations. In addition to the controllable actuators, there are also external influences on a household that affect its state, such as the ambient temperature or the position of the sun.

Also, we did not implement self-consumption optimization, as this is an additional research question, which is not the main focus of this work (as depicted in Section I). Yet in the community simulation, we considered common storage for all participating households, which can be used for intra-community trading. Thus it allows for dealing with local imbalances, without interfering with the DSO. Electric energy storage devices hence by nature provide optimization to a certain extent. Finally, our prototype never overrides the built-in limits of the household appliances. Especially, safety measures (e.g., temperature limits of hot water storages) are kept fully functional; i.e., control is done only within normal operating parameters of the participating appliances.

### C. CONTROLLER

The controller is the instance used to apply these cluster-wide control strategies, as well as user-defined configurations for an associated household. From a software side perspective, it is a monolithic component representing the control algorithm, as shown in Figure 4. It reads multiple sensors, which monitor the current state of the household. Based on the sensor values, the current CUC, and additional user-defined configurations, the controller sets different actuator values, which in turn affect the household’s state. Of course, external influences such as the ambient temperature or the position of the sun also have been considered.

This is especially important for the simulations, as these external influences have to be modeled as accurately as possible. As many system parameters show some inertia, i.e., the influence of actuators may show only after a certain delay, the controller is executed only once per minute. In practical tests, this granularity turned out to be sufficient (see Section VI). Due to the fact that the appliances of a household offer different actuators, the controller needs to be able to control each appliance based on its type. Therefore, the system needs to know the following information:

- Which devices are available in the household?
- Which sensors and actuators can be accessed?

- What are the current values of the accessible sensors?
- What is the current state of additional configurations (e.g., preferred charging times for EVs)?
- What is the current CUC?
- How shall the appliances be controlled (based on the current state information)?

Based on the aforementioned CUCs, concrete control strategies are implemented as follows. As mentioned, no actions are taken for CUC 0; i.e., the native appliance control logic implemented by the respective vendors applies. Native (vendor defined) safety and security limitations of appliances also apply to each CUC.

#### D. USE CASE STRATEGIES

For the CUC +2, electric storages are discharged with 2 kW to 0% State of Charge (SOC). Wallboxes are interrupting all charging processes with the only exception of “urgent charging” (for user convenience, activation takes place in order to have a charged EV available at a user-defined departure time on individual user requirements). When the room temperature is within the user-defined tolerance band, the heat pump is set on pause mode and the room temperature is untouched by the heating system until it is as low as the pre-defined lowest tolerable limit.

For the CUC +1, electric storages are discharged to a pre-defined lower limit; the default value for that limit is set to 20% SOC. Wallboxes are deactivated again with the exception of immediate charging processes triggered by EV users. When the room temperature is in the defined band, the heat pump is again paused, as in CUC +2; however, the upper limit is slightly higher than in CUC +2, thus keeping the heating mode for a longer time (this allows for a bigger hysteresis).

For the CUC –1, electric energy storages are charged with 2 kW up to a pre-defined upper limit; the default value for that limit is set to 80% SOC. Wallboxes are triggered to charge with a defined charging current. Depending on the current SOC, cars may use flexible values lower than calculated by the controller. Heat pumps are enabled for heating operation. The concrete heating cycles are performed according to the internal control algorithm; i.e., in that case, the controller does not influence the heating (just like with CUC 0).

For the CUC –2, electric storages are charged up to 100% SOC. Wallboxes are operated at the maximum charging current and thus also at the maximum charging power (as the voltage is fixed in LV grids). This of course depends on the electrical installation, as well as on the capabilities of the EV. Usual values for home installations are 22 kW/32 A or 11 kW/16 A. Finally, the heat pump is enabled for heating operation (again, heating cycles are performed according to internal control).

As mentioned, users may manually override the settings of the controller in some cases for convenience reasons; especially, the urgent charging of EVs may have a big impact on the effects of the controller. Also, all the band limits for temperature and SOC are given by the system customers.

Thus, an incentive system called “wallet” has been created, which rewards cooperative users while doing nothing when users counteract the control goals. Hereby, not the absolute power consumption is considered, but the changes in power consumption of users’ appliances as a reaction to the flexibility request.

For each 15 min timeslot, an average power value is calculated ex post ( $P_{curr}$ ). When the new timeslot starts with a changed CUC (such changes are only possible at the beginning of a new timeslot), the current value of  $P_{curr}$  is stored as a reference value  $P_{zero}$ .  $P_{zero}$  is valid for the time being without further changing the CUC. After finishing new timeslots, again new  $P_{curr}$  values are calculated, and the differences to the reference value  $P_{zero}$  are determined.

Differences in the “wrong” direction (load shedding in negative CUCs, load adding in positive CUCs) are ignored (no penalties are given); however, differences in the “right” direction are rewarded with a pre-defined amount of money per changed energy. Hereby, fluctuations within a timeslot are also ignored; i.e., constant power is assumed within a timeslot, allowing direct calculation of the energy changes (in kWh) out of the power differences:

$$E_{flex} = |P_{curr} - P_{zero}| * 0.25 \quad (1)$$

Here,  $E_{flex}$  is the energy flexibility provided to the DSO in exchange for the flexibility incentive (the “wallet”).  $P_{curr}$  and  $P_{zero}$  are used as defined above. The factor 0.25 is derived from the duration of a CUC timeslot (which is a quarter of an hour) in  $h$ .

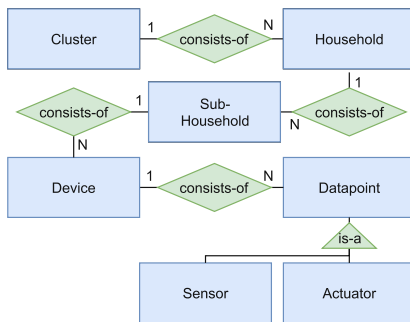
#### IV. CLUSTER ARCHITECTURE

This section describes the different architectural considerations of the presented community energy management system in the following terms:

- The system architecture (the structure of the intended storage cluster) contains the system components and their interrelation.
- The communication architecture depicts the communicating nodes of that system and their used protocols.
- The energy grid architecture contains the consumers, producers, and storage elements of the community at hand, as well as the energy flows in-between these power nodes.
- The software architecture of the central logic component (i.e., the controller device) holds the logic of the distributed control system.

##### A. STORAGE CLUSTER SYSTEM ARCHITECTURE

Storage clusters as described in this work are used to manage the power consumption as well as the generation of multiple controllable households depending on the grid’s energy prices and demands. A household may in turn be separated into multiple sub-households. This concept is introduced (i) to be able to map multi-family households with different optimization goals and (ii) to separate devices that can actively be controlled (e.g., heat pumps, battery energy storage systems)



**FIGURE 5.** Structure of a storage cluster can be represented in form of an EER diagram that consists of multiple households, each of which may be separated into sub-households. A sub-household represents a set of devices with readable and/or writable datapoints in form of sensors and actuators.

from other power consumers that can only be monitored, usually using smart meters (e.g., TV, washing machines). Sub-households however usually share one CEMS; i.e., one central instance which is containing the controller logic as well as the gateway for external communication.

Every sub-household is a set of controllable or non-controllable units that offer different datapoints. They usually represent a readable sensor value or a setpoint writable to an actuator. However, some static configuration parameters (e.g., type of participating EVs and their battery capacities) or user preferences (room temperatures, preferred charging times, etc.) also have to be considered. Finally, the current CUC and wallet-related data ( $P_{zero}$ ,  $P_{curr}$ ) are relevant to the controller and have to be communicated in a proper (i.e., secure and timely) manner. In Figure 5, the most important components and their logical interrelations are shown in form of an Enhanced Entity Relationship (EER) [53] diagram.

The main system components and the used communication links between them are outlined in the following. Basically, a backend infrastructure (the “SPC Test Server”) is connected to a number of test households (for the field test bed, two households have been considered; for simplicity, only one is depicted in the figure) via a public cloud. Each test household comprises a number of (controlled) devices, as well as an edge device implementing the CEMS functionality. As edge device, a Simatic IOT2050 Internet of Things (IOT) gateway has been used; it is equipped with an ARM TI AM6528 GP dual-core processor, and holds 1 Gigabyte (GB) of Random Access Memory (RAM). This edge device contains the actual controller, which manages the devices within a household, as outlined in Section III.

Additionally, the edge device holds device drivers for all controlled devices in order to receive sensor information and to set actuator values, usually utilizing technologies like Modbus, ZigBee, etc. The device drivers translate these data exchange formats to MQTT messages, which are sent to and received from the controller via an MQTT broker which may be located anywhere in the cloud.

In later implementations, the broker instance is planned to reside within the edge device to avoid unnecessary cloud communication; however, for testing purposes, one central broker instance had been utilized. This has been done for simplicity reasons, as the central broker instance is also used for forwarding necessary central data from and to the back-end (e.g., the indication of the current CUC, changing user preferences, or also wallet data).

In older implementations, a usual standard Personal Computer (PC) had been used to hold the CEMS functionality, which worked without problems. For the edge device, however, we faced difficulties when processing all MQTT messages that the broker provided, especially due to the fact that some sensors produced many more measurements than actually needed. As we identified the limited RAM as main origin of these difficulties, we limited the sensor data to one message per datapoint and cycle. As we are storing measurement data in local variables and using these variables as input for the calculations in each cycle, only the latest measurement per cycle is used. Thus, this limitation had no effect on the functionality of the control algorithm.

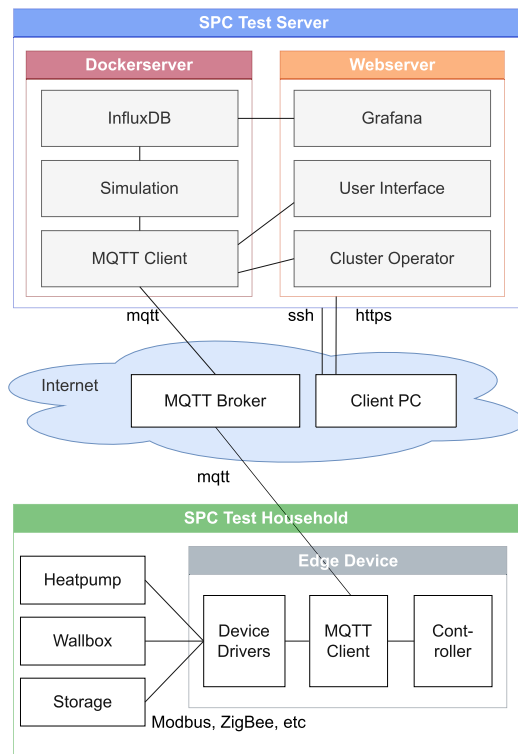
The backend server contains an MQTT client to communicate with the MQTT cloud broker (which could optionally also reside at the backend itself). An appropriate user interface allows households to define user settings like charging times or temperature preferences. Furthermore, a cluster operator can set the current CUCs for the households in a cluster in order to request energy flexibilities from the participating households. These CUCs are distributed using MQTT cluster command messages, which have to be acknowledged by the household controllers.

Both for end users and for operators, Web interfaces are available at the test server. The system thus allows adding additional services like a monitoring service, time series databases, or visualization dashboards to show temporal differences of the measured values. Specifically, Prometheus [54] is used for the monitoring of the system, as well as an InfluxDB time series database [55] together with Grafana [56] as visualization tool for later validation and evaluation of measurements from households’ appliances.

Furthermore, some environmental data (e.g., outside temperatures) are used as input for simulations; again, the simulations can store their results in the time series database. In Section V, a couple of simulations are sketched which have been performed based on that simulation environment. Some of the components mentioned here have been realized as Docker containers [57] in order to allow for a later porting to other system components. This does not affect the functionality, however. Figure 6 gives a brief overview of the mentioned system components and the communication protocols used for their interactions.

## B. COMMUNICATION INSTANCES AND FLOWS

In this work, a loosely coupled architecture for a storage cluster system is proposed; i.e., the inner logic of the controller should not reflect the way, appliances are connected to the



**FIGURE 6.** This figure outlines the main components of the field testbed in Stegersbach. Only one test household is depicted here; a second household has been connected in an identical way. The system architecture contains physical devices (e.g., the heat pump) as well as controlling devices; the protocols of the main communication paths are also given.

system. However, at least the datapoints must be known to the controller, as the appliance control is based on them. Yet, further details should be hidden to the controller communication. Also, best practice solutions for secure communication should be used, even when security is not the primary focus of this work.

As mentioned, this is ensured by using the messaging protocol MQTT [58]. MQTT is a publish/subscribe network protocol that allows broadcasting messages for given topics via an additional decoupled message broker. These topics can in turn be subscribed by other communication participants to receive the published messages from the MQTT broker. MQTT also provides sufficient security by using Secure Socket Layer (SSL) connections as well as client authentication and authorization.

To be able to clearly identify the different types of messages that are broadcasted in the system, the following hierarchical topic structure has been developed:

```
<prefix>/<clusterid>/<type>/<gateway>/
<subhousehold>/<asset>/<datapoint>
```

Such, an MQTT topic reflects the cluster structure shown in Figure 5 and builds up on different placeholders, where `prefix` and `clusterid` uniquely identify the addressed

storage cluster. The following subtopic `type` defines the message type and differentiates between sensor, actuator, actuator acknowledgment, cluster command, and cluster acknowledgment messages. The addressed edge device of a household is associated in `gateway`. An edge device may combine multiple sub-households, which are in turn represented by the `subhousehold` section. The last two parts link the message with the used `asset` and its concrete source or target `datapoint`.

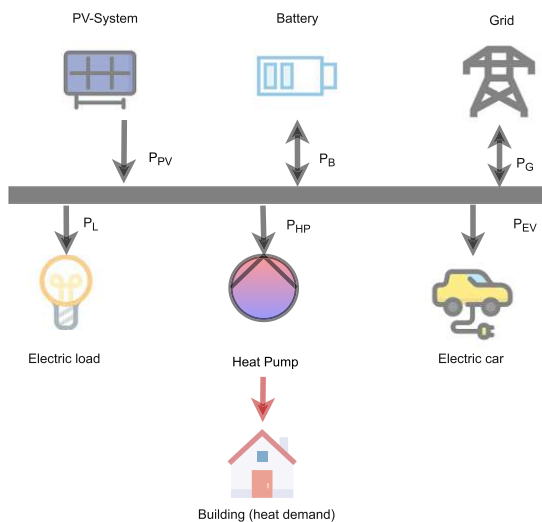
An asset thus aggregates multiple datapoints depending on the message type. While for cluster commands, the CUC may be the addressed asset, for sensors and actuators this placeholder is used for the actual physical appliance. Messages are exchanged within the system based on this topic structure. The message structure itself is again defined based on the JavaScript Object Notation (JSON) [59] exchange format. Every message contains two main entries: (i) the actual get or set value and (ii) a Coordinated Universal Time (UTC) timestamp containing the message's creation date. In addition to that, there may be additional metadata depending on the message type.

Besides MQTT, Web applications are used for supervising, evaluating, and maintaining the control logic, as mentioned above. Finally, field buses are used for the direct connection to the appliances in the households. Due to the inhomogeneity of the connected devices, there is a wide range of communication protocols within a household; for example, Zigbee [60], Modbus [61] or Controller Area Network (CAN) bus [62] are commonly used in this context. These field bus protocol stacks are connected to the system via drivers, which translate the respective field bus protocols to an MQTT messaging in order to hide appliance-specific details from the controller. Especially, specific MAC-Layer addressing can thus be avoided.

### C. POWER COMPONENTS AND FLOWS

A single household, as it is used later in simulations (see Section V) and field tests (see Section VI), contains several power components as shown in Figure 7 and listed in the following itemization:

- **PV system:** The PV system feeds power to the household. In reality, this follows the respective weather situations. Especially, daylight duration and cloudiness have an impact on the output. These values are not measured and stored in our system; yet, the outside temperature is gathered with the sensory associated with the heat pump. For the latter, a statistical correlation with the PV output can be proven (daylight length and temperature are both dependent on the season). In the simulation, no curtailment limit is implemented. The feed-in is calculated according to a predefined weather data set (which may be aligned with measured values) or it is derived from a defined PV profile.
- **Battery:** The local battery energy storage systems can consume or produce power, depending on the balance



**FIGURE 7.** In this overview of devices and main power flows within a simulated or real household, the red arrow represents the thermal power invested to heat the building, whereas the gray arrows represent electric power. The component *Electric load* comprises all non-controllable loads of the household; the *Building* stands for a thermal energy consumer here.

of PV production and consumption in the household. It can either be working in autonomous mode (self-consumption optimization), which is for example the case in CUC 0, or be controlled by the household controller. In the latter case, upper and lower limits are used dependent on the current CUC. For the simulation, the battery is modeled as a limited integrator with continuous power dissipation and a capacity of 13 kWh. In the field tests, two different batteries have been utilized: A Siemens Junelight SB-13,2 with a capacity of 13.2 kWh and an inverter power of 3.5 kW, and a Neoom Kjuube Light HV-SO with a capacity of 14.2 kWh and an inverter power of 8 kW. Additionally, in multi-household simulations, cluster storage with a capacity of 90.6 kWh is used.

- *EV charging station:* The EV charging station is of type Keba KeContact P30. It represents a controllable load where the maximum charging current can be set. By its nature, it is only available when a car is connected. The power can be scaled up to 22 kW depending on the fuse rating and the capability of the connected EV. In a Vehicle to Grid (V2G) scenario, negative values for the charging power could be set. In this case, the wallbox control would be similar to battery control, provided that an EV is connected that is capable of providing energy to the grid. In the field tests as described in Section VI, no such car had been used for obvious economic reasons.
- *Heat pump:* The type iDM Terra SW 30S heat pump also represents a controllable load with a nominal electrical power of 10 kW and a strong restriction on maintaining comfort within the building. Again, it can

be operated in an autonomous mode for CUC 0. For the positive CUCs, the restrictions are user-defined. The basic idea is to keep the inside temperature derived by the room sensor of the heat pump within a defined band. In the Net-Price High CUC, a hysteresis from 21 to 22 °C is used for the on/off control. In the Flex Up CUC, this hysteresis is kept smaller (21 to 21.5 °C) to shorten or delay the operation time and thus save electric power for the time being.

- *Household load:* The electric household load consists of all appliances, which are not directly controllable by the household controller/CEMS. However, indirect control may be possible, if humans in the household get appropriate energy feedback to change their power consumption behavior. Yet, this has not been researched in this project context.
- *Building:* The building is representing a thermal energy sink, as can be seen in Figure 7. This is resulting from heating demands in the winter season; cooling in summer has not been researched, however. As the heating is performed with a heat pump in the considered household, this can be used as a controllable consumer. As thermal effects in the building have comparably high inertia, it is well suited for short-term power shifts. In the simulation, a close-to-reality model has been applied (see Section V).
- *Grid:* The grid finally represents the connected LV grid. After balancing production and consumption, the imbalance is tried to compensate with the (local or cluster) battery storage. If this is impossible due to battery limits, the remainder has to be exchanged with the grid. The main goal of the control strategy has been to minimize this exchange.

Heat pumps, batteries, and wallboxes have been chosen due to their controllability and their ability to provide a substantial amount of flexibility, compared with appliances like dishwashers, etc. For the real-world test-bed, existing households and their appliances have been used for economic reasons, but also to demonstrate the system's ability to co-operate with all given appliances which provide a common control interface (in many cases, but not necessarily, this is realized via Modbus). As mentioned, the edge device contains a gateway functionality to translate the individual control interfaces to the controller's MQTT based communication. The effort for this translation varies significantly. Standard compliant control interfaces like Modbus SunSPEC [63], which is common for PV systems, can be integrated much easier as appliances which use proprietary control protocols. However, if comparable appliances use different internal semantics (e.g., one storage device uses just charge/discharge control, while another device also allows controlling the used power), the controller would need to provide different logics to cope with that. To keep the effort reasonable, we used semantically compatible devices, such that the customization is only affected by syntactic issues (names and structures of comparable datapoints).



#### D. CONTROLLER SOFTWARE ARCHITECTURE

The actual control logic (as described in Section III) can be mapped into the system realization using object-oriented paradigms, where an abstract `UseCase` class supports a base control method to apply the control algorithms for all devices. This base method needs to iterate all devices of the household and apply a type-specific control method, that in turn can be separated into three single steps for pre- and post-controlling, next to the actual control process. Pre-processing is reserved for plausibility checks for sensor values, yet not used in the current implementation. Post-processing is used for checking violations of side conditions.

Besides the base algorithm mentioned above, the controller may also call combined control algorithms for multiple devices of different types. Based on this foundation, there may exist multiple implementations for the different CUCs, as shown in the upper part of Figure 8. This creates a “Per Controller Behavior (PCB)”, just like the PHB in routers allowing for different implementations of given QoS requirements.

Next to the actual control algorithm, the controller needs also connectivity components to be able to exchange information with the remaining system. For this purpose, an `MqttManager` class is required that is able to react on MQTT specific events, like getting connected to the broker or losing the connection. In addition to that, it must also be able to subscribe to household relevant topics in order to allow to react on cluster commands, but also to get notified when new sensor values are available and forward those to the `Controller` class.

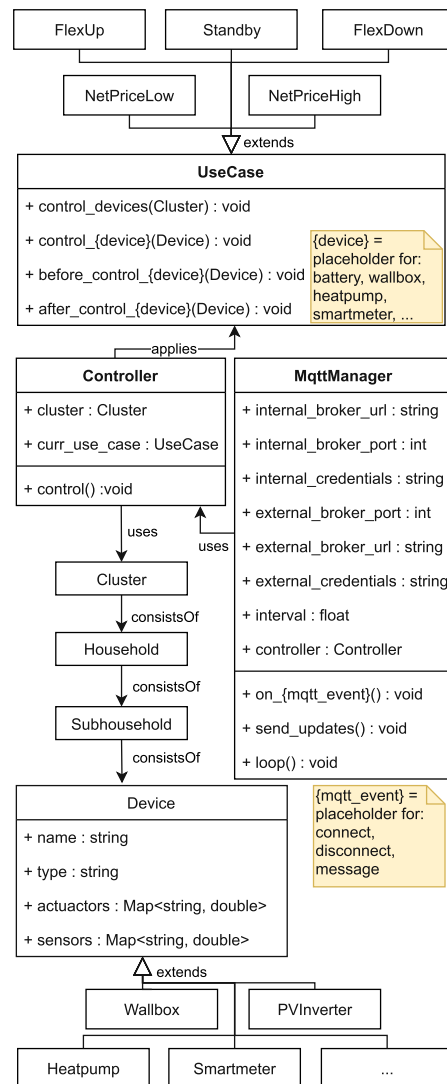
The `MqttManager` must also be able to broadcast actuator messages based on CUC specific optimization strategies, that are applied within the injected `Controller`. This task is performed in a loop with a definable interval, to be able to map inert systems, where a reaction to an executed action is not immediately measurable (e.g., the changing water temperature after adapting the target value of the heat pump). The software structure of the controller, as well as the MQTT component, are shown in the bottom half of Figure 8.

#### V. HOUSEHOLD SIMULATIONS

This section describes the simulation environment for a single household as well as for a group of households with similar equipment. The scenarios, the executions, and the corresponding results of all the simulations in a single-household environment and in a multi-household environment are also given here. The aim of this simulation environment is to test the algorithm, find potential errors in the control outputs, and see the influence of the control mechanism before implementing in the field. The simulated scenarios shall also demonstrate the potential of smart energy management.

##### A. SIMULATION ENVIRONMENT AND SCENARIOS

The simulation environment was created in Matlab/Simulink [10]; a standard PC was used as hardware base



**FIGURE 8.** Software architecture of the household controller shows the use of MQTT to receive sensor values, as well as cluster commands, and to send cluster acknowledgment and actuator messages to control the devices of a household based on the current CUC.

for all simulations. For the appliances, component models have been built specifically for this project. Additionally, the thermal model has been derived from the Carnot toolbox [64]; yet solar gains have been modified according to existing measurements of PV power, as this correlates well with real weather conditions. Technically, an interface between Matlab and Python is used, such that direct communication between the simulation and the controller (see also Section III-C) is established. The cluster command is directly set in the simulation environment, according to the defined test scenarios.

Basically, the simulation thus builds a wrapper around the controller, which is included as an external function within a

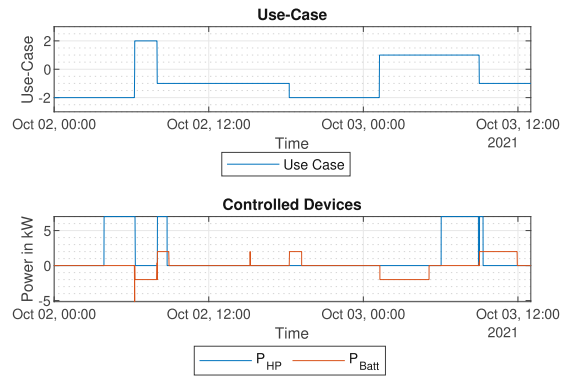
household. Thus, the MQTT broker is not necessary within the simulation; also the gateway function of the edge device (appliance drivers) is not needed for the simulation, as no communication to real hardware appliances is used. The single components in the simulation for both single-household and multi-household environments (appliances, controller, database) are the same as in the real-world tests, as described in Section IV-C.

To execute several simulation runs in this simulation environment, the respective scenarios had to be defined. The scenario definition has been based on the set of input parameters, i.e., the CUC, the ambient temperature, the power consumption of the non-controllable appliances, the cable state of connected EVs, and the instantaneous power of the connected PV system. As simulation output, the power demand of the participating appliances over time is derived. To be able to guarantee as realistic scenarios as possible, we have used stored values in our database, which have been collected by our sensors over more than a year, as inputs. By doing so, we are also able to retrace different context conditions due to seasonal effects or different usage patterns at weekends or holidays, etc.

For these simulations, we have used the autonomous behavior of the considered appliances as a baseline, and explored the effects of our control algorithm (see Section III) in contrast; i.e., for the baseline scenarios, no optimization is performed (also no self-consumption optimization), and each appliance works autonomously as foreseen in the respective given appliance controls. In comparison to the baseline, load shedding or load adding is performed by our control algorithm (as described in Section III), depending on the flexibility requests of the DSO. For single-household simulations, CUC 0 is identical to the baseline, since the DSO has no control requirements on the cluster participant. The effects of our algorithm on the power consumption can then be measured, while all other input parameters remain the same (*ceteris paribus* condition).

### B. SINGLE-HOUSEHOLD SIMULATION

For evaluating the system reaction on our control logic, we simulated leaps from CUC 0 to all the others, while leaving all other input parameters the same. After issuing the non 0 CUCs, the implemented control algorithm overrides the autonomous behavior of the appliances. Thus, we could derive the changes which have been caused by the controller, and compare them with the required flexibilities. An illustrative example of these effects is shown in Figure 9, where the upper curve denotes the CUCs for a given time period, and the lower curves represent the power exchange with battery and heat pump respectively. It can be seen for instance, that in the morning of the first day, CUC +2 leads to an interruption of the heating period, as well as to a provision of energy stored in the battery. However, these effects can not always be observed in that clarity, as convenience conditions override the flexibility requests (see Section III).



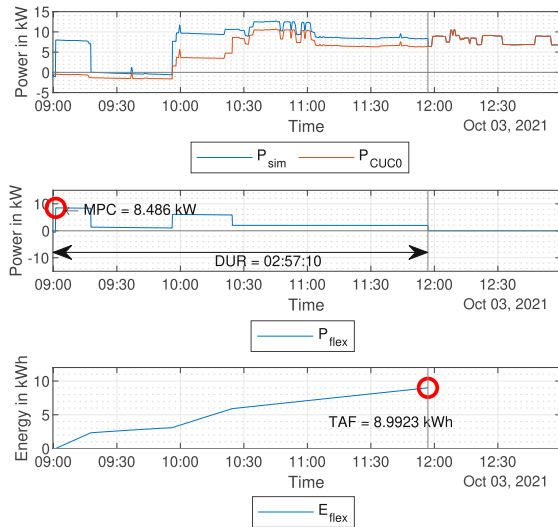
**FIGURE 9.** Simulation example shows the sequence of CUCs, and the reaction of controlled appliances on these flexibility requests, taking into consideration user-defined convenience limitations.

In order to quantitatively assess the provided flexibility, appropriate metrics had to be defined. In this work, we evaluated the Maximum Power Change (MPC) triggered by a changing CUC, the Duration of Change (DUR) (i.e., the time between the trigger and the return of the flex curve to the value of the baseline curve), and the Total Amount of Provided Flexibility (TAF) (i.e., the area between the baseline curve and the flex curve, calculated by the integral of the difference). With these metrics, an assessment of the provided flexibility can be performed, as exemplarily depicted in Figure 10. The upper curves show the power balance in the simulations of CUC 0 (the baseline) and CUC -1; the middle curve depicts their difference (which is the reaction of the system on the different CUC), and the lowest curve the integral of the middle curve (for calculating the TAF metric). Again, the example demonstrates the effects of the control under beneficial conditions.

### C. MULTI-HOUSEHOLD SIMULATION

For the multi-household simulations, the environment for single-household simulations has been cloned several times, such that at the end, six households (with one sub-household each) could be used for the simulations [65]. Each of the clones then has been modified a little regarding the appliances present in the respective households. Additionally, a Community Energy Storage (CES) has been introduced, which is connected to each of the participating households. Each household is able to charge the CES, if the upper limit has not yet been reached, and discharge it, as long as the lower limit has not been reached. Both activities can be performed with the same CES tariff, which is an intermediate tariff between purchase and feed-in tariff when exchanging power with the DSO.

By doing so, the CES may be used by participating households as cost neutral extension of the internal storage; moreover, it can be used by different households to transfer energy from a household with an energy surplus to an other household with power demand. In comparison with a feed-in into



**FIGURE 10.** The metrics on flexibility requests are assessed in an illustrative example, showing the baseline (the simulation of CUC 0), the simulated reaction on issuing CUC -1, as well as the resulting values for the metrics.

the associated LV grid, or an energy purchase from the DSO, both the energy-providing party and the energy-consuming party profit from such an exchange. For the simulations, 0.21 € per kWh has been used as purchase tariff, 0.10 € as feed-in tariff, and 0.15 € as internal tariff. The CES allows for balancing the several households of the community before they exchange energy with the DSO; thus, it helps to relieve the power lines (in opposite to a just financial balancing of energy communities). However, it also provides additional storage capacity and thus strengthens the flexibility potential of the community.

Table 1 shows the electrical and financial effects of the CES on the community according to the simulated test households, for a simulated time of 30 days. The CUC has to be set to 0 throughout the whole multi-household simulation (since in the other CUCs, flexibility provision will override the autonomous behavior of participating appliances). The other input parameters known from single-household simulation are set to realistic values (which have either been directly derived from real-world measurements in a similar time period, or calculated by using the mentioned thermal model). Here, the purchase or selling is done either to the CES (if present), or directly to the DSO.

In the simulation, the limits of the CES have never been reached, such that no mixed purchases or sales have occurred. As it can be seen, the amount of purchased and sold energy for all households remains the same, i.e., the CES does not influence the appliance behavior. In the case of direct exchange with the DSO, the cost benefits are negative, i.e., the households have to pay the DSO, although the balance of energy is positive, which results from the different tariffs for purchasing and selling power. However, from a purely financial point

**TABLE 1.** Energy and cost effects of the usage of common cluster storage for an energy community consisting of six individual households, partly equipped with individual storages, heat pumps, EVs, and PV systems for a simulation of 30 days reveal some additional benefits [65].

	Without CES	With CES
Energy sold	94.86 kWh	94.86 kWh
Energy purchased	89.46 kWh	89.46 kWh
Revenues	9.486 €	14.229 €
Expenditures	18.786 €	13.419 €
Cost benefits	-9.30 €	0.81 €

of view, the effects of the CES from about 10 € per month for the whole energy community would not justify the investment of a CES. Yet, the additional flexibility, as well as a smoothing effect helping the connected LV grid for peak clipping and valley filling, might raise the interest of DSOs to provide such a device as an additional incentive.

## VI. FIELD TESTS

For the tests in a real-world environment, there was no obvious baseline since environmental as well as usage patterns are never exactly identical, and we could either apply our control algorithm or not. To be able to assess the effect of our control, in reality, we thus used two reference scenarios for comparison:

- 1) The “wallet reference” is based on the assumption that usage and environmental influences stay constant for the duration of a use case. Consumption changes in the requested direction are thus a result of the applied control strategy and deserve a reward in form of financial incentives (i.e., the wallet, see Section III).
- 2) The “simulation reference” is based on the simulation scenarios, which are again derived from the collected historical data persisted in our database. If the simulation represents reality well, it can be used as a reference, that approximates reality without using our controller entity.

### A. TEST SCENARIOS

The simulation reference is the same as used in the evaluation of the simulations. As a baseline for both, the simulation of CUC 0 is taken; yet, the evaluated curve here represents the real data in other CUCs and not the simulated data in other CUCs. The simulation reference is used only for evaluation purposes for the work at hand and has not been integrated into the calculation of incentives. For the following, we concentrated on the simulation reference, as it delivers more accurate results than the wallet reference when the simulation works well (see Section VII and Figure 10). However, the definition for  $E_{flex}$  from Equation (1) has to be adopted in this case, as flexibility can be measured now continuously, not only for 15 min timeslots, as follows:

$$E_{flex}(t) = \int_a^t P_{flex} \quad (2)$$

Here,  $P_{flex}$  is the difference between the flex curve and the baseline curve,  $t$  is the instantaneous time of the measurement, and  $a$  is the time of the flexibility request as described in Section V-B and visualized in Figure 10. When  $b$  is the time where the flex curve returns to the baseline curve, it can easily be seen that the TAF metric equals  $E_{flex}(b)$ .

As a next step, the scenarios for the field tests have been defined for some appliances present at the used testbed in Stegersbach. We have concentrated on controllable energy-consuming appliances here. The non-controllable consumers can not be influenced by our algorithmic logic, such that only sensor readings could be tested. The energy-producing roof-top PV could yet be influenced, but should not be down-regulated for keeping purchased energy as small as possible unless stability reasons enforce a regulation. By doing so, we have defined scenarios for a heat pump, a wallbox, and the electric energy storage(s).

For each of these devices, we have defined 3 kinds of functional test scenarios:

- *Tests of base functionality:* Under given appliance-specific conditions, each CUC has been applied to the controller component, and the reactions of this component (in form of MQTT messages) have been measured and documented. Furthermore, the application of control commands to the appliances under test (MQTT actuator commands translated to the respective appliance interfaces) has been conducted, and again the reaction of the appliances has been observed and documented. Thus, it can also be verified that the actions have real-life effects as intended (which can not be validated in a simulation environment only).
- *Tests of stochastic CUC sequences:* These tests are designed as long-term tests in order to not only cover all CUCs, but also all possible CUC leaps (especially from CUC 0 to the others), and this under different environmental conditions. By doing so, white-box testing should be avoided, which is hard to define properly, as the environmental conditions are not predictable and can hardly be grouped into meaningful categories. These tests should reveal potential shortcomings of the applied logic which may occur very rarely in practical environments, but can nevertheless be theoretically possible. By examining these rare situations, we can prove that the applied control algorithms are robust against extreme conditions.
- *Tests of realistic CUC sequences:* In order to conduct tests in near realistic conditions, a sequence of cluster CUCs is derived from typical power price curves at the Austrian spot market. These price curves represent shortages and surpluses of electric power in transmission and distribution grids and can thus incorporate different technical conditions in the LV grid associated with the field test environment. A price curve of a typical weekday has been selected as a representative.

The delimitation of CUCs has been done on the base of manually chosen limits at 49.9, 56.1, 61.9, and 72.7 €/MWh in order to support “higher” CUCs (which trigger energy savings) while keeping a reasonable distribution of all defined use cases. Prices and CUCs have been defined on an hourly basis here.

### B. CONTROLLER VALIDATION

For the base functionality tests, the expected results of the control (see Section III) could be obtained after eliminating initial implementation difficulties. Hereby, scaling issues played the most important role; by limiting the number of messages (per controller cycle, only one value per data-point has been transmitted), the controller hardware was then able to proceed as foreseen. This means, that the heat pump showed an on/off behavior according to the room temperature limits of the respective CUC; the storage kept the respective SOC limits, and the wallbox deferred charging as long as possible when required by the CUC.

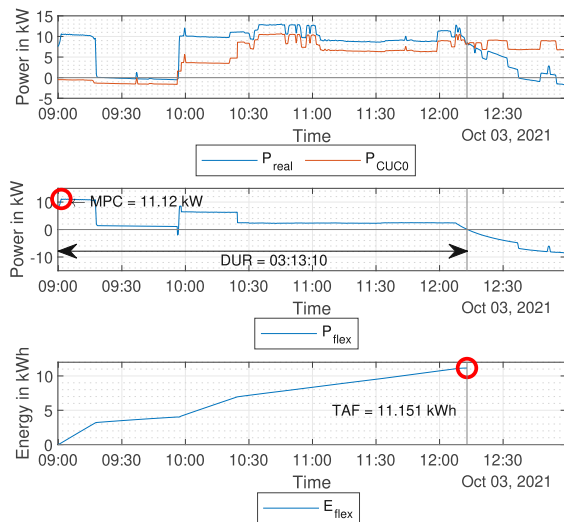
For the heat pump, functionality tests have been conducted in different environmental situations, i.e., in winter tests with cold ambient temperatures have been conducted and in spring with medium ambient temperatures. For the wallbox, functionality tests have been conducted in different usage situations, i.e., with or without a car plugged; with a car plugged and additional user requirements set (immediate charging or charging shall be finished within 12 hours), as well as plugging or unplugging within a charging session. For the stochastic CUC sequences, the tests showed similar effects as in the simulation (mentioned in Section V and visualized in Figure 9).

After validation of the controller capabilities, some close-to-reality test sequences have been conducted. The daily sequence of CUCs is defined as mentioned above. Additionally, different environmental and/or usage patterns are issued again. Here, heat pump tests include different weather conditions (warm&sunny, warm&cloudy, cold&sunny, and cold&cloudy). The second parameter is as well important here, as the installed PV could be used for operating the heat pump in sunny weather conditions, serving as an electric power sink, if needed.

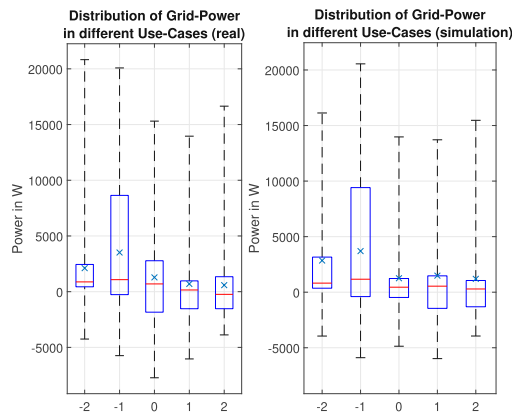
For the wallbox tests, besides the usual tests with given CUCs and different usage patterns, also an additional scenario with a higher power supply during the day (due to PVs production) is considered. Finally, for the storage, the CUC sequences are tested with different SOC values ( $SOC < 20\%$ ,  $20\% \leq SOC < 80\%$ ,  $80\% \leq SOC$ ). Figure 11 shows a typical example of the conducted field tests with respective values of the evaluation metrics. Hereby, the parameters are the same as in Figure 10, except for the blue line in the upper part, which is the real power here.

### VII. EVALUATION AND DISCUSSION

When analyzing simulations and field tests, the system’s behavior is in focus. As the controller’s main purpose is to provide flexibility to the DSO, the ability to shed or add loads



**FIGURE 11.** Typical example of the conducted field tests reveals the evaluation metrics in a close-to-reality scenario (the CUCs are based on typical price curves and not real-time flexibility requests; all other parameters are resulting from real-world measurements and given user preferences).



**FIGURE 12.** Boxplot shows the mean value, the median, the quartiles, as well as the range of the power values which are fed into the LV grid, depending on the CUCs set by the grid operator. The left part shows the reaction of the system in real field tests, whereas the right part depicts the respective simulated values.

on demand has to be proven. Concrete, the power consumption should have a clear dependence on the CUC, which will be shown in the following. Furthermore, a validation of the simulation model has to be done. As the simulation is used as a reference in Section VI, the validity of this reference has to be shown. For that purpose, a comparison of measured values with simulated values, which are derived by using the same input parameters, has to be performed.

#### A. INFLUENCE OF CLUSTER USE CASES

Figure 12 shows the distribution of grid power values between the CUCs in the simulation and in the field test. Here, the red

line represents the median of all power values in a certain CUC. The blue box represents the 25th and the 75th percentile, and therefore 50% of all power values in the respective use case are within those borders. The mean power value is depicted with a cross. Considering the CUC definitions, high consumption should take place in the negative CUCs, whereas the power values should be lower in the positive CUCs.

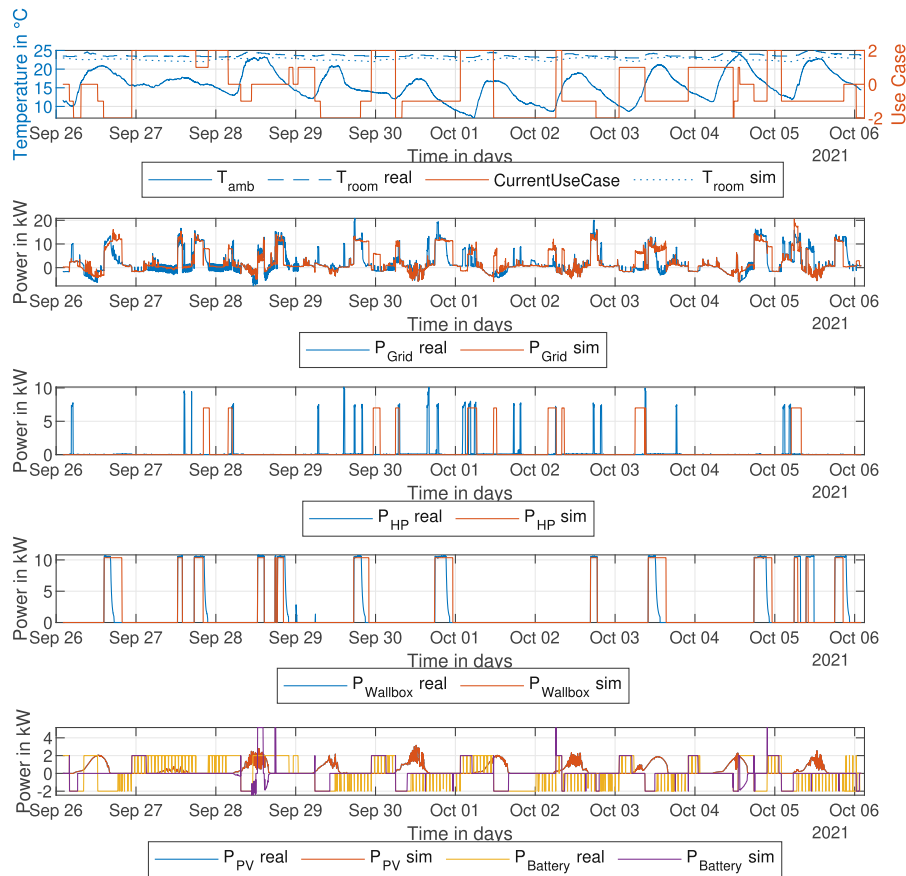
In fact, the CUCs  $-2$  and  $-1$  show higher power values than  $+1$  or  $+2$ , as intended. However, the “flex” CUCs, especially CUC  $-2$ , appear to have less influence. Here, we are evaluating time series with realistic CUCs sequences (see Section VI) in order to derive meaningful results; yet, this means that leaps from CUC 0 to the “flex” CUCs are very rare. As we only adapt the limits (battery SOC, etc.) when changing from a “net price” CUC to a “flex” CUC (e.g., if CUC  $+2$  comes after  $+1$ ), the remaining flexibility potential might be too low to still receive a remarkable effect. Thus, flexibility effects might have already been exploited.

However, when considering the power provided to the DSO, the CUCs  $-1$  to  $+1$  work as expected. For a roll-out in real world, either the used limits should be adapted (in order to keep a higher reserve), or the strategies should be changed (e.g., by providing appliances that are only touched in the “flex” CUCs). Simulation and field test measurements show similar behavior here, indicating a well-modeled system; however, this has to be explored in more detail.

#### B. VALIDATION OF SIMULATION RESULTS

To provide the required validation, we have chosen a time series showing the most important energy-related values for 10 days in spring, as it is depicted in Figure 13. As the ambient temperature is highly fluctuating these days, leading to an on/off behavior of the heat pump, the period seems adequate for the analysis. As can be seen, the simulation results and the measurements show a good match. The upper line shows temperature values and the CUC. The second line represents the simulated and real grid power, where the simulation only deviates slightly from the measurement, indicating a well-modeled battery energy storage system and wallbox. Lines 3 and 4 present the respective power data for the appliances heat pump and wallbox, and line 5 again for the battery and the PV system.

Modeling the heat pump is more complex, as it requires a precise model of the building and the heating system. Both are strongly influenced by environmental variables (temperature, humidity, wind, solar radiation, etc.) and user behavior (presence, ventilation, hot water demand, etc.). The standby consumption of the heat pump ranging from 20 to 100 W in the measurements was not considered in the simulation model. The simulation model of the wallbox is even more dependent on user behavior (plugged or unplugged car status, driven distance, etc.). The model makes use of the logged, real plug state of the EV in order to allow a fair comparison between simulation and reality. It was assumed, that the longer the



**FIGURE 13.** Timeseries of the heat pump, the wallbox, the PV, and the electric storage device are compared with the power fed into the connected grid, both for the simulation and the real field test in the given period of time. The first row shows input parameters as the CUC and the ambient temperature, as well as the measured and simulated output parameter room temperature.

car was not connected, the lower the SOC would be. With these inputs and assumptions, a good correlation between simulation and the wallbox consumption profiles could be reached.

As can be seen in the lowest graph, there was a communication problem with the battery leading to data quality issues; i.e., leaps in the measured battery power occurred. Alignment between the simulated and measured power can still be observed. The PV production measurement was directly used in the simulation as an environmental variable since it was not being influenced by the controller. Altogether it can be stated, that the simulation model gives a good estimate of the grid power. The thermal modeling of the building and heating system in order to simulate the heat pump behavior and electricity consumption is difficult. From a statistical point of view, the heat pump behavior shows many similarities (especially, average consumption over time); yet, the observed time shifts are stochastic.

### C. COMPARISON AND DISCUSSION

After having validated the approach, an assessment of the afore-defined research questions can be given. A short summary is depicted in Table 2. The contribution of one household to one flexibility request (CUC change) seems promising; however, questions of scaling (regarding the number of households, or the number of requests) should be explored in a bigger real-world testbed. Thus, the effects of aggregated flexibility provision could be as well researched as architectural effects. However, by using distributed controller instances on the respective edge devices for the critical functionality, no devastating scalability effects in the ICT infrastructure can be expected. The community effects are negligible; additional stability effects could only result from the exceeded storage capacity by introducing cluster storage; however, the additional storage is just constituting one additional household, where the provided flexibility amount is again dependent on the storage capacity.

**TABLE 2. Qualitative and quantitative assessments of the defined research questions listed in Section I demonstrate the validity of the chosen approach.**

Issue	Assessment
RQ1	The defined metrics MPC, DUR, and TAF allow for a quantitative assessment of single flexibility requests. Under auspicious circumstances, a volume of more than 11 kWh (as visualized in Figure 11) can be shifted by one common household for a single request.
RQ2	The simulation reference (defined in Section VI) has turned out to form a useful baseline, which could be compared well with real-life measurements, as depicted in Section VII-B. As the most important parameter, the total electric load in the simulation usually differs by under 10% from the measurements, e.g. by 8.6% for a 20 days period in 2021.
RQ3	As stated in Table 1, the financial effect of intra-community trading for customers was only about 10 € after a 30 days simulation period for CUC 0. For the other CUCs, the appliances are controlled directly via the given CUC strategy (i.e., no trading between households occurs). However, the cluster storage provides an additional buffer for shifting power consumption to more suitable times.
RQ4	A controller using a loop time of 1 min provided a sufficient amount of flexibility on demand of a cluster operator. Public Internet as underlay network and utilizing MQTT over TCP/IP provided sufficient network performance, as well did standard PCs and embedded PCs (as depicted in Section IV-C) as endpoints.
RQ5	The use of MQTT allowed for loosely coupled systems with yet sufficient performance. The customization effort can thus be kept acceptable, as the control logic is set up on a generic topic system (however, translations to this topic system are still necessary, as pointed out in Section IV-A). State information is included in this topic system.

When comparing this work with existing literature, it can be said that there are more advanced strategies for controller algorithms than our simple rule-based approach; also, there are testbeds with many more participants than we used, and protocol suites may be better suitable for real-time requirements than the one we have used here. However, only very rare contributions combine these two approaches, resulting either in purely theoretical assessments of algorithmic advances while not having proven real-world capability, or in ICT infrastructures which have not been used with real grid-related control logic. To the best knowledge of the authors, those rare contributions which are trying to combine these approaches, mainly concentrate on issues like self-consumption optimization (e.g., [66]), taking into account the better simultaneity factors in energy communities. On the other hand, flexibility issues are explored in some new research papers, yet concentrating on user convenience aspects [67] or on market participation [68] rather than effects on the associated LV grid. However, the assessment of flexibility potentials is important for grid stability, such that a realistic assessment using a combined approach as in this work may impose a noteworthy contribution to grid management.

Another point that has not been sufficiently solved in existing literature, is the definition of a realistic baseline to compare the effects of the control with the initial situation. In literature, usually mathematical assessments are made

instead of field tests (e.g., [69]); however, a practical assessment further increases the degree of realism and validity. Due to the volatility of energy consumption in households, measuring the difference to past situations is not satisfactory. To overcome this, we have made simulations additional to the field tests and validated these comparatively by using the same set of input parameters. We used these simulations to represent the situation without using our control with more degree of realism (the “simulation reference” mentioned in Section VI).

Finally, the uniqueness of this contribution has been rounded by providing a rigorous metrical assessment of flexibility, by defining the two dimensions of flexibility (represented by DUR and MPC), as well as the total amount (represented by TAF). Related approaches have been listed by [70]; however, the mentioned flexibility metrics are bound to concrete appliances there (mostly in combination with thermal storages), and utilized to assess appliances’ capacities rather than measuring realized flexibilities. All these unique contributions may help operators with their stability goals, but they also support the scientific grounding of practical work in the field.

## VIII. CONCLUSION

With the work at hand, it could be shown that energy communities have the potential to yield a remarkable impact on the flexibility market, even when applying very simple rule-based control strategies. Together with self-consumption optimization, which is much better explored in current research, such systems may be used in the future to gain financial benefits for participants of energy communities, while at the same time contributing to more stable and better balanced LV grids. The effectivity of such systems may be further enhanced by utilizing better optimizers, especially incorporating ML algorithms and predictions of relevant input parameters like weather conditions, etc.

Further research is also necessary to explore the practical differences between grid instabilities due to a surplus of power and those which are originating in a lack of electrical power. Especially for appliances that are starting their operation immediately when certain preconditions are fulfilled (e.g., EV charging systems usually start charging the car right after plugging it), additional logic (e.g., default idle times) may be required.

Also, it could be shown that a message-based infrastructure (in this case employing MQTT message queues) is capable of delivering control relevant data promptly and in sufficient quality to allow for control mechanisms that fulfill the requirements of LV grid control. Hereby, standard security mechanisms like encryption and authentication/authorization have been utilized. However, security will stay an important research topic in such an environment, as grids are critical infrastructures, and as much personal data are processed.

In future realizations, MQTT communication may be separated into two steps using an internal and an external MQTT broker. To be able to handle this, the `MqttManager` is

already capable of managing multiple connections in form of a Unified Resource Locator (URL) and a port, as well as authentication credentials where required. Thus, the system is designed in a scalable manner allowing for a broader adaptation of the researched technologies in the field.

#### ACRONYMS

<b>CAN</b>	Controller Area Network
<b>CEMS</b>	Customer Energy Management System
<b>CES</b>	Community Energy Storage
<b>CORBA</b>	Common Object Request Broker Architecture
<b>COSEM</b>	Companion Specification for Energy Metering
<b>CUC</b>	Cluster Use Case
<b>DER</b>	Distributed Energy Resource
<b>DLMS</b>	Device Language Message Specification
<b>DSCP</b>	Differentiated Services Codepoint
<b>DSM</b>	Demand Side Management
<b>DSO</b>	Distribution System Operator
<b>DUR</b>	Duration of Change
<b>EER</b>	Enhanced Entity Relationship
<b>EV</b>	Electric Vehicle
<b>GB</b>	Gigabyte
<b>GWAC</b>	GridWise Architecture Council
<b>ICT</b>	Information and Communication Technology
<b>IEC</b>	International Electrotechnical Commission
<b>IED</b>	Intelligent Energy Device
<b>IOT</b>	Internet of Things
<b>IP</b>	Internet Protocol
<b>ISO</b>	International Organization for Standardization
<b>JSON</b>	JavaScript Object Notation
<b>LAN</b>	Local Area Network
<b>LV</b>	Low Voltage
<b>MAC</b>	Medium Access Control
<b>ML</b>	Machine Learning
<b>MPC</b>	Maximum Power Change
<b>MPLS</b>	Multi-Protocol Label Switching
<b>MQTT</b>	Message Queuing Telemetry Transport
<b>OPC UA</b>	Open Process Control Unified Architecture
<b>OSI</b>	Open Systems Interconnection
<b>P4</b>	Programming Protocol-independent Packet Processors
<b>PC</b>	Personal Computer
<b>PCB</b>	Per Controller Behavior
<b>PHB</b>	Per Hop Behavior
<b>PLC</b>	Power Line Communication
<b>PoC</b>	Proof of Concept
<b>PV</b>	Photovoltaic System
<b>RAM</b>	Random Access Memory
<b>RQ</b>	Research Question
<b>QoS</b>	Quality of Service
<b>SDN</b>	Software Defined Networking
<b>SGAM</b>	Smart Grid Architecture Model
<b>SOC</b>	State of Charge
<b>SSL</b>	Secure Socket Layer
<b>TAF</b>	Total Amount of Provided Flexibility
<b>TCP</b>	Transmission Control Protocol

<b>TE</b>	Traffic Engineering
<b>TSO</b>	Transmission System Operator
<b>UDP</b>	User Datagram Protocol
<b>URL</b>	Unified Resource Locator
<b>UTC</b>	Coordinated Universal Time
<b>V2G</b>	Vehicle to Grid
<b>VLAN</b>	Virtual LAN
<b>VXLAN</b>	Virtual eXtensible LAN
<b>WAN</b>	Wide Area Network

#### REFERENCES

- [1] T. Strasser, "A review of architectures and concepts for intelligence in future electric energy systems," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2424–2438, Apr. 2015.
- [2] A. Kiani and A. Annaswamy, "A hierarchical transactive control architecture for renewables integration in smart grids," in *Proc. IEEE 51st IEEE Conf. Decis. Control (CDC)*, Dec. 2012, pp. 4985–4990.
- [3] W. Yu, D. Liu, and Y. Huang, "Operation optimization based on the power supply and storage capacity of an active distribution network," *Energies*, vol. 6, no. 12, pp. 6423–6438, Dec. 2013.
- [4] M. Pichler, A. Veichtlbauer, and D. Engel, "Evaluation of OSGI-based architectures for customer energy management systems," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Mar. 2015, pp. 2455–2460.
- [5] *Regulatory Recommendations for the Deployment of Flexibility*, Smart Grid Taks Force (EG3), Brussels, Belgium, Jan. 2015.
- [6] K. Heussen, S. You, B. Biegel, L. H. Hansen, and K. B. Andersen, "Indirect control for demand side management—A conceptual introduction," in *Proc. 3rd IEEE PES Innov. Smart Grid Technol. Eur. (ISGT Eur.)*, Oct. 2012, pp. 1–8.
- [7] A. Caramizaru and A. Uihlein, *Energy Communities: An Overview of Energy and Social Innovation*. Luxembourg, Luxembourg: Publications Office of the European Union, 2020.
- [8] U. J. J. Hahnel, M. Herberz, A. Pena-Bello, D. Parra, and T. Brosch, "Becoming prosumer: Revealing trading preferences and decision-making strategies in peer-to-peer energy communities," *Energy Policy*, vol. 137, Feb. 2020, Art. no. 111098.
- [9] J. Gugeneder, C. Muck, G. Steinmaurer, and A. Veichtlbauer, "Simulative analyse der potenziellen energieflexibilitäten von einzelhaushalten langfassung," in *Proc. Tagungsband des 16th Symp. Energieinnovation*, Graz, Austria, Feb. 2020, pp. 363–364.
- [10] The MathWorks, Inc. (2013). *MATLAB—The Language Of Technical Computing*. Accessed: Apr. 24, 2013. [Online]. Available: <http://www.mathworks.com/products/MATLAB/>
- [11] N. Iqtiyaniilham, M. Hasanuzzaman, and M. Hosenuzzaman, "European smart grid prospects, policies, and challenges," *Renew. Sustain. Energy Rev.*, vol. 67, pp. 776–790, Jan. 2017.
- [12] F. Andren, R. Brundlinger, and T. Strasser, "IEC 61850/61499 control of distributed energy resources: Concept, guidelines, and implementation," *IEEE Trans. Energy Convers.*, vol. 29, no. 4, pp. 1008–1017, Dec. 2014.
- [13] B. P. Koirala, E. Koliou, J. Friege, R. A. Hakvoort, and P. M. Herder, "Energetic communities for community energy: A review of key issues and trends shaping integrated community energy systems," *Renew. Sustain. Energy Rev.*, vol. 56, pp. 722–744, Apr. 2016.
- [14] R. Niemi, J. Mikkola, and P. D. Lund, "Urban energy systems with smart multi-carrier energy networks and renewable energy generation," *Renew. Energy*, vol. 48, pp. 524–536, Dec. 2012.
- [15] P. Palensky and D. Dietrich, "Demand side management: Demand response, intelligent energy systems, and smart loads," *IEEE Trans. Ind. Informat.*, vol. 7, no. 3, pp. 381–388, Aug. 2011.
- [16] S. Bessler, D. Drenjanac, E. Hasenleithner, S. Ahmed-Khan, and N. Silva, "Using flexibility information for energy demand optimization in the low voltage grid," in *Proc. 4th Int. Conf. Smart Cities Green ICT Syst.*, May 2015, pp. 1–9.
- [17] D. T. Nguyen and L. B. Le, "Joint optimization of electric vehicle and home energy scheduling considering user comfort preference," *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 188–199, Jan. 2014.
- [18] A. Barbato, A. Capone, G. Carello, M. Delfanti, M. Merlo, and A. Zaminga, "House energy demand optimization in single and multi-user scenarios," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGrid-Comm)*, Oct. 2011, pp. 345–350.



- [19] G. Seyfang, J. J. Park, and A. Smith, "A thousand flowers blooming? An examination of community energy in the U.K.," *Energy Policy*, vol. 61, pp. 977–989, Oct. 2013.
- [20] G. Seyfang, S. Hielscher, T. Hargreaves, M. Martiskainen, and A. Smith, "A grassroots sustainable energy niche? Reflections on community energy in the U.K.," *Environ. Innov. Societal Transitions*, vol. 13, pp. 21–44, Dec. 2014.
- [21] D. S. Wiyono, S. Stein, and E. H. Gerding, "Novel energy exchange models and a trading agent for community energy market," in *Proc. 13th Int. Conf. Eur. Energy Market (EEM)*, Jun. 2016, pp. 1–5.
- [22] H. R. Gholinejad, A. Loni, J. Adabi, and M. Marzband, "A hierarchical energy management system for multiple home energy hubs in neighborhood grids," *J. Building Eng.*, vol. 28, Mar. 2020, Art. no. 101028.
- [23] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander, and M. S. H. Sunny, "Application of big data and machine learning in smart grid, and associated security concerns: A review," *IEEE Access*, vol. 7, pp. 13960–13988, 2019.
- [24] E. Mocanu, P. H. Nguyen, W. L. Kling, and M. Gibescu, "Unsupervised energy prediction in a smart grid context using reinforcement cross-building transfer learning," *Energy Buildings*, vol. 116, pp. 646–655, Mar. 2016.
- [25] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "Smart grid technologies: Communication technologies and standards," *IEEE Trans. Ind. Informat.*, vol. 7, no. 4, pp. 529–539, Nov. 2011.
- [26] IEC Central Office. (2019). *The Smart Grid Standards Map*. [Online]. Available: <http://smartgridstandardsmap.com/#tabs-2>
- [27] *Gridwise Interoperability Contextsetting Framework*, GridWise Architecture Council, Washington, DC, USA, 2008.
- [28] *Information technology—Open Systems Interconnection Basic Reference Model: The Basic Model*, Standard ISO/IEC 7498-1:1994 International Organization for Standardization, Nov. 1994.
- [29] *Framework Document*, CENCENELEC, ETSI Smart Grid Coordination Group, Sophia Antipolis, France, Nov. 2012.
- [30] D. K. Panda and S. Das, "Smart grid architecture model for control, optimization and data analytics of future power networks with more renewable energy," *J. Cleaner Prod.*, vol. 301, Jun. 2021, Art. no. 126877.
- [31] *Common Object Request Broker Architecture (CORBA) Specification, Version 3.3*, Object Management Group, Boston, MA, USA, Nov. 2015.
- [32] OPC Foundation. (2012). *OPC—The Interoperability Standard for Industrial Automation & Other*. [Online]. Available: <http://www.opcfoundation.org>
- [33] *Welcome to openHAB*, openHAB Community and the openHAB Foundation e.V., Wiesbaden, Hessen, 2019.
- [34] V. K. Sood, D. Fischer, J. M. Eklund, and T. Brown, "Developing a communication infrastructure for the smart grid," in *Proc. IEEE Electr. Power Energy Conf. (EPEC)*, Oct. 2009, pp. 1–7.
- [35] A. Veichtlbauer, O. Langthaler, F. P. Andr n, C. Kasberger, and T. I. Strasser, "Open information architecture for seamless integration of renewable energy sources," *Electronics*, vol. 10, no. 4, p. 496, Feb. 2021. [Online]. Available: <https://www.mdpi.com/2079-9292/10/4/496>
- [36] *Telecontrol Equipment and Systems—Part 1: General Considerations*, document IEC 60870-1, International Electrotechnical Commission, 1988.
- [37] *Smart Grid Reference Architecture*, CEN/Cenelec/ETSI, Smart Grid Coordination Group, Brussels, Belgium, Nov. 2012.
- [38] *Communication Networks and Systems for Power Utility Automation—Part 1: Introduction and Overview*, document IEC 61850-1, International Electrotechnical Commission, 2013.
- [39] J. Postel, *Transmission Control Protocol DARPA Internet Program Protocol Specification*, document RFC 793, IETF, 1981.
- [40] J. Postel, *User Datagram Protocol*, document RFC 768, IETF, Aug. 1980.
- [41] J. Postel, *Internet Protocol DARPA Internet Program Protocol Specification*, document RFC 791, 1981.
- [42] *Electricity Metering Data Exchange—The DLMS/COSEM Suite*, document IEC 62056, International Electrotechnical Commission, Geneva, Switzerland, 2014.
- [43] Z. Zhang, X. Huang, B. Keune, Y. Cao, and Y. Li, "Modeling and simulation of data flow for VLAN-based communication in substations," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2467–2478, Dec. 2017.
- [44] M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks Over Layer 3 Networks*, document RFC7348, Internet Engineering Task Force (IETF), 2014.
- [45] M. A. Ridwan, N. A. M. Radzi, W. S. H. M. Wan Ahmad, F. Abdullah, M. Z. Jamaludin, and M. N. Zakaria, "Recent trends in MPLS networks: Technologies, applications and challenges," *IET Commun.*, vol. 14, no. 2, pp. 177–185, Jan. 2020.
- [46] A. Aydeger, "Software defined networking for smart grid communications," M.S. thesis, College Eng. Comput., Florida Int. Univ., Miami, FL, USA, Jul. 2016.
- [47] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–95, Jul. 2014.
- [48] H. Zhang, B. Liu, and H. Wu, "Smart grid cyber-physical attack and defense: A review," *IEEE Access*, vol. 9, pp. 29641–29659, 2021.
- [49] S.-X. Wang, H.-W. Chen, Q.-Y. Zhao, L.-Y. Guo, X.-Y. Deng, W.-G. Si, and Z.-Q. Sun, "Preserving scheme for user's confidential information in smart grid based on digital watermark and asymmetric encryption," *J. Central South Univ.*, vol. 29, no. 2, pp. 726–740, Feb. 2022.
- [50] A. Veichtlbauer and T. Pfeiffenberger, "Generic middleware for user-friendly control systems in home and building automation," *Int. J. Adv. Netw. Services*, vol. 6, nos. 1–2, pp. 51–67, Jul. 2013.
- [51] The OSGi Alliance. (2012). *OSGi Core Release 5*. [Online]. Available: <http://www.osgi.org/download/r5/osgi.core-5.0.0.pdf>
- [52] F. Kupzog, A. Veichtlbauer, A. Heinisch, F. von T llenburg, O. Langthaler, U. Pache, O. Jung, R. Frank, and P. Dorfinger, "The impact of virtualisation techniques on power system control networks," *Electron.*, vol. 9, no. 9, pp. 1433–1454, Sep. 2020.
- [53] T. J. Teorey, D. Yang, and J. P. Fry, "A logical design methodology for relational databases using the extended entity-relationship model," *ACM Comput. Surveys*, vol. 18, no. 2, pp. 197–222, Jun. 1986.
- [54] J. Turnbull, *Monitoring With Prometheus*. London, U.K.: Turnbull Press, 2018.
- [55] S. N. Z. Naqvi, S. Yfantidou, and E. Zim ny, *Time Series Databases and InfluxDB*. Brussels, Belgium: Universit  Libre de Bruxelles, 2017, p. 12.
- [56] M. Chakraborty and A. P. Kundan, "Grafana," in *Monitoring Cloud-Native Applications*. Cham, Switzerland: Springer, 2021, pp. 187–240.
- [57] I. Miell and A. Sayers, *Docker in Practice*. New York, NY, USA: Simon and Schuster, 2019.
- [58] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," in *Proc. IEEE Int. Syst. Eng. Symp. (ISSE)*, Oct. 2017, pp. 1–7.
- [59] T. Bray, "The Javascript object notation (JSON) data interchange format," IETF, Fremont, CA, USA, Tech. Rep. RFC 7159, 2014.
- [60] S. Safaric and K. Malaric, "ZigBee wireless standard," in *Proc. ELMAR*, Jun. 2006, pp. 259–262.
- [61] A. Swales, "Open Modbus/TCP specification," *Schneider Electr.*, vol. 29, pp. 3–19, Mar. 1999.
- [62] M. Farsi, K. Ratcliff, and M. Barbosa, "An overview of controller area network," *Comput. Control Eng. J.*, vol. 10, no. 3, pp. 113–120, Jun. 1999.
- [63] *Sunspec Device Information Model Specification*, SunSpec Alliance, San Jose, CA, USA, 2019.
- [64] C. Wemh ner, B. Hafner, and K. Schwarzer, "Simulation of solar thermal systems with CARNOT blockset in the environment MATLAB Simulink," in *Proc. Eurosun*, 2000, pp. 1–6.
- [65] V. K. Velmurugan, "Multi-homed energy storage control," M.S. thesis, Dept. Inform., Commun., Media, Univ. Appl. Sci. Hagenberg, M hlkreis, Austria, 2021.
- [66] I. F. G. Reis, I. Gonalves, M. A. R. Lopes, and C. H. Antunes, "A multi-agent system approach to exploit demand-side flexibility in an energy community," *Utilities Policy*, vol. 67, Dec. 2020, Art. no. 101114.
- [67] A. Mar, P. Pereira, and J. Martins, "Energy community flexibility solutions to improve users' wellbeing," *Energies*, vol. 14, no. 12, p. 3403, Jun. 2021.
- [68] L. Mendicino, D. Menniti, A. Pinnarelli, N. Sorrentino, P. Vizza, C. Alberti, and F. Dura, "DSO flexibility market framework for renewable energy community of nanogrids," *Energies*, vol. 14, no. 12, p. 3460, Jun. 2021.
- [69] M. Stephant, D. Abbes, K. Hassam-Ouari, A. Labrunie, and B. Robyns, "Distributed optimization of energy profiles to improve photovoltaic self-consumption on a local energy community," *Simul. Model. Pract. Theor.*, vol. 108, Apr. 2021, Art. no. 102242.
- [70] G. Reynders, R. A. Lopes, A. Marszal-Pomianowska, D. Aelenei, J. Martins, and D. Saelens, "Energy flexible buildings: An evaluation of definitions and quantification methodologies applied to thermal storage," *Energy Buildings*, vol. 166, pp. 372–390, May 2018.



**ARMIN VEICHTLBAUER** received the Diploma degree in applied informatics from the Paris Lodron University of Salzburg, in 1999. From 1999 to 2020, he was a Researcher and the Project Manager at the University of Salzburg, at Salzburg Research, and the University of Applied Sciences Salzburg, in the fields of software testing, network technologies, embedded systems, home automation, internet technologies, the IoT, and energy informatics. Since 2017, he has been a

Lecturer with the University of Applied Sciences Upper Austria, Hagenberg, where he is also a Researcher and the Project Manager, since 2020. As a Researcher, he has a long time of experience in the projects EU Minerva NODE (2002–2003), EU EFRE ESYCS (2003–2005), EU EFRE CoMoNet (2005–2006), FFG IV2Splus sTCnet (2009–2011), FFG ICT of the Future OpenNES (2014–2017), FFG Energy Research VirtueGrid (2017–2020), and FFG Energy Research SCSB (2018–2022). He was also the Project Manager of the projects FFG FIT-IT ASki (2005–2007) dealing with hybrid modeling of energy and information, FFG KIRAS RescueNet (2007–2008), FFG KIRAS CaR (2008–2009) dealing with ICT interfaces to safety technologies, and FFG COIN ROFCO (2009–2012) dealing with generic IP based ICT infrastructures in home and building automation.



**CHRISTOPH PRASCHL** received the bachelor's and master's degrees in software engineering from the University of Applied Sciences Upper Austria, Campus Hagenberg, in 2017 and 2019, respectively. Since 2017, he has been working as a Research Associate with the Research Group for Advanced Information Systems and Technology (AIST), University of Applied Sciences Upper Austria, in the field of software architecture, mixed reality, and computer vision. Additionally, he is

active as a Lecturer for software development with the University of Applied Sciences Wiener Neustadt (Campus Wieselburg), a databases at the Management Center Innsbruck, as well as a computer graphics and a image processing at the University of Applied Sciences Upper Austria (Campus Hagenberg).



**LUKAS GAISBERGER** received the Diploma degree from the University of Applied Sciences Upper Austria, Campus Wels, in 2018. Since 2018, he has been working with the University of Applied Sciences Upper Austria, Research Group ASIC, as a Research Associate in the field of photovoltaics, energy management systems, optimization in energy communities, and renewable production forecasting. In addition to that, he also works as a Lecturer for photovoltaics laboratory

exercises at the University of Applied Sciences. He was recently involved in the research projects FFG PV-go-Smart, FFG Storage Cluster South Burgenland, and FFG Serve-U. He is working with the Austrian partners in the IEA PVPS Task 15.



**GERALD STEINMAURER** (Member, IEEE) received the D.I. degree in electrical engineering from TU Graz and the Ph.D. degree in mechatronics on optimal control of hybrid systems with storage units from Johannes Kepler University Linz. In 2005, he took over the management and the research responsibility of the non-university research institution Austria Solar Innovation Center. Since 2016, he has been with the University of Applied Sciences Upper Austria and leads the

Center of Excellence Energy as well as the Research Group ASIC. His research interests include optimal power flow coordination of energy communities and renewable energy systems, mainly in controlling PV-inverters with integrated stores. He has coordinated several research projects on national level (Flagship project: Industrial Microgrids, CASGRIS-Center for Smart Grid Systems, PVgoSmart, and OPTTEMO). He also represents Austria in several Tasks (SHC—Solar Heating and Cooling Program and PVPS—Photovoltaic Power Systems Program) at the International Energy Agency IEA.



**THOMAS I. STRASSER** (Senior Member, IEEE) received the master's and Ph.D. degrees and the Venia Docendi (Habilitation) degree in automation from the Technische Universität Wien (TU Wien), Vienna, Austria, in 2001, 2003, and 2017, respectively. For several years, he has been a Senior Scientist with the Center for Energy of the AIT Austrian Institute of Technology. His main research interests include power utility/smart grid automation and corresponding engineering and validation approaches. Before joining AIT, he spent more than six years as a Senior Researcher investigating advanced and reconfigurable automation and control systems at PROFACTOR Research. He is active as a Senior Lecturer at TU Wien. He is leading and led several national and European research projects. He is a member of IEC and IEEE standardization working groups and as a Senior Member of IES (AdCom Member-at-Large 2018–2020 and TC Cluster Delegate Energy 2020–2021), SMCS (BoG Member-at-Large 2018–2020 and VP Systems Science and Engineering 2021–2022), SysCo (AdCom Member-at-Large 2021–2022), and PES. He serves also as the Austrian Representative in the CIGRE Study Committee C6. He is an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and further IEEE, Springer, and Hindawi journals.

...

# Bibliography

- [1] SMB Smart Grid Strategic Group (SG3). IEC Smart Grid Standardization Roadmap. Technical Report Ed. 1.0, International Electrotechnical Commission (IEC), Geneva, Switzerland, June 2010.
- [2] International Organization for Standardization. ISO/IEC 7498-1:1994 Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model, November 1994.
- [3] NIST. NIST Framework and Roadmap for Smart Grid Interoperability Standards. Technical Report NIST Special Publication 1108, National Institute of Standards and Technology - U.S. Department of Commerce (NIST), USA, 2010.
- [4] Institute of Electrical and Electronics Engineers (IEEE). IEEE Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads. Technical Report 2030-2011, Institute of Electrical and Electronics Engineers (IEEE), New York, USA, October 2011.
- [5] Filip Prössl Andrén, Thomas Strasser, Oliver Langthaler, Armin Veichtlbauer, Christian Kasberger, and Gregor Felbauer. Open and Interoperable ICT Solution for Integrating Distributed Energy Resources into Smart Grids. In *Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016)*, 2016.
- [6] IEC Central Office. The Smart Grid Standards Map, 2019.
- [7] Fraunhofer. OGEMA: Open Gateway Energy Management. Technical report, Fraunhofer IWES, 2012.
- [8] The openHAB Community and the openHAB Foundation e.V. Welcome to openHAB. Technical report, The openHAB Community and the openHAB Foundation e.V., 2019.

- [9] Manfred Broy, Mario Gleirscher, Stefano Merenda, Doris Wild, Peter Kluge, and Wolfgang Krenzer. Toward a Holistic and Standardized Automotive Architecture Description. *Computer*, 42(12):98–101, December 2009.
- [10] Bidyut Mukherjee, Roshan Lal Neupane, and Prasad Calyam. End-to-end iot security middleware for cloud-fog communication. In *Proceedings of the 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, pages 151–156. IEEE, 2017.
- [11] Abdel Rahman Alkhawaja, Luis Lino Ferreira, and Michele Albano. Message Oriented Middleware with QoS Support for Smart Grids. In *Proceedings of the INFOrum 2012 Conference on Embedded Systems and Real Time*, 2012.
- [12] Xuan Zhou, Rongpeng Li, Tao Chen, and Honggang Zhang. Network slicing as a service: enabling enterprises’ own software-defined cellular networks. *IEEE Communications Magazine*, 54(7):146–153, 2016.
- [13] Open Networking Foundation. Software-Defined Networking: The New Norm for Networks. Technical report, Open Networking Foundation, April 2012.
- [14] S. Rohjans, C. Danekas, and M. Usler. Requirements for Smart Grid ICT-architectures. In *Proc. IEEE Int. Conf. Innovative Smart Grid Technologies (ISGT Europe)*, pages 1–8, October 2012.
- [15] Boyang Zhou, Chunming Wu, Qiang Yang, and Xiang Chen. DRTP: A Disruption Resilient Hop-by-Hop Transport Protocol for Synchrophasors Measurement in Electric Transmission Grids. *IEEE Access*, 10:133898–133914, 2022.
- [16] The GridWise Architecture Council. Gridwise interoperability context-setting framework. Technical report, The GridWise Architecture Council, 2008.
- [17] Smart Grid Coordination Group. Smart Grid Reference Architecture. Technical report, CEN/Cenelec/ETSI Smart Grid Coordination Group, November 2012.
- [18] Smart Grid Coordination Group. Smart Grid Mandate. Technical report, European Commission Directorate-General for Energy, November 2012.
- [19] International Electrotechnical Commission. IEC 61970-301: Energy management system application program interface (EMS-API) - Part 301: Common information model (CIM) base, 2016.
- [20] Sebastian Rohjans. *(S2)In – Semantic Service Integration for Smart Grids*. PhD thesis, Carl von Ossietzky University, Oldenburg, Germany, September 2012.

- [21] International Electrotechnical Commission. IEC 61850-1: Communication networks and systems for power utility automation - Part 1: Introduction and overview, 2013.
- [22] International Electrotechnical Commission. IEC 62056: Electricity metering data exchange - The DLMS/COSEM suite, 2014.
- [23] OPC Foundation. OPC – The Interoperability Standard for Industrial Automation & Other, 2012.
- [24] Patrick Denzler, Thomas Frühwirth, Daniel Scheuchenstuhl, Martin Schoeberl, and Wolfgang Kastner. Timing analysis of tsn-enabled opc ua pubsub. In *2022 IEEE 18th International Conference on Factory Communication Systems (WFCS)*, pages 1–8. IEEE, 2022.
- [25] Peter Saint-Andre. Extensible messaging and presence protocol (XMPP): Core. RFC6120, 2011.
- [26] Object Management Group. Data distribution service (dds). Technical report, Object Management Group, April 2015.
- [27] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC3261, June 2002.
- [28] World Wide Web Consortium. Extensible markup language (xml) 1.0 (fifth edition). Technical report, World Wide Web Consortium, 2008.
- [29] T. Bray. The JavaScript Object Notation (JSON) Data Interchange Format. RFC 8259, 2017.
- [30] YAML Language Development Team. YAML Ain’t Markup Language (YAML™) version 1.2, 2021.
- [31] International Electrotechnical Commission. IEC 61499: Function blocks, 2012.
- [32] A. Zoitl, T. Strasser, and A. Valentini. Open source initiatives as basis for the establishment of new technologies in industrial automation: 4DIAC a case study. In *2010 IEEE International Symposium on Industrial Electronics (ISIE)*, Bari, Italy, 2010.
- [33] International Electrotechnical Commission. IEC 8824-1: Abstract Syntax Notation One (ASN.1): Specification of Basic Notation, November 2008.

- [34] Mike Pichler, Armin Veichtlbauer, and Dominik Engel. Evaluation of OSGi-based Architectures for Customer Energy Management Systems. In *Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT2015)*, Sevilla, March 2015.
- [35] The OSGi Alliance. OSGi core release 5, 2012.
- [36] Armin Veichtlbauer, Dominik Engel, Fabian Knirsch, Oliver Langthaler, and Felix Moser. Advanced Metering and Data Access Infrastructures in Smart Grid Environments. In *Proceedings of the 7th International Conference on Sensor Technologies and Applications (SensorComm 2013)*, Barcelona, August 2013.
- [37] International Electrotechnical Commission. Iec 62746-10-1:2018: Systems interface between customer energy management system and the power management system - part 10-1: Open automated demand response, 2018.
- [38] European Telecommunications Standards Institute. ETSI TS 104 001 Open Smart Grid Protocol, 2016.
- [39] International Electrotechnical Commission. ISO/IEC 14908: Information technology – Control network protocol, 2012.
- [40] International Electrotechnical Commission. IEC 60870-1: Telecontrol equipment and systems - Part 1: General considerations, 1988.
- [41] Jon Postel. Transmission Control Protocol – DARPA Internet Program Protocol Specification. RFC 793, 1981.
- [42] Jon Postel. Internet Protocol – DARPA Internet Program Protocol Specification. RFC 791, 1981.
- [43] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC8200, July 2017.
- [44] Zach Shelby and Carsten Bormann. *6LoWPAN: The Wireless Embedded Internet*, volume 43. John Wiley & Sons, 2011.
- [45] Jon Postel. User Datagram Protocol. RFC 768, August 1980.
- [46] J. Iyengar and M. Thomson. QUIC: A UDP-Based Multiplexed and Secure Transport. RFC 9000, 2021.
- [47] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 8684, 2020.

- [48] Mubashir Husain Rehmani, Alan Davy, Brendan Jennings, and Chadi Assi. Software Defined Networks-Based Smart Grid Communication: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials*, 21(3):2637–2670, 2019.
- [49] Khirota Gorgees Yalda, Diyar Jamal Hamad, and Nicolae Țăpuș. A survey on Software-defined Wide Area Network (SD-WAN) architectures. In *Proceedings of the 2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–5. IEEE, 2022.
- [50] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, June 2001.
- [51] Armin Veichtlbauer, Alexander Heinisch, and Ferdinand von Tüllenburg. Virtualisierung für Energienetze. In *Newsletter 2020-03 - Virtualisierungstechnologien für das Energienetz*. OVE - Österreichischer Verband für Elektrotechnik, March 2020. In German.
- [52] Meryem Simsek, Adnan Aijaz, Mischa Dohler, Joachim Sachs, and Gerhard Fettweis. 5G-enabled tactile internet. *IEEE Journal on selected areas in communications*, 34(3):460–473, 2016.
- [53] Thomas Stüber, Lukas Osswald, Steffen Lindner, and Michael Menth. A Survey of Scheduling Algorithms for the Time-Aware Shaper in Time-Sensitive Networking (TSN). *IEEE Access*, 2023.
- [54] Ethernet POWERLINK Standardisation Group. Ethernet POWERLINK Communication Profile Specification DS301, 2016.
- [55] Shalitha Wijethilaka and Madhusanka Liyanage. Survey on network slicing for Internet of Things realization in 5G networks. *IEEE Communications Surveys & Tutorials*, 23(2):957–994, 2021.
- [56] Smart Grid Coordination Group. Framework Document. Technical report, CEN-CENELEC-ETSI Smart Grid Coordination Group, November 2012.
- [57] Mathias Uslar and Dominik Engel. Towards generic domain reference designation: How to learn from smart grid interoperability. *D-A-Ch Energieinformatik*, 1:1–6, November 2015.
- [58] The SunSpec Alliance. Sunspec device information model specification. Technical report, SunSpec Alliance, 2019.

- [59] Rob Sherwood, Glen Gibb, and K-K et al. Yap. FlowVisor: A network virtualization layer. *OpenFlow Switch Consortium, Tech. Rep*, 2009.
- [60] Saba Al-Rubaye, Ekhlas Kadhum, Qiang Ni, and Alagan Anpalagan. Industrial internet of things driven by sdn platform for smart grid resiliency. *IEEE Internet of Things Journal*, 6(1), February 2019.
- [61] Monika Wenger, Alois Zoitl, Roman Froschauer, Martijn Rooker, Gerhard Ebenhofer, and Thomas Strasser. Model-driven Engineering of Networked Industrial Automation Systems. In *Proceedings of the 2010 8th IEEE International Conference on Industrial Informatics (INDIN 2010)*, 2010.
- [62] International Electrotechnical Commission. IEC PAS62559: IntelliGrid methodology for developing requirements for energy systems. Technical report, International Electrotechnical Commission (IEC), Geneva, Switzerland, 2008.
- [63] Object Management Group. About the Unified Modeling Language Specification Version 2.5.1, 2017.
- [64] German Federal Office for Information Security. Protection Profile for the Gateway of a Smart Metering System (Smart Meter Gateway PP). Technical report, German Federal Office for Information Security, Bonn, Germany, March 2014.
- [65] Friederich Kupzog, Armin Veichtlbauer, Alexander Heinisch, Ferdinand von Tülingen, Oliver Langthaler, Ulrich Pache, Oliver Jung, Reinhard Frank, and Peter Dorfinger. The Impact of Virtualisation Techniques on Power System Control Networks. *Electronics*, 9(9):1433–1454, September 2020.
- [66] Elyoenai Egozcue, Daniel Herreras Rodriguez, Jairo Alonso Ortiz, Victor Fidalgo Villar, and Luis Tarrafeta. Smart Grid Security - Annex II. Security aspects of the smart grid. Technical report, European Network and Information Security Agency (ENISA), April 2012.
- [67] Armin Veichtlbauer, Manuel Parfant, Oliver Langthaler, Filip Prössl Andrén, and Thomas Strasser. Evaluating XMPP Communication in IEC 61499-based Distributed Energy Applications. In *Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016)*, 2016.
- [68] Tobias Gawron-Deutsch, Florian Kintzler, Friederich Kupzog, Ferdinand von Tülingen, and Ulrich Pache. Grid Based Routing - Virtuegrid SDN Whitepaper. In *Proceedings of the 2018 Global Internet of Things Summit (GIoTTS 2018)*, pages 1–6. IEEE, 2018.



- [69] Julian Gugeneder, Christoph Muck, Gerald Steinmaurer, and Armin Veichtlbauer. Simulative Analyse der potenziellen Energieflexibilitäten von Einzelhaushalten. In *Tagungsband des 16. Symposiums Energieinnovation*, Graz, Austria, February 2020. In German.
- [70] International Electrotechnical Commission. ISO/IEC 14977: Information technology - Syntactic metalanguage - Extended BNF. Technical report, International Electrotechnical Commission, 1996.
- [71] Oliver Jung, Paul Smith, Julian Magin, and Lenhard Reuter. Anomaly Detection in Smart Grids based on Software Defined Networks. In *Proceedings of the 8th International Conference on Smart Cities and Green ICT Systems (SmartGreens 2019)*, 2019.
- [72] Christof Brandauer, Stefan Linecker, Filip Prörtl Andrén, Catalin Gavriluta, Thomas I. Strasser, Armin Veichtlbauer, Gerald Steinmaurer, Jürgen Resch, and Sebastian Schöndorfer. A Collaborative Engineering and Validation Framework for Smart Grid Automation Applications - The PowerTeams Approach. In *Proceedings of the 28th International Conference and Exhibition on Electricity Distribution (CIRED 2023)*, 2023.
- [73] Ferdinand von Tüllenbug, Peter Dorfinger, Armin Veichtlbauer, Ulrich Pache, Oliver Langthaler, Helmut Kapoun, Christian Bischof, and Friederich Kupzog. Virtualising Redundancy of Power Equipment Controllers Using Software-defined Networking. *Energy Informatics*, 2(14), September 2019.
- [74] Armin Veichtlbauer and Thomas Pfeiffenberger. Generic Middleware for User-friendly Control Systems in Home and Building Automation. *International Journal on Advances in Networks and Services*, 6(1&2):51–67, July 2013.
- [75] Armin Veichtlbauer, Ulrich Pache, Oliver Langthaler, Helmut Kapoun, Christian Bischof, Ferdinand von Tüllenbug, and Peter Dorfinger. Enabling Application Independent Redundancy by Using Software Defined Networking. In *Proceedings of the IEEE 10th International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT 2018)*, Moscow, Russia, November 2018.
- [76] Armin Veichtlbauer, Oliver Langthaler, Dominik Engel, Christian Kasberger, Filip Prörtl Andrén, and Thomas Strasser. Towards Applied Security-by-Design for DER Units. In *Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016)*, 2016.

- [77] Christian Neureiter, Günther Eibl, Armin Veichtlbauer, and Dominik Engel. Towards a Framework for Engineering Smart-Grid-Specific Privacy Requirements. In *Proceedings of the 39th Annual Conference of the IEEE Industrial Electronics Society (IECON 2013), Special Session on Energy Informatics*, Vienna, Austria, November 2013. IEEE.

# Appendix A

## List of Scientific Publications

In the following, additional publications, which have been co-authored by the candidate and are relevant to the dissertation topic, are listed.

### A.1 Peer-reviewed International Journals

- F. Kupzog, A. Veichtlbauer, A. Heinisch, F. v. Tüllenbug, O. Langthaler, U. Pache, O. Jung, R. Frank, P. Dorfinger:

**The Impact of Virtualisation Techniques on Power System Control Networks.**

*In: Electronics, Vol. 9, No. 9, pp. 1433, Sep. 2020 [65]*

- F. v. Tüllenbug, P. Dorfinger, A. Veichtlbauer, U. Pache, O. Langthaler, H. Kapoun, C. Bischof, F. Kupzog:

**Virtualising Redundancy of Power Equipment Controllers Using Software-defined Networking.**

*In: Energy Informatics, Vol. 2, Suppl. 1, No. 14, Sep. 2019 [73]*

- A. Veichtlbauer, T. Pfeiffenberger:

**Generic Middleware for User-friendly Control Systems in Home and Building Automation.**

*In: International Journal on Advances in Networks and Services, Vol. 6, No. 1 & 2, pp. 51 - 67, Jul. 2013 [74]*

## A.2 Peer-reviewed Conference Proceedings

- C. Brandauer, S. Linecker, F. Prörtl Andrén, C. Gavriluta, T. I. Strasser, A. Veichtlbauer, G. Steinmaurer, J. Resch, S. Schöndorfer:

**A Collaborative Engineering and Validation Framework for Smart Grid Automation Applications - The PowerTeams Approach.**

*In: Proceedings of the 28th International Conference and Exhibition on Electricity Distribution (CIRED 2023), Rome, Jun. 2023 [72]*

- J. Gugeneder, C. Muck, G. Steinmaurer, A. Veichtlbauer:

**Simulative Analyse der potenziellen Energieflexibilitäten von Einzelhaushalten.**

*In: Tagungsband des 16. Symposiums Energieinnovation, Graz/Austria, Feb. 2020 (in German) [69]*

- A. Veichtlbauer, U. Pache, O. Langthaler, H. Kapoun, C. Bischof, F. v. Tüllenburg, P. Dorfinger:

**Enabling Application Independent Redundancy by Using Software Defined Networking.**

*In: Proceedings of the IEEE 10th International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT 2018), Moscow, Nov. 2018 [75]*

- F. Prörtl Andrén, T. I. Strasser, O. Langthaler, A. Veichtlbauer, C. Kasberger, G. Felbauer:

**Open and Interoperable ICT Solution for Integrating Distributed Energy Resources into Smart Grids.**

*In: Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Sep. 2016 [5]*

- A. Veichtlbauer, M. Parfant, O. Langthaler, F. Prörtl Andrén, T. I. Strasser:

**Evaluating XMPP Communication in IEC 61499-based Distributed Energy Applications.**

*In: Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Sep. 2016 [76]*

- A. Veichtlbauer, O. Langthaler, D. Engel, C. Kasberger, F. Prössl Andrén, T. I. Strasser:

**Towards Applied Security-by-Design for DER Units.**

*In: Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Sep. 2016 [67]*

- M. Pichler, A. Veichtlbauer, D. Engel:

**Evaluation of OSGi-based Architectures for Customer Energy Management Systems.**

*In: Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT 2015), Sevilla, Mar. 2015 [34]*

- C. Neureiter, G. Eibl, A. Veichtlbauer, D. Engel:

**Towards a Framework for Engineering Smart-Grid-Specific Privacy Requirements.**

*In: Proceedings of the IEEE IECON 2013, Special Session on Energy Informatics, Vienna, Nov. 2013 [77]*

- A. Veichtlbauer, D. Engel, F. Knirsch, O. Langthaler, F. Moser:

**Advanced Metering and Data Access Infrastructures in Smart Grid Environments.**

*In: Proceedings of the 7th International Conference on Sensor Technologies and Applications (SensorComm 2013), Barcelona, Aug. 2013 [36]*

## A.3 Invited Papers and Book Chapters

- A. Veichtlbauer, A. Heinisch, F. v. Tüllenburger:

**Virtualisierung für Energienetze.**

*In: OVE Newsletter Energie 2020, Mar. 2020 (in German) [51]*

# Appendix B

## Abbreviations

<b>AAA</b>	Authentication, Authorization, and Accounting
<b>AMI</b>	Advanced Metering Infrastructure
<b>API</b>	Application Programming Interface
<b>ARQ</b>	Automatic Repeat ReQuest
<b>ASN.1</b>	Abstract Syntax Notation One
<b>CEMS</b>	Customer Energy Management Systems
<b>CIA</b>	Confidentiality, Integrity, Availability
<b>CIM</b>	Common Information Model
<b>CNP</b>	Control Network Protocol
<b>COSEM</b>	Companion Specification for Energy Metering
<b>DDS</b>	Data Distribution Service
<b>DER</b>	Distributed Energy Resources
<b>DLMS</b>	Device Language Message Specification
<b>DoS</b>	Denial of Service
<b>DR</b>	Demand/Response
<b>DSO</b>	Distribution System Operator
<b>EBNF</b>	Extended Backus-Naur Form

---

<b>ESB</b>	Enterprise Service Bus
<b>EV</b>	Electric Vehicle
<b>GUI</b>	Graphical User Interface
<b>GWAC</b>	GridWise Architecture Council
<b>HA</b>	Home Automation
<b>HAL</b>	Hardware Abstraction Layer
<b>HMI</b>	Human-Machine Interface
<b>HTML</b>	Hypertext Markup Language
<b>IaaS</b>	Infrastructure as a Service
<b>ICT</b>	Information and Communication Technology
<b>IDS</b>	Intrusion Detection System
<b>IEC</b>	International Electrotechnical Commission
<b>IED</b>	Intelligent Energy Device
<b>IEEE</b>	Institute of Electrical and Electronics Engineers
<b>IoT</b>	Internet of Things
<b>IP</b>	Internet Protocol
<b>JSON</b>	JavaScript Object Notation
<b>LAN</b>	Local Area Network
<b>LDAP</b>	Lightweight Directory Access Protocol
<b>LON</b>	Local Operating Network
<b>LV</b>	Low Voltage
<b>MAC</b>	Medium Access Control
<b>MQTT</b>	Message Queuing Telemetry Transport
<b>MPLS</b>	Multi-protocol Label Switching

---

<b>MV</b>	Medium Voltage
<b>NaaS</b>	Network as a Service
<b>NFV</b>	Network Function Virtualization
<b>OLTC</b>	On Load Tap Changer
<b>OPC UA</b>	Open Process Control Unified Architecture
<b>OpenADR</b>	Open Automated Demand Response
<b>OpenALP</b>	Open Application Layer Protocol
<b>OS</b>	Operating System
<b>OSGi</b>	Open Services Gateway initiative
<b>OSGP</b>	Open Smart Grid Protocol
<b>OSI</b>	Open Systems Interconnection
<b>OT</b>	Operating Technology
<b>P4</b>	Programming Protocol-independent Packet Processors
<b>PDP</b>	Policy Decision Point
<b>PEP</b>	Policy Enforcement Point
<b>PKI</b>	Public Key Infrastructure
<b>PoC</b>	Proof of Concept
<b>PV</b>	Photovoltaic
<b>QoS</b>	Quality of Service
<b>QUIC</b>	Quick UDP Internet Connections
<b>RBAC</b>	Role-based Access Control
<b>RTU</b>	Remote Terminal Unit
<b>SAP</b>	Service Access Point
<b>SCADA</b>	Supervisory Control and Data Acquisition



---

<b>SDN</b>	Software Defined Networking
<b>SD-WAN</b>	Software Defined Wide Area Networking
<b>SGAM</b>	Smart Grid Architecture Model
<b>SIP</b>	Session Initiation Protocol
<b>SLA</b>	Service Level Agreement
<b>SOA</b>	Service Oriented Architecture
<b>SOAP</b>	Simple Object Access Protocol
<b>SoC</b>	State of Charge
<b>TCP</b>	Transmission Control Protocol
<b>TLS</b>	Transport Layer Security
<b>TSN</b>	Time Sensitive Networking
<b>UDP</b>	User Datagram Protocol
<b>UML</b>	Unified Modeling Language
<b>URI</b>	Unified Resource Identifier
<b>UTF</b>	Unicode Transformation Format
<b>VCI</b>	Virtualized Communication Infrastructure
<b>VFB</b>	Virtual Functional Bus
<b>VLAN</b>	Virtual Local Area Network
<b>VM</b>	Virtual Machine
<b>VPN</b>	Virtual Private Network
<b>VxLAN</b>	Virtual Extended Local Area Network
<b>XML</b>	Extensible Markup Language
<b>XMPP</b>	Extensible Messaging and Presence Protocol
<b>XSD</b>	XML Schema Definition
<b>YAML</b>	Yet Another Markup Language