


Article

Smart Grid Virtualisation for Grid-Based Routing

Armin Veichtlbauer ^{1,*} , Alexander Heinisch ², Ferdinand von Tüllenburg ³, Peter Dorfinger ³, Oliver Langthaler ⁴ and Ulrich Pache ⁴

¹ Campus Hagenberg, University of Applied Sciences Upper Austria, 4232 Hagenberg, Austria

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

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

⁴ 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, Pfeiffenberger 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.

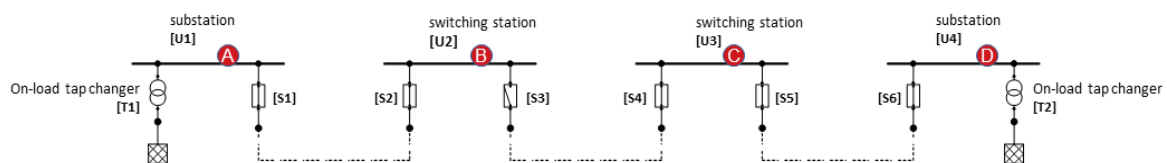


Figure 1. Testbed architecture.

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).

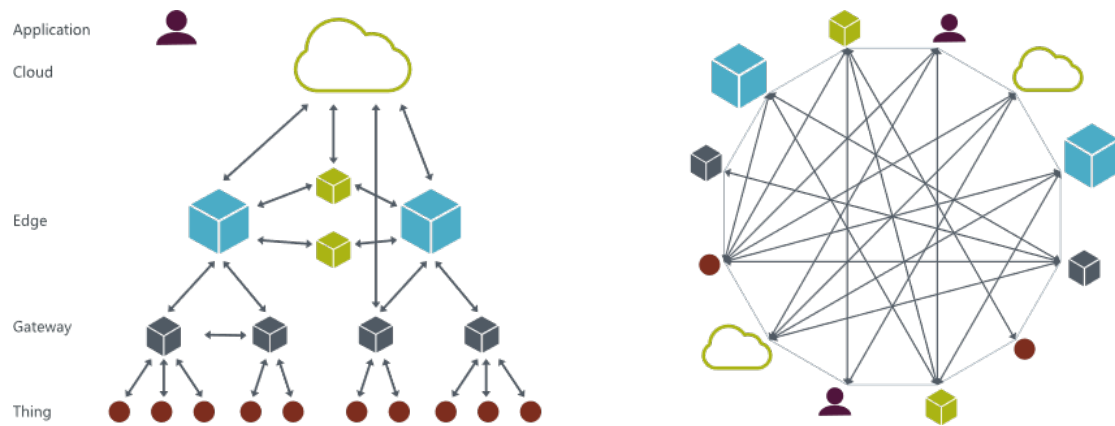


Figure 2. Conventional cloud/edge architecture and Coaty architecture (Source: [31]).

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].

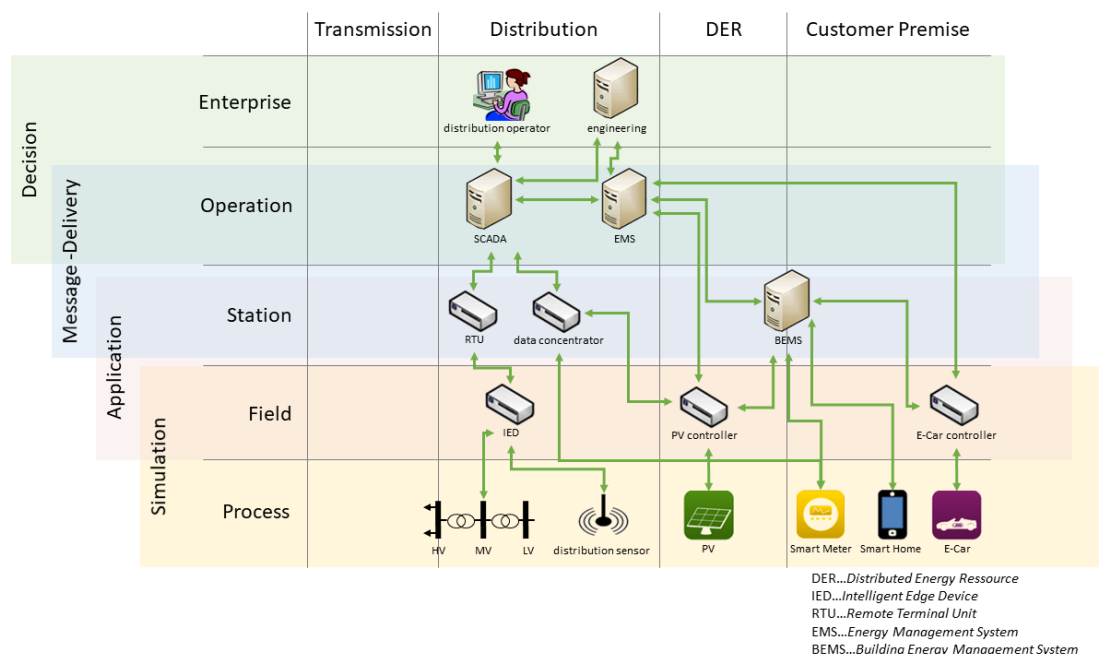


Figure 3. Mapping to SGAM.

- **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 OLTC) 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:

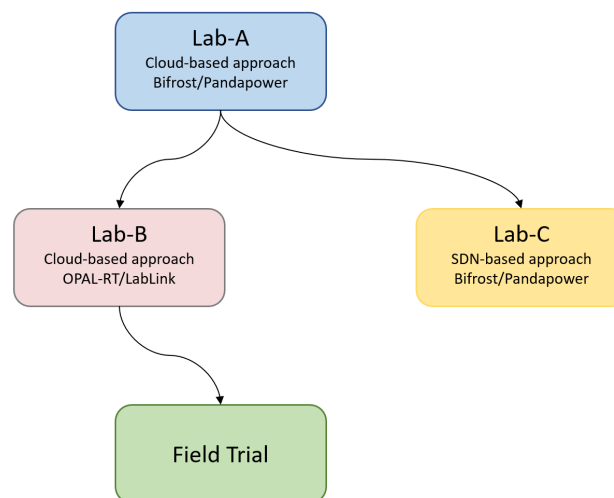


Figure 4. Lab setups.

- 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.

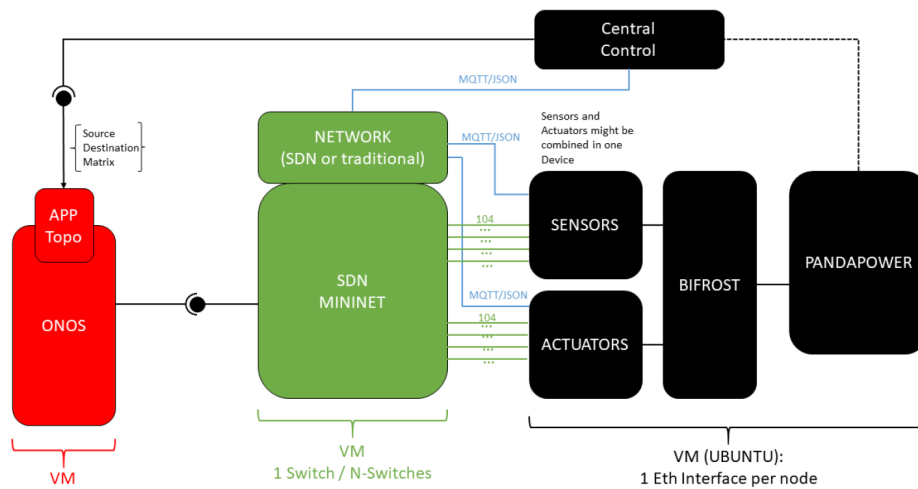


Figure 5. Lab C Virtual Machine Setup.

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.

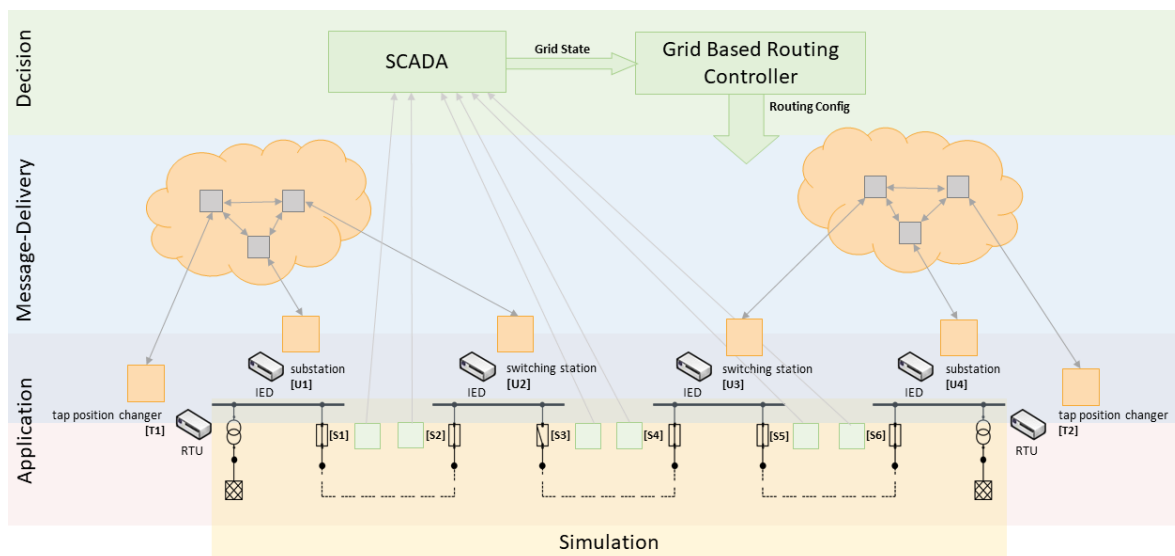


Figure 6. Architecture: Mapping of the top level components.

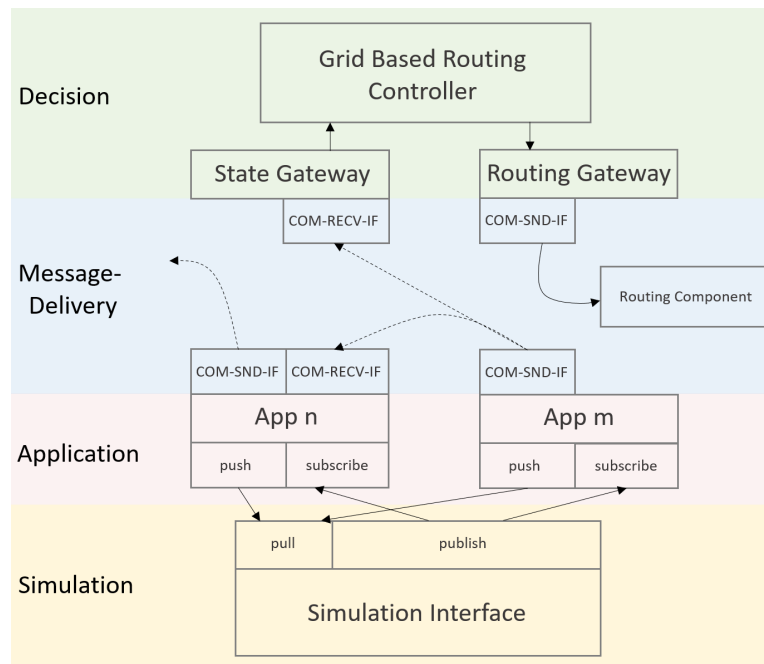


Figure 7. Architecture: Overview of the top level components.

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.

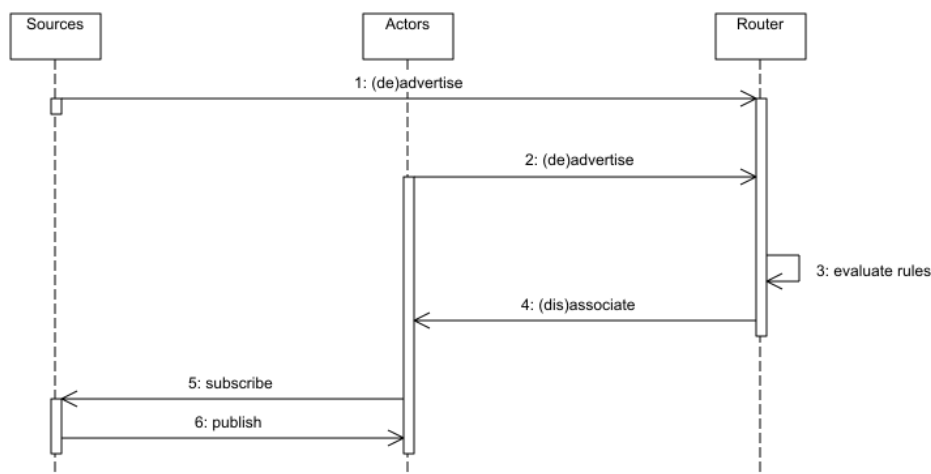


Figure 8. Coaty: Interactions of Sensors and Actors with the Router.

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.

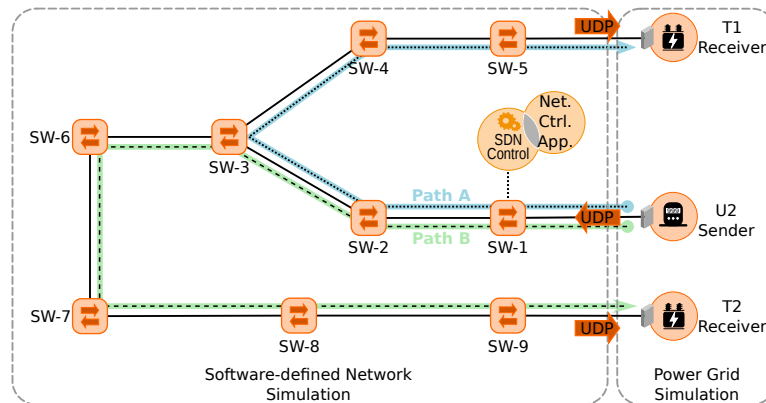


Figure 9. The SDN prototype as used within the project.

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.

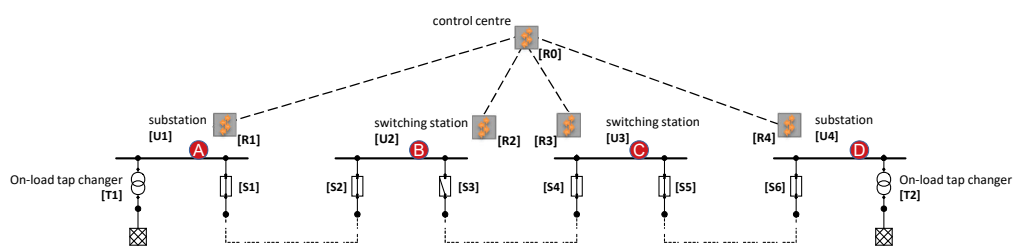


Figure 10. Evaluation scenario.

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 (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 (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/>).