Improving the Reliability of Smart Distribution Grids Using Software-based Networking

Gerhard Errol Brown

Supervisor:

Mr Neco Ventura

Co-supervisor:

Dr Joyce Mwangama

Dissertation submitted to the Department of Electrical Engineering in fulfilment of the requirements for the Master of Science in Electrical Engineering degree at the University of Cape Town

October 2020

APPLICATION FORM

Please Note:
Any person planning to undertake research in the Faculty of Engineering and the Built Environment (EBE) at the University of Cape Town is required to complete this form **before** collecting or analysing data. The objective of submitting this application *prior* to embarking on research is to ensure that the highest ethical standards in research, conducted under the auspices of the EBE Faculty, are met. Please ensure that you have read, and understood the **EBE Ethics in Research Handbook** (available from the UCT EBE, Research Ethics website) prior to completing this application form: http://www.ebe.uct.ac.za/ebe/research/ethics1

APPLICANT'S DETAILS		
Name of principal researcher, student or external applicant		Gerhard Errol Brown
Department		Electrical Engineering
Preferred email address of applicant:		Gerhard.brown@gmail.com
If Student	Your Degree: e.g., MSc, PhD, etc.	MSc (Eng) Electrical Engineering
	Credit Value of Research: e.g., 60/120/180/360 etc.	180
	Name of Supervisor (if supervised):	Neco Ventura
If this is a researchcontract, indicate the source of funding/sponsorship		Click here to enter text.
Project Title		The Internet of Things in Metropolitan Smart Grids

I hereby undertake to carry out my research in such a way that:

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

SIGNED BY	Full name	Signature	Date
Principal Researcher/ Student/External applicant	Gerhard Errol Brown	Signed by candidate	**13 Mar 2019**

APPLICATION APPROVED BY	Full name	Signature	Date
Supervisor (where applicable)	Neco Ventura	Signature Removed	Click here to enter a date.
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section1; and for all Undergraduateresearch (Including Honours).	A/Prof Olabisi Falowo Click here to enter text.	Signature Removed	15/09/19 Click here to enter a date.
Chair : Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the above questions.			

Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for the properly acknowledged, is my own. This dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

Signed by candidate

Gerhard Errol Brown
October 2020

Acknowledgements

I would like to thank the following people for their help and support throughout this project:

Mr Neco Ventura and Dr Joyce Mwangama, for their excellent supervision and guidance, especially with the challenges we experienced during COVID-19 pandemic.

Prof. Thomas Magedanz, for his lectures on Machine-to-Machine communication, the Industrial Internet of Things, 5G communication and the OpenMTC platform.

Mr Paul Inglesby, for educating me on the foundational principles of IoT and for sharing his experience with IoT communication standards.

My uncle Errol and aunt Anni, for reading through my dissertation and pointing out things I could change to improve its readability.

My parents, for all their support and encouragement.

My wife, for taking care of me and our baby boy while I spent hours staring at a computer screen.

I would also like to thank the City of Cape Town for providing information about the Cape Town distribution grid that made this study possible and I acknowledge that views expressed in this study are not regarded as official City of Cape Town policy.

Abstract

Smart grid is a combination of technologies that emerged in response to the rapid changes in the way humans generate, transfer, distribute and use energy. The smart grid paradigm shifts the focus from bulk generation with centralised grid control to distributed electricity generation, energy storage and greater consumer participation in grid operations. Many organisations and institutions are contributing to this initiative by developing new frameworks, architectures and standards that aim to support its adoption and improvement. Most of these new developments are based on modern Information Communication Technology (ICT) such as the Internet of Things (IoT). Although many governments have made commitments to pursue smart grid as part of their national policies and strategies, some nations, especially those with developing economies, have struggled to make significant strides in smart grid deployment.

An important characteristic of smart grid is its resilience and its ability to self-heal, enabled by better use of grid knowledge and the distribution of grid intelligence. This creates a challenge for many utilities that need to improve their existing grid ICT infrastructure to meet more stringent communication requirements. These requirements will likely include very high communication reliability with high throughput and low latency. Although most networks can be developed or improved to meet these requirements using modern network hardware, the cost and complexity involved in implementing such designs in large-scale distribution grid networks may be too high. To overcome this challenge alternative ways of designing smart grid communication networks using software approaches are needed.

This study proposes a new software-based networking platform, based on Industrial Internet of Things (IIoT) technology, that aims to support smart grid reliability by enabling reliability-centred smart grid systems and by reacting immediately to communication problems using real-time monitoring techniques. Using the principles of software defined networking (SDN), network functions virtualisation (NFV) and machine-to-machine communication (M2M), the design proposed in this work aims to provide a more flexible and affordable approach to developing and maintaining large-scale grid communication networks while offering several features that improve grid reliability and performance.

Experimental simulations were conducted with this architecture implemented in an emulated network environment, using a topology based on a model of a real city distribution grid. Results from the experimental evaluation show that a software-based communication network is easy to set up, maintain and scale using virtual machines capable of running on existing grid IT infrastructure. Furthermore, the results show that by using the features of SDN, NFV and M2M a smart grid communication network can be designed that can automatically detect and recover from at least six different simulated communication failures without impacting the operation of a functional smart grid application supported by the network. The results also support this platform's capability to reduce network congestion using a scheduled network data buffering service, resulting in end-to-end network latency improvements from 0.6 seconds to 0.05

milliseconds. From these results, we conclude that software-based networking can offer promising design alternatives for smart distribution grids, capable of improving the grid's overall reliability. This conclusion is drawn from the fact that software-based networks not only offer many features that can improve communication reliability and performance, but also have the potential to reduce the cost and complexity of network implementation and maintenance. This study can potentially improve the uptake of smart grid as it offers utilities design options that are more flexible and affordable to implement and maintain.

Table of contents

Table of Figures

List of Tables

List of Acronyms

AGC	Automatic gain control
AMI	Advanced metering infrastructure
API	Application Programming Interface
CLI	Command Line Interface
CT-IAP	Communication Technology Interoperability Architectural Perspective
CVO	Composite Virtual Objects
DNS	Domain Name Service
E2E	End-to-end
EAM	Enterprise Asset Management
EAN	Extended Area Network
FAN	Field Area Network
FCAPS	Faults, Configuration, Accounting, Performance and Security
FFR	Fast Failover Recovery
FTP	File Transfer Protocol
FTPS	File Transfer Protocol over Secure Sockets Layer/ Transport Layer Security
GENI	Global Environment for Network Innovations
GOOSE	Generic Object-Oriented Substation Events
GUI	Graphical User Interface
HAN	Home Area Network
HTTP	Hypertext Transfer Protocol
IAN	Industrial Area Network
ICT	Information Communication Technology
IEC	International Electrotechnical Commission
IED	Intelligent Electronic Device
IEEE	Institute of Electrical and Electronics Engineers
IIoT	Industrial Internet of Things
IoT	Internet of Things
IP	Internet Protocol
IPE	Internetworking Proxy Entity
IT	Information Technology
ITU	International Telecommunications Union
Kbps	Kilobits per second
km	Kilometre
LAN	Local Area Network
LPWAN	Low-powered Wide Area Networks
M2M	Machine-to-Machine (Communication)
MAN	Metropolitan Area Network
MANO	Management and Orchestration
MAS	Multi-agent System
Mbps	Megabits per second
MDM	Meter Data Management
MEC	Mobile Edge Computing
ms	Millisecond
MTTR	Mean Time to Repair
NAN	Neighbourhood Area Network
NFV	Network Functions Virtualisation
NFVI	Network Functions Virtualisation Infrastructure
NIST	National Institute for Standards and Technology
ONOS	Open Network Operating System
OVS	Open Virtual Switch
PAAS	Platform as a Service
PMU	Phasor Measurement Unit

QoS	Quality of Supply
REST	Representational State Transfer
RTL	Round Trip Latency
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SCS	Substation Control Systems
SDE	Software Development Environment
SDN	Software Defined Networking
SDN-SG	Software Defined Networking – Smart Grid
SDS	Software Defined Systems
SDSec	Software Defined Security
SFTP	File Transfer Protocol over Secure Shell
SGAM	Smart Grid Architecture Model
SV	Sampled Values
TCP	Transmission Control Protocol
VM	Virtual Machine
VNF	Virtualised Network Function
VNFC	Virtual Network Function Chains
VO	Virtual Objects
WAN	Wide Area Network
WSN	Wireless Sensor Network

Chapter 1

Introduction

For many years, the conventional electric power grid was built to deliver electricity from large power stations to customers. These power grids expanded over time to accommodate the energy needs of millions of people, resulting in the development of massive energy networks that can span across entire continents. In the past decade, electricity supply disruptions have inconvenienced the daily lives of people that make use of these energy networks. Many of these disruptions can be attributed to failures in the grid infrastructure used to generate, transfer, and distribute electricity. There is thus a need to make the current electricity grid more reliable, efficient, and secure. This can be achieved by the next-generation power grid, i.e., the smart grid, which is characterised by a two-way flow of electricity and information, creating an automated, widely distributed energy delivery network [1].

Smart grid incorporates diversified, renewable energy resources and systems that enable more interaction between customers and utilities. It also includes automated and intelligent management systems that focus on improving the grid's effectiveness and efficiency. These systems are envisioned to offer a variety of advantages over current systems in terms of digitalization, customisation, flexibility, intelligence, sustainability, and resilience which entitles the name smart grid to these next generation power systems. Technologies that have emerged as part of the smart grid paradigm include smart control centres that monitor and control grid devices remotely in real-time; smart transmission components that employ new technologies to enhance power quality; and smart substations that can coordinate local devices self-consciously. The significant advancements in system automation and intelligence offered by these technologies have even led to the proposal of the concept of Energy Internet.

The system-wide intelligence that characterises smart grid is only feasible if the information exchange among its various functional units is expedient, reliable, and trustable. A communication network that is fast, reliable, and secure is therefore one of the keys enabling technologies of a smart grid. Unfortunately, the current communication capabilities of existing electric power grids are limited, which means they do not yet meet the communication requirements that most smart grid systems demand. Most of these communication networks are small in scale and restricted to local regions. Others were developed mainly to support basic functionalities for specific monitoring and control systems such as Supervisory Control and Data Acquisition (SCADA). A diverse collection of systems and technologies will be used in the world's modern power grids. These systems will be spread across entire cities, countries and continents and will be connected into the same management network. Real-time bidirectional communication will form the foundation of the tasks that utilities will perform when using these systems and in certain cases, these tasks will require time-sensitive and data-intensive information exchanges.

Because the world is currently still in a transitional phase of shifting to smart grids, studies on the communication architectures that will support various automated and intelligent grid management systems are still in its early stages. Although the energy industry has shown great interest in smart grid, its adoption by utilities has also been slower than anticipated. This is especially true for nations with developing economies that do not have access to resources for major grid improvements [2], [3]. Besides the regulation, standardisation, socio-economic and economic challenges utilities face with implementing smart grids, there are also various technical challenges that are awaiting solutions [4] - [6]. Many of these challenges can be attributed to the fact that smart grids have been categorised as complex, dynamic systems-of-systems that integrate within legacy systems and infrastructure [7].

One of the major technical challenges smart grid poses, is an increased dependence on better information communication technology (ICT). Data is the cornerstone of all smart grid systems and the need for accurate data that is readily available drives requirements for high performance communication networks that must be stable and very reliable. These networks must be capable of supporting mission critical grid functions, while accommodating a wide variety of devices and applications, often developed by different companies according to different standards. Furthermore, many power grids already have existing communication infrastructure in place that might not be capable of meeting the new stringent communication requirements of modern smart grid solutions. Because many of these legacy networks cannot be retrofitted with a new technology, it leaves utilities with the massive task of performing major network overhauls that can lead to early retirement of expensive infrastructure that could otherwise still have had a long functional life span.

Designing reliable communication networks that can meet the requirements of the next-generation power grid is not a simple task. Besides the varying requirements for both time-sensitive and high-data-volume systems that share the same network infrastructure, network architects must also consider how each application and device introduced into the network may impact the grid's overall reliability. The dynamic nature of large-scale grid networks also makes them difficult to manage, operate and maintain without the appropriate tools for network monitoring and control. Therefore, many modern grid solutions include software that supports centralised network monitoring and control of faults, configuration, accounting, performance, and security (FCAPS).

Conventional methods for improving communication reliability, quality of service (QoS) and security are based on the implementation of advanced network hardware. These methods have proven to be effective in smaller localised networks such as those supporting factories, power stations or campuses. However, in networks such as those supporting electricity distribution grids in cities, their scale and spread can make these approaches unfeasible. The extensive use of network hardware in massive networks not only raises development and maintenance costs but also increases the risk of reduced network reliability because each physical component added to the network topology introduces another potential point of failure. Furthermore, as these networks change and evolve to accommodate new grid stakeholders, the network

hardware utilities use will inevitably change over time which will result in networks consisting of heterogeneous devices. The potential of incompatibility between these devices will therefore also increase the risk of network failure and reduced network performance.

As an alternative to these hardware-based approaches, the Internet of Things (IoT) paradigm has provided new software-based communication frameworks such as Software Defined Networking (SDN) and Network Functions Virtualisation (NFV). These networking frameworks have proven advantages for large-scale networks and have the potential to overcome many of the challenges and limitations associated with conventional hardware-based networking. These benefits are achieved by decoupling the network control plane from network hardware and using virtualisation to provision crucial network functions on commodity computer hardware and devices in the field with virtual machines. The main differences between conventional hardware-based networking approaches and these software-based approaches are illustrated in Figure 1 and Figure 2.

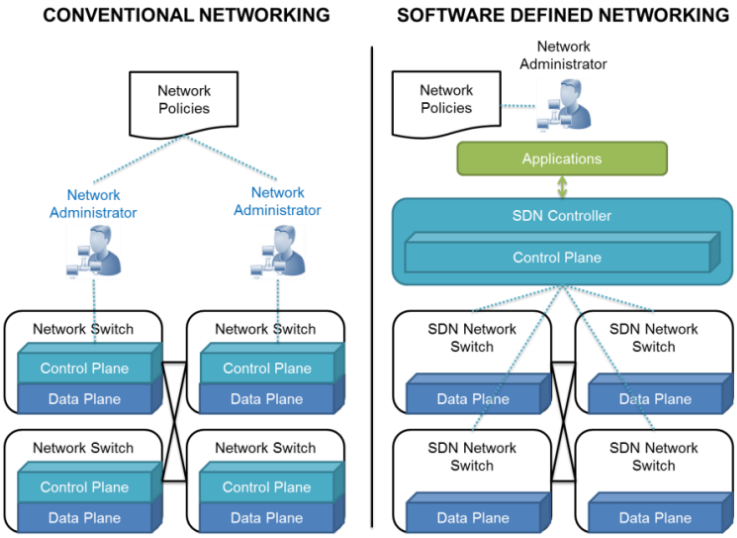


Figure 1: Difference between conventional networking and Software Defined Networking.

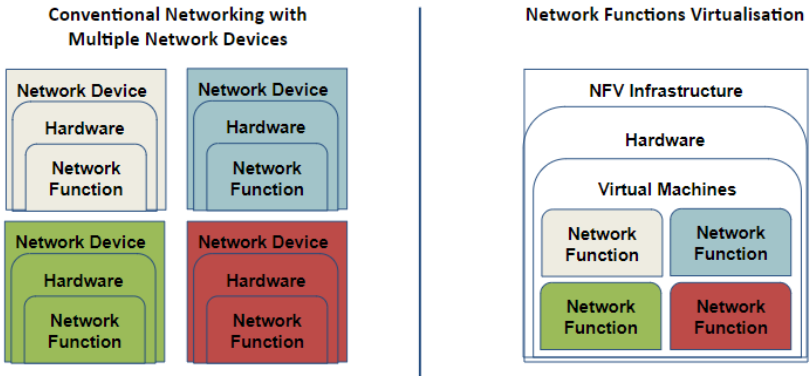


Figure 2: Difference between conventional networking and Network Functions Virtualisation.

In the information technology (IT) sector, local area network (LAN) implementations in enterprise networks [8] and data centres [9] have already benefited from software-based networking. Although deployment in metropolitan area networks (MANs) and wide area networks (WANs) still poses some design challenges, recent literature on the topic has shown that many of these challenges can be overcome with new design approaches [10]. The opportunities created by software-based networking have also been recognised by the telecommunication industry with SDN and NFV incorporated in the architecture for 5G cellular networks [11]. International standards organisations such as the International Telecommunications Union (ITU) [12] and the Institute of Electrical and Electronic Engineers (IEEE) [13] have started publishing standards related to software-based networking while the market value of software-based networking technologies has been realised by companies such as Cisco [14] and Nokia [15] who are incorporating SDN into their commercial products. With these successes in other industries new questions are posed for the application of these IoT frameworks in smart grids. One of these questions, that this dissertation will aim to answer, is whether these IoT-based network architectures can be implemented to improve power grid reliability in the electricity distribution domain.

1.1 Research Motivation

Smart grid covers several grid domains, each with their own set of communication requirements and challenges. The electricity distribution domain is considered one of the most important domains due to its interfaces with both electricity utilities and their customers. This domain is also very complex, as it includes a vast array of grid entities and interfaces. The networks that support the electricity distribution domain are likely to be the most vulnerable and are often the most difficult to manage and maintain. In the past, distribution grids were mainly responsible for distributing electricity to industrial, commercial, and residential customers located in cities, towns, and rural areas. This is changing rapidly as the world's energy landscape evolves. These days, the distribution grids in cities and metropolitan regions are likely to be the locations where smart grid adoption will occur first since governments are driving green policies with incentives to residents and businesses to reduce their carbon footprints [16].

As more consumers adopt distributed generation technologies such as rooftop photo-voltaic generation, energy storage and energy management systems, distribution grids must cater for more electricity producing consumers, or "prosumers". These prosumers will likely have equipment that requires reliable interfaces with the grid for both power and data exchange. Recent technology advances have enabled original equipment manufacturers to offer utilities a variety of new technologies that aim to support these new requirements, while improving the grid's efficiency and effectiveness with better grid knowledge. These technologies include smart sensors, smart meters, and smart substations that all depend on data communication. By implication, in addition to expanding and improving their electricity distribution infrastructure, utilities now need to consider the design and development of effective communication smart grid networks as part of their grid designs. Unfortunately, many utilities simply lack the knowledge, tools, and resources to do so.

1.1.1 Problem Definition

Because many smart grid systems support mission-critical grid functions, network faults can result in critical grid failures. This can have catastrophic consequences for utilities and their customers. Many governments have electricity supply regulations in place that specify a minimum amount of allowable forced (i.e., unplanned) supply interruptions for grid operators to comply with. In South Africa, this requirement is more than 99% availability per year for transmission and distribution grid circuits supplying established residential, commercial, and industrial customers [17]. Before undertaking expensive smart grid development projects to meet these regulatory requirements, utilities require the assurance that their grid communication networks will be able to support the systems they envisage to implement. Most of these systems depend on reliable communication and assurances that these networks can provide the required level of communication reliability that will limit the risk of system faults, are therefore also required.

Reliable communications are one of the problem areas identified for further research into communication architectures for smart grid [18]. For a network to be considered reliable it must ensure that each message reaches its destination correctly and timely, thus minimising the probability of system faults occurring in the grid. Some smart grid applications make use of messages that have critical timing requirements which means the communication network reliability should be evaluated under strict timing constraints. Communication problems can be reduced, but they can never be eliminated. Reliable smart grid communication networks must therefore also limit the impact communication faults have on the whole power system by restoring dysfunctional components to their normal working status in the shortest amount of time, thereby improving communication resilience. The communication reliability of a network should be analysed by defining quantitative reliability metrics such as error probabilities, failure probabilities and restoration delays. This will help utilities to predict the likelihood of network problems occurring and allow them to allocate network resources to prioritised functions and applications. The impact of communication problems should also be evaluated, and the reliability evaluation of a communication network should be mapped back to the power grid's ability to provide continuous electricity supply without disruption. Unfortunately, due to the associated unpredictability, cost, as well as the complexity associated with developing large-scale electricity distribution networks that rely on the use of advanced communication hardware infrastructure, it may be very difficult for network architects to provide the required assurances that their network designs will meet these requirements.

Software-based networking may offer a solution to this problem. Because these networks can be easily created for any network topology or network configuration, entire grid communication networks can be set up as software. Compared to hardware, this will not only reduce the time and cost related to network set up and evaluation, but it will also allow network architects to evaluate more design alternatives with simulated failure scenarios before they commence with full-scale implementations. These benefits and many others have been demonstrated in literature, but there remains plenty of opportunity for improvement and more

specialisation in this area. Recent studies on the application of software-based networking in smart grids have focused mainly on improving specific network functions or implementing combined network functions in attempts to meet the stringent communication requirements of specific smart grid systems [19]. Most of these studies seem to focus more on smart grid systems associated with novel technologies such as electric vehicles and microgrids. Studies that considered smart grids in the distribution domain, focused mainly on networks that aimed to support substation systems which rely on the transfer of small, time-sensitive messages, with little consideration for systems that make use of batch data transfers and big-data. Therefore, the opportunities to improve electricity distribution grids using software-based networking have not yet been fully investigated. Furthermore, many of the communication architectures proposed in these studies were evaluated using simple network topologies based on hypothetical grid network designs that could be set up in laboratory environments. Research that considers the implementation of these architectures on models based on actual distribution grids also requires further investigation. As [20] so rightly states: "For all stakeholders involved in the evolution of smart grid, the focus must be a well-defined path to migrate technologies from the four walls of the laboratory to actual field deployment".

Therefore, an overall problem that must be addressed is this lack of research in communication architectures for reliability focused, software-based communication networks that consider their implementation in real-world distribution grids. Complementary to these studies that focus on low latency communication using small messages, research that focuses on networks that can support big-data systems also needs further exploration. This is a design problem that not only requires a thorough understanding of the applicable technologies and standards, but also needs to consider the various aspects and constraints of real-world smart distribution grid implementations. This is required because the communication reliability of a distribution grid will have a direct impact on a utility's ability to supply electricity reliably. By attempting to solve this problem, this research must focus on the development of a design platform that will streamline and improve the development of communication networks for real distribution grids using software. It must also consider various methods that can be used to improve the communication reliability of a network using real-time monitoring and control techniques. Lastly, this research must consider a means for network architects and grid designers to evaluate various network design alternatives with minimal effort. These evaluations need to be performed in response to changing smart grid environments with consideration of the impact that communication issues can have on both the communication goals and grid objectives. Positive outcomes achieved with a solution to this problem may support further adoption of software-based networking in smart grid, which also may also lead to an increase in the overall uptake of smart grid by utilities around the world.

1.1.2 Research Questions

The main research question investigated in this dissertation is whether software-based networking can be used to improve the reliability of smart distribution grids. Following from this question, this dissertation will also aim to answer the following questions:

- Considering available literature, what are the proven benefits of network architectures based on the IoT frameworks for SDN, NFV and Machine-to-Machine (M2M) communication for smart grids, and how have these architectures been implemented and evaluated?
- Considering a model of a real city distribution grid, what would the key requirements and design considerations be for a software-based communication network that aims to improve the reliability of this grid?
- Using a design platform hosted on a desktop or laptop computer, can software-based networking principles be used to improve the set up and evaluation of communication network designs for this distribution grid, and how long will it typically take to set up such a network?
- Will the implemented software-based communication network be able to support and enable the automated batch data acquisition functions required by an Enterprise Asset Management (EAM) system?
- Can network functions and services implemented in this network reduce the probability of communication failures as well as communication delays, and can these functions be automated to reduce the overall network repair time?

1.2 Dissertation Objectives

The main objective of this work is to design and implement a communication network based on a model of a real city distribution grid, using a platform design, based on SDN, NFV and M2M, with the purpose of improving the grid's reliability. Because most smart grid systems depend on reliable data exchange, this work will aim to improve the overall reliability of an integrated smart grid, by first focusing on improving the communication reliability of a distribution grid. Secondly, this work will focus on improving grid reliability by enabling a reliability centred smart grid system, i.e., an EAM system that relies on the transfer of big data. The implemented communication network will therefore be evaluated in terms of its ability to recover from various communication problems while it supports uninterrupted operation of data acquisition functions. A model of a real city distribution grid will be used as a reference for this communication network design. An important objective of this work will be to implement and evaluate this network design in a virtual environment hosted on a single desktop computer. This will demonstrate how software-based networking can simplify and streamline the development of distribution grid communication networks. Overall, this work aims to add to the existing body of knowledge that supports the use of smart grid technology and the adoption of software-based networking in smart grid designs.

This dissertation will review current literature on the use of SDN, NFV and M2M in smart grid communication architectures and analyse the methods used to evaluate these architectures. This knowledge will be applied in the establishment of requirements for a reliability focused, software-based communication network for a distribution grid and to define a communication architecture for such a network. Using the proposed architecture, a communication network design will be proposed for a section of the City of Cape Town's distribution grid. This design will then be implemented in a testbed that uses virtualisation and network emulation to enable the creation and evaluation of software-based networks on a single desktop or laptop computer. Computer scripts will be developed to simulate the generation of data by substation devices connected to the communication network, while data aggregators will be implemented to facilitate the transfer of these data sets to a central data repository. The implemented network will then be evaluated in terms of its capability to support EAM acquisition functions and its ability to automatically recover from simulated communication problems. The ease of setting up these software-based networks will also be evaluated.

1.3 Research Contribution

As smart grid is a broad field of research covering multiple disciplines, this research may have applications in different parts of the energy, telecommunications, and IT industries. For utilities, this work may create new opportunities to pursue smart grid implementations that are more economically viable or to improve their existing implementations to be more reliable and resilient to failure. This research may also create new market opportunities for technology developers, application developers and network operators, allowing them to provide new products and service offerings for smart grids and other smart- industries. Overall, this research can contribute to the development of technologies that can have a positive impact on the environment by promoting energy efficiency and the use of more renewable energy resources.

1.4 Research Outputs

The following publications were developed during the registered period of master's work:

- G. Brown, N. Ventura, and J. Mwangama, "IoT Based Smart Grid Communication for Metropolitan Electricity Distribution Networks," in 2019 Southern Africa Telecommunication Networks and Applications Conference (SATNAC 2019), South Africa, Ballito, 1-4 September 2019.
- G. Brown, N. Ventura, and J. Mwangama, "A Software Defined Approach for Improving Resilience in Smart Distribution Grids," in 2020 International SAUPEC/RobMech/PRASA Conference, South Africa, Cape Town, 29-31 January 2020.

1.5 Scope and Limitations

Power grids consist of massive networks that interface with a wide variety of complex systems performing many different functions. Research into the smart grid paradigm can therefore cover a very broad list of topics, which is why the scope of this work must be limited. The focus of this dissertation will be on elements that form part of the communication layer of the smart grid architecture. The communication layer describes the protocols and mechanisms that enable information exchange between smart grid components. Examples of elements in this layer include descriptions of communication mediums and devices, network topologies as well as the communication standards and the associated protocols they use. As this is an important integration layer, elements that form part of the other layers of the smart grid architecture will however also be discussed briefly. The literature reviewed in this dissertation will also consider a broader view of software-based networking in smart grids. This will not only provide the reader with more context for this research and a better understanding of the solutions proposed in this platform, but also provide the basis that may support further work on the use of the proposed design platform and communication architecture.

The scope is also limited to networks in the smart grid distribution domain that fall within a single administrative domain and are controlled by a single operator i.e., an electricity utility. More specifically the design is limited to the neighbourhood area networks (NANs), field area networks (FANs), extended area networks (EANs), metropolitan area networks (MANs) and wide area networks (WANs) that support smart grid systems in distribution grids. Other networks that interface with these distribution domain networks will also be mentioned briefly. Figure 3 illustrates the electricity and communication flows corresponding to the distribution domain, as well as its interfaces with other domains.

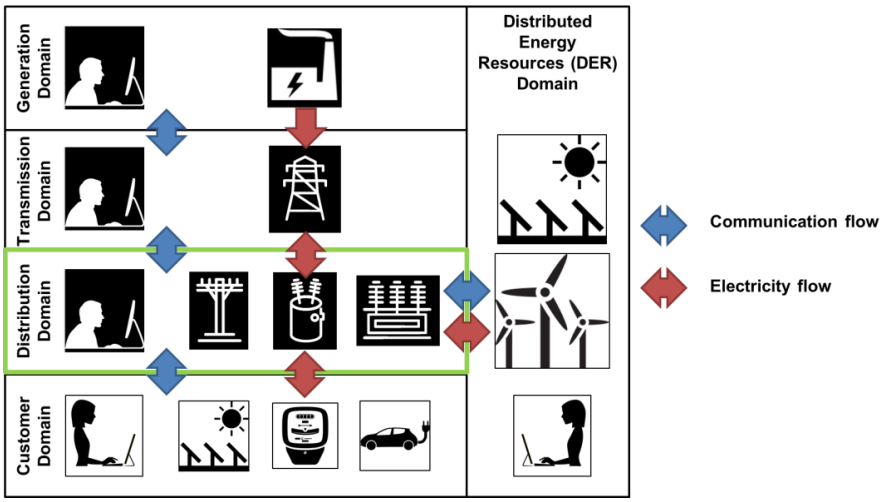


Figure 3: Electricity flows, communication flows and interfaces of the distribution domain.

Several important communication requirements and design considerations for smart grid communication networks will be highlighted and discussed briefly in this dissertation. Communication reliability and communication latency will, be the focus of this research, as both these requirements form the foundation for

a reliable smart grid design. The scope will also be limited to the implementation of only one reliability centred smart grid system that relies on big-data transfers, i.e., an EAM system. A few other smart grid systems that may benefit from the software-based networking approaches discussed in this research dissertation will however be mentioned briefly. Furthermore, it is widely acknowledged that real smart grid implementations will likely consist of various heterogeneous systems and components. For this implementation, an assumption will be made that all devices and applications in the network use the same communication protocols. Network functions offered through SDN, NFV and M2M to address this heterogeneity challenge will only be discussed briefly as a basis for future work. Since the city grid chosen for this design has not undergone any smart grid development yet, some assumptions will also be made about the availability network and grid infrastructure.

The proposed design will consider a process of developing a software-based network on a single computer and then deploying the design on a distributed network of computers in the grid. The city distribution grid communication network proposed in this work, will however only be implemented on a design platform hosted on a single computer for evaluation. This decision was made because this research aims to study and emphasise the simplification that software-based networking brings to the network design and evaluation process. In addition, because the deployment of the virtual machines in distributed platforms have already been successfully demonstrated there is no need to do so again, as this work aims to build on the successes of other related research. The limited number of resources that will be available on the computer used for this implementation and evaluation, will however limit the scale of the network topology that can be created in the test environment. By implication, the size of the city distribution grid that will be used as a model for this design and implementation will be limited to a specific grid section. Furthermore, in this grid section, only a limited-number of substation devices will be simulated as this will also be limited by resource availability on the computer. Approaches for expanding this platform to implement a larger distribution grid communication network will be discussed briefly as possible future work.

1.6 Dissertation Outline

The remainder of this research dissertation is structured as follows:

Chapter 2 presents the background and a review of relevant literature related to this research. Firstly, a brief overview of the smart grid architecture model and communication networks that support smart distribution grids is presented, along with an overview of the SDN, NFV and M2M architectures to provide background information and context to the work. Secondly, this chapter provides an in-depth literature review of related research that also focused on smart grid communication architectures based on SDN, NFV and M2M.

Chapter 3 provides an overview of the key requirements and design considerations for a communication network that aims to improve the reliability of a distribution grid. Then a proposal for a software-based communication network architecture for distribution grids is presented, along with a proposal for an implementation process that this architecture can enable. Finally, a communication network design for a section of the Cape Town distribution grid based on this architecture will be presented.

Chapter 4 covers the design implementation and provides a description of the implementation platform. A description of each of the developed computer scripts and open-source software applications that were implemented in this platform is provided, along with a description of the evaluation tools. A description of a test plan that was used to evaluate the design is also presented.

In Chapter 5 the test results obtained from an evaluation of the implemented communication network design are presented. During this evaluation, emphasis was placed on the ability of the network to support both the communication objectives and functional objectives of an EAM system that included uninterrupted, automated data acquisition while the network was subjected to various simulated communication problems. An analysis of the network set up and preparation process, using the proposed design platform is also presented in this chapter.

Chapter 6 summarises this work and presents the conclusions that can be drawn from it. This chapter also discusses the contributions of the research as well as recommendations for future work on the topic.

Chapter 2

Background and Literature Review

The previous chapter introduced the basic concepts and aims of smart grid, as well as some of the principles of smart grid communication. In particular, the current challenges utilities face with designing and developing communication networks for reliable smart distribution grids were explored. This chapter provides background on the smart grid architecture model and the communication networks that support smart distribution grids. It also provides details on the IoT frameworks for SDN, NFV and M2M as necessary background for the literature reviewed in this chapter as well as the designs proposed in later chapters. Finally, a review of literature that focused on solutions that aimed to improve smart grid communication with software-based networking principles is presented. This review will highlight the similarities and differences between other relevant work and the work proposed in this research dissertation.

2.1 Smart Grid

To describe a smart grid, the Smart Grid Architecture Model (SGAM) is commonly referenced in related research [21]. The SGAM provides a detailed description of the elements that make up a smart grid, presented as a three-dimensional reference model. The model consists of five grid domains on the X-axis, six grid zones on the Z-axis and five interoperability layers on the Y-axis as shown in Figure 4.

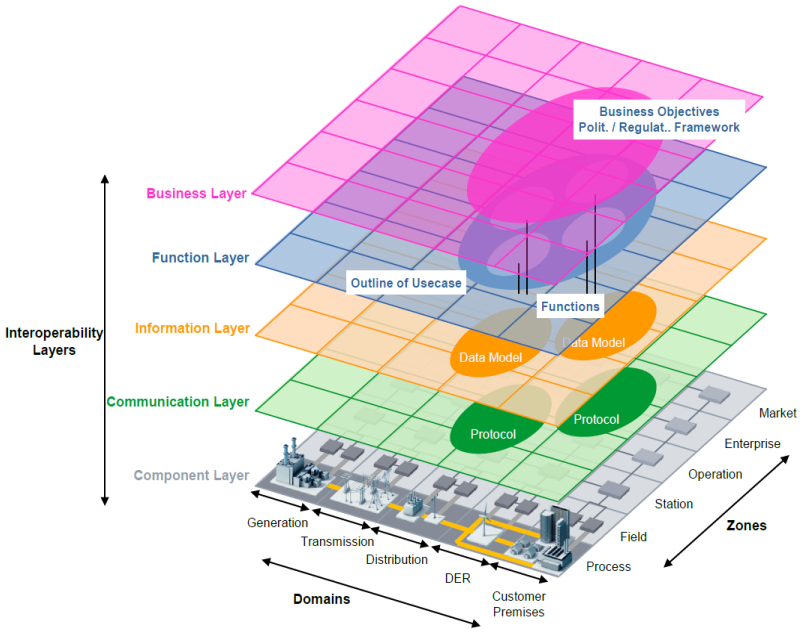


Figure 4: SGAM Framework [21].

The SGAM's uppermost business interoperability layer defines the smart grid's objectives and policies. These objectives are realised through grid functions and use cases defined in the function layer. Grid functions that depend on data and data transfer will require data models for data management, processing,

and storage as well as communication standards and protocols. These are defined in the information and communication layers, respectively. The component layer represents the grid technology that generates and utilises these data sets to support electricity generation, transmission, distribution, and consumption. Because a smart grid is a fully integrated system requiring interfaces across multiple domains and zones, each of the five interoperability layers requires consideration when designing end-to-end smart grid systems. The networks in the communication layer form the crucial links between various smart grid components and applications that exist in the various grid domains and zones. Besides the component layer that is usually already established in most existing power grids, the communication layer of a smart grid is usually the most complex and capital intensive to develop, especially in grids where legacy communication infrastructure already exists.

In the distribution domain, which is the focus of this work, hundreds of grid devices can interface with these networks. Data generated by these devices enable utilities to monitor distribution infrastructure and to control specific grid components remotely using smart grid applications. To facilitate efficient, reliable, and secure data exchange between these devices, large-scale communication networks that can potentially consist of hundreds of network devices are needed. The scale, complexity and importance of these networks have made communication one of the focus areas in research and development of next generation power grids.

2.1.1 Smart Grid Communication

Various organisations including the IEEE [22], the International Electrotechnical Commission (IEC) [23] and the National Institute for Standards and Technology (NIST) [24], have worked towards standardising smart grid communication for interoperability across all grid domains. Figure 5 shows the IEEE's communication technology interoperability architecture perspective for a smart grid that consists of seven grid domains, 23 grid entities and multiple interfaces. Figure 6 illustrates an end-to-end communications model for smart grids that describes these networks in terms of four key domains. Because smart grids can span across entire countries and continents, they will likely make use of wide area networks (WANs) to connect infrastructure in different domains. These WANs also serve as the interfaces with utility's local area networks (LANs), core networks and regional and metropolitan area networks (MANs). The WAN usually includes a high-bandwidth backbone network that handles long-distance data transmission with very high reliability. Electricity networks that support the distribution domain can consist of a combination of NANs in towns or cities, as well as EANs and FANs in rural areas. These networks interface with the WANs using backhaul networks while grid devices access these networks using either wired or wireless communication through a series of strategically located network "hot spots". The utility's networks can also interface with customer networks to monitor and control customer owned electrical devices. Smart meters may communicate either through the distribution grid communication networks, customer networks or dedicated advanced metering infrastructure (AMI) networks depending on the utility's communication policies.

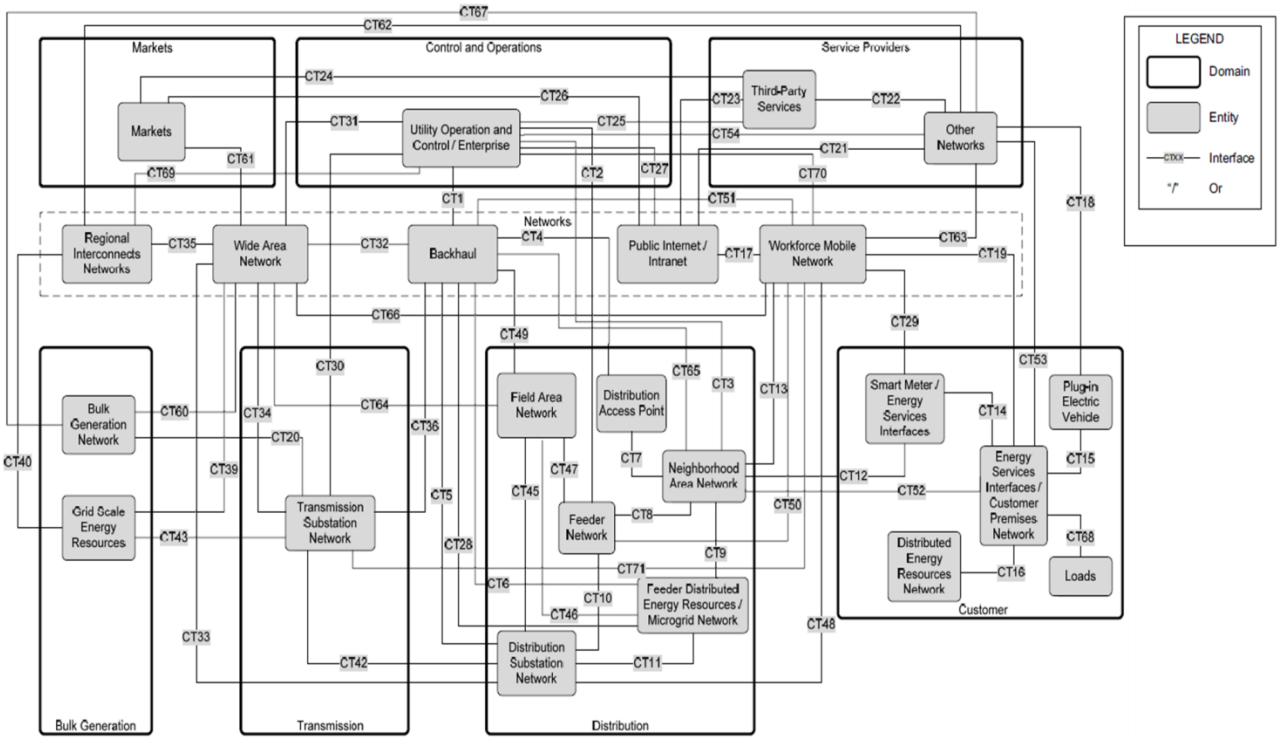


Figure 5: Communication Technology Interoperability Architectural Perspective (CT-IAP) [22].

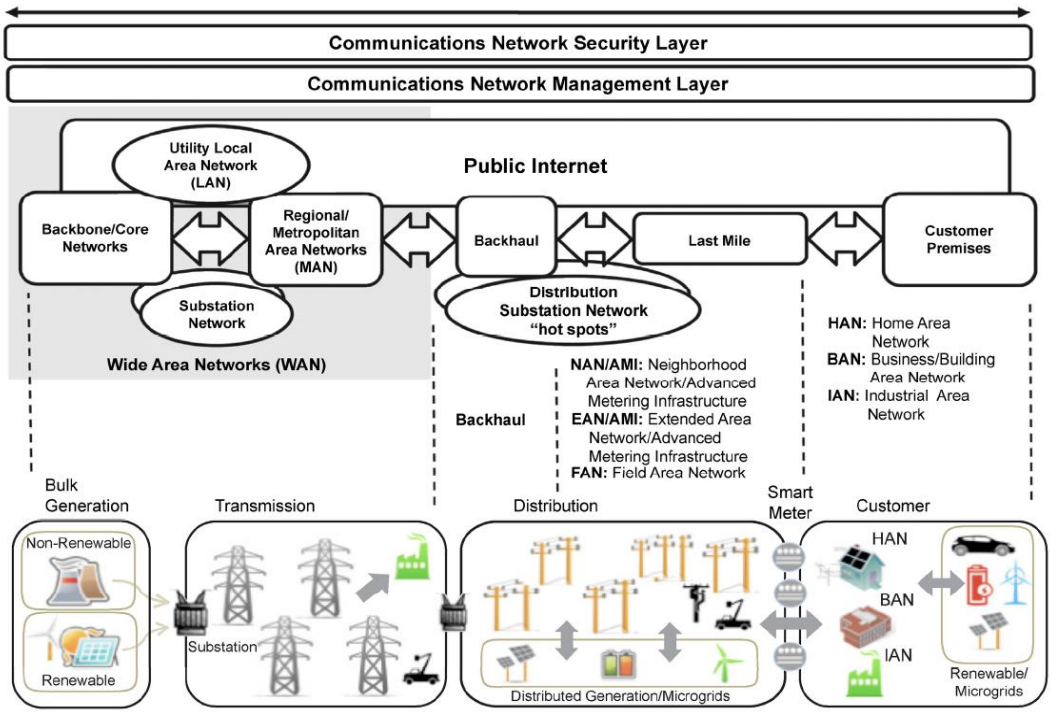


Figure 6: End-to-End Smart Grid Communications Model [22].

2.1.2 Communication Networks in Distribution Grids

The component layer of a typical electricity distribution grid usually consists of transformers, switchgear, distribution panels and other components used to distribute electricity. Connected by power cables or overhead power lines, most of these components are in protected distribution substation buildings while

others are free-standing in protected enclosures. Each distribution grid will have its own unique network footprint based on the locations of this distribution equipment and the grid's customers. Some distribution grids have densely clustered footprints with many supply points for large groups of customers in cities, while others have widespread footprints that supply only a few customers in rural areas. Electricity distribution grids tend to follow a hierarchical grid topology consisting of main substations at the top, followed by primary, secondary, and mini substations in the middle tiers. Low voltage distribution panels and other distribution equipment appear at the bottom of the hierarchy. To improve supply reliability, these grids are designed with multiple alternative supply links at important points in the distribution grid network, usually forming radial and ring supply networks [25]. Figure 7 shows an example of an electricity distribution grid layout in a section of the City of Cape Town's city bowl with densely clustered customers.

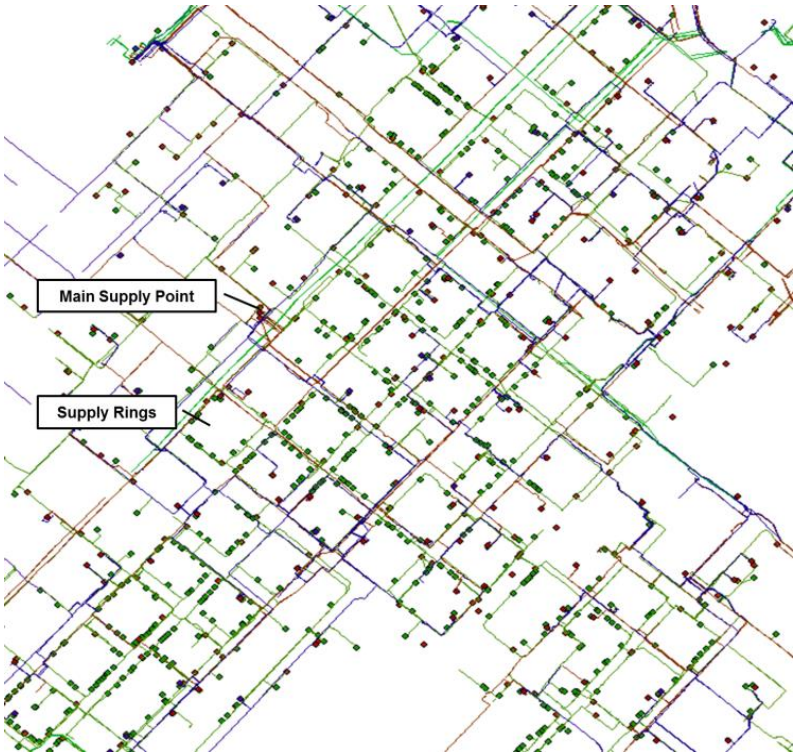


Figure 7: Example of an electricity distribution grid section layout in the City of Cape Town.

In this figure, each dot represents a node in the grid that can be a substation, distribution panel or customer connection point. The different coloured lines indicate power cables or power lines connecting these nodes. Note the supply rings that form around each city block and their radial connections via main power cables and lines running in parallel to primary supply points in the area. To monitor, control and automate the protection of these distribution grids, industrial management systems that rely on communication have been included in their design for many years. A common example of one of these systems is a substation supervisory, control and data acquisition (SCADA) system. Industrial networks that support SCADA systems usually mirror the topologies of the grids they support, and favour dedicated wired communication due to its reliability for mission critical applications. Figure 8 shows an example of a typical network configuration for a conventional SCADA system in a power grid.

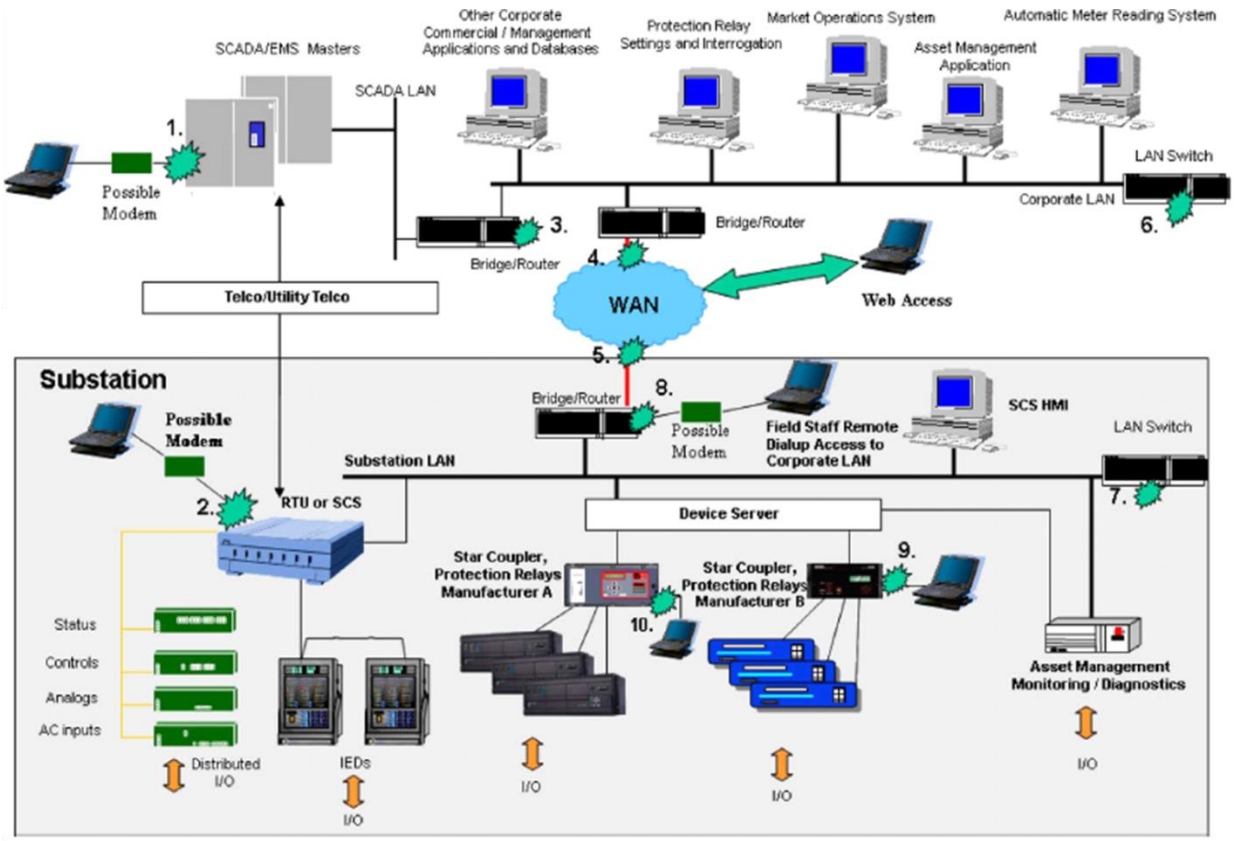


Figure 8: Example of a conventional substation SCADA network [5].

A typical SCADA topology will consist of substation networks that interface with various substation devices. These substation devices can include intelligent electronic devices (IEDs), remote terminal units (RTUs), substation control systems (SCS) and devices for asset management. Processed data from these devices can be accessed locally via local computers or these data may be transferred to servers connected to upstream networks. Substation network switches and routers route traffic between these computers and servers. Some substation SCADA networks may include additional network interfaces to utility networks or the internet.

Dedicated wired networks have been favoured for substation SCADA communication networks for many years due to their higher reliability and ability to better ensure communication service quality. With recent improvements in QoS and bandwidth offered by modern wireless communication technologies such as Wi-Fi, cellular and low-powered wide area networks (LPWAN), more utilities are now considering wireless communication in their grid network designs [26]. These wireless networks are much more flexible to develop and reconfigure and they also allow utilities to set up wireless sensor networks in their grids [27]. Most modern electricity distribution grid communication networks will therefore usually include hybrid networks that combine the benefits of both wireless and wired networks. Fibre networks are usually preferred for backhaul and backbone networks [28]. An example of a distribution grid communication network that combines different communication mediums in a mesh topology is shown in Figure 9.

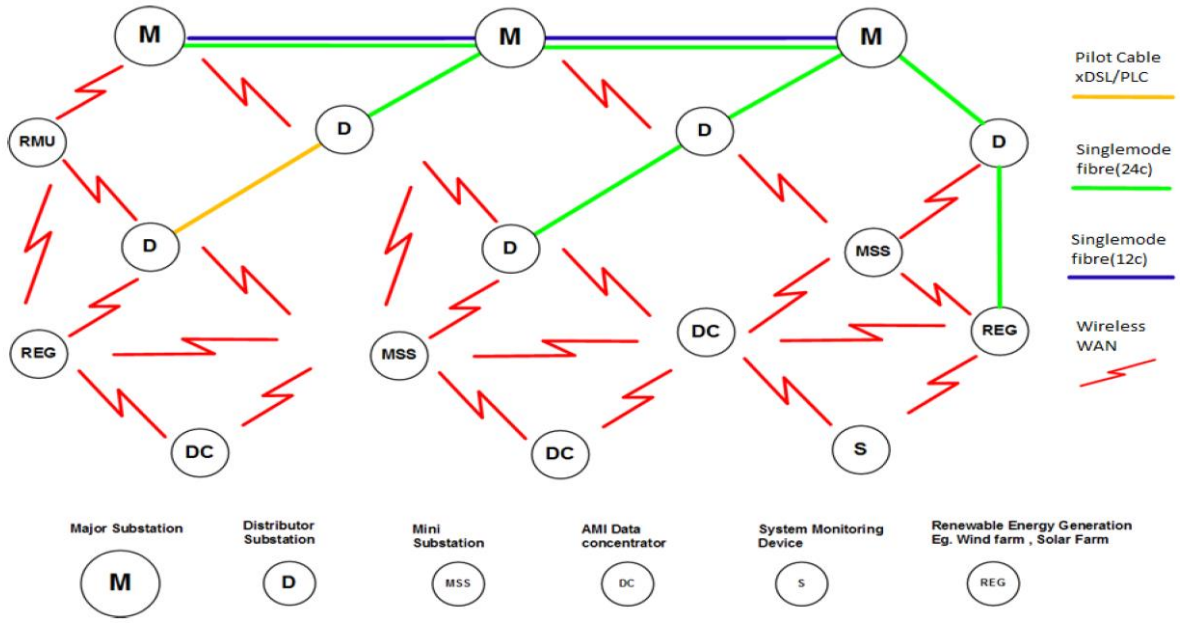


Figure 9: Example of a wireless mesh configuration for an electricity distribution grid [28].

Improvements and expansion of SCADA networks have made network monitoring and control systems crucial for managing these communication networks. By monitoring the network configuration and communication flows, network administrators can identify faults, security threats and other anomalies that threaten grid communication, allowing them to correct these issues as quickly as possible. Security has become another major concern for these networks necessitating solutions that allow administrators to implement network security policies that will prevent unauthorised access, misuse, modification, or denial of use in grid communication networks and network-accessible resources. In smaller networks these functions may be performed manually with direct configuration on hardware. However, for larger, more complex networks, utilities may choose to implement network and security management applications such as SolarWinds [29] that offer dashboard views of the network using standardised models that support FCAPS.

In recent years new systems and technologies have emerged that improve and expand the functions that were originally performed by these SCADA systems in distribution grids. As these systems become more advanced, so do their communication requirements. Conventionally, most network designs start off as models and simulations. These modelled designs then get implemented directly as physical network installations using a variety of network hardware components. Unfortunately, since most of these modelled designs are often based on assumptions, the actual networks do not always deliver the simulated results. An example of one of these assumptions is the performance of the network switches that can vary depending on the manufacturers' specifications, technology, and software. The result of this unpredictability in how a network will behave is the frequent redesign, reconfiguration, and redeployment of the network until all unforeseen issues are resolved. These tasks can become very time-consuming and capital intensive as the scale of a network increases. Some systems rely on data exchange with devices in lower tier grid components and even customer networks which drastically increase the complexity of the network designs. This

challenge can make the cost of developing and maintaining large-scale distribution grid communication networks in densely populated metropolitan areas an unfeasible option and often limits implementations to systems that only consider upper tier grid elements such as main and primary substations. These limitations might explain why few utilities have pursued full scale smart grid implementations in their distribution grids.

2.2 Networking Based on IoT

Recent advances in telecommunications and information technology have led to new innovations characterised as IoT [30]. These advances include major improvements in the capabilities of telecommunications technology as well devices that make use of sensors and actuators to observe and interact with the physical world. Large reductions in the cost to manufacture these devices have made them abundant in everyday life. IoT has also had a major impact in industry with many new affordable industrial IoT (IIoT) devices entering the market that provide better monitoring, control and automation functions to plants and systems. In distribution grids, these IIoT devices offer the ability to expand monitoring, control, and automation capabilities to more grid assets. These days, almost any component in the distribution grid can be fitted with a smart device, which means that each node in the distribution grid can potentially become a node in the grid's communication network. This new capability offered by IIoT is one of the cornerstones of most smart grid solutions and it has major implications for grid communication networks.

The International Telecommunications Union's (ITU) IoT reference model shown in Figure 10 describes the typical components of IoT implementations, consisting of an application layer, a service and application support layer, a network layer, and a device layer [31]. This architecture model also has cross-cutting capabilities for management and security capabilities.

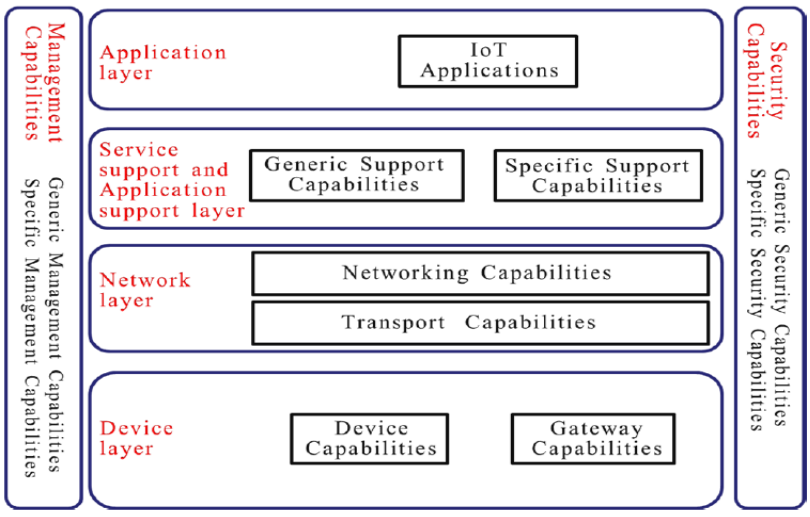


Figure 10: ITU-T IoT Reference Model [31].

This IoT reference model and its various derivations in literature form the basis of many of the solutions that use software applications and devices connected to networks to solve problems. With its development,

the IoT paradigm has provided a series of frameworks that aims to improve the exchange of data in massive communication networks. One of these frameworks, known as cloud services, is an improvement on conventional client-server-based networking that offers services such as computing and storage in shareable decentralised platforms. Cloud services have contributed towards alleviating some of the problems associated with resource allocation, utilisation, and management in IoT networks, but faces challenges in terms of its manageability, flexibility, dependability, and security [32]. To accommodate vast and dynamic smart grid environments, cloud computing systems need to become more flexible and adaptable. This flexibility can be achieved with better service abstraction offered by another set of IoT frameworks known as Software Defined Systems (SDSys). SDSys are systems that have added software components that provide this abstraction from physical hardware and other layers, thereby offering opportunities for system administrators to construct and manage their systems through flexible software layers more easily. SDN is a popular example of this abstraction in networking environments.

2.2.1 Software Defined Networking (SDN)

Most conventional networks rely on hardware such as network switches to manage network traffic with pre-programmed control functions configured on the physical device according to network policies. These control functions execute in the switch's control plane, controlling the data flowing through the network switch's data plane. Network switches often have limited visibility of the entire network which can impact the overall network performance because any changes to the network configuration or network policies that require changes to these control functions will require network administrators to manually reconfigure each switch individually. In SDN the control functions run as applications in logically centralised SDN controllers. These SDN controllers provide network administrators with a global network view, as well as programmatic interfaces that provide control of the network's forwarding devices using SDN applications. SDN therefore decouples the control plane from the data plane, allowing network switches to become simple traffic forwarding devices without the control plane installed on the device itself. It also allows these switches and controllers to run on normal computer hardware. Figure 11 provides an overview of the SDN architecture, as laid out in a recommendation from the ITU.

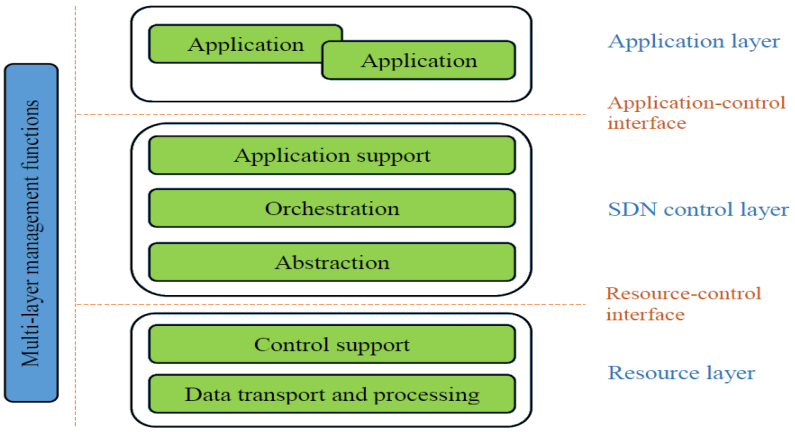


Figure 11: SDN architecture recommendation [33].

In this architecture the application layer defines the network services and service-aware behaviour of the network resources. The control layer provides the means of flexibly controlling this behaviour using SDN controllers; and control instructions are defined in the application layer. The resource layer is where network elements perform the transport and processing of data packets as prescribed by the control layer. These layers are supported by multi-layer management functions that include functionalities for supporting FCAPS. Interfaces between the layers are handled by application programming interfaces (APIs). The southbound interfaces handle the resource-control interfacing while northbound interfaces provide the application control. APIs based on representational state transfer (REST) are commonly used for northbound interfaces because this architecture ignores the details of component implementation and protocol syntax and rather focuses on the roles of components, their constraints on their interaction with other components and the interpretation of significant data elements. Because SDN is not vendor specific, it can accommodate a variety of networks and devices, with vendors such as Cisco already incorporating open SDN standards in their technology [34].

Several SDN standards have emerged that include IEEE P1520 [35], OpenFlow [36], ForCES [37] and SoftRouter [38] of which OpenFlow has become one of the most popular open source standards. OpenFlow consists of controllers, communicating with OpenFlow switches [39]. Each SDN switch has a group table and flow tables that contain flow entries as illustrated in Figure 12.

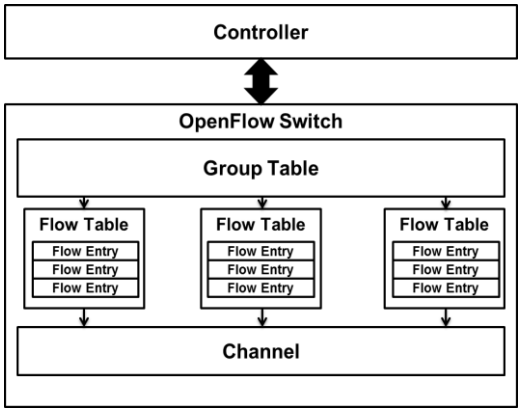


Figure 12: OpenFlow model adapted from [39].

OpenFlow switches are responsible for forwarding, dropping, or modifying packets in the network based on flow rules defined in flow entries. A switch can have multiple flow tables allowing it to differentiate between packets and the flows they should follow. Switches communicate with other switches and the controller via an available communication channel using the OpenFlow protocol. They also include a group table for more complex and specialised packet operations that can easily be defined in flow tables, such as load-balancing, multicast, and fast failover recovery. OpenVSwitch (OVS) [40] is an example of a popular open source OpenFlow switch that works as a virtual switch in virtualised environments. The SDN controller is software that can abstract device-specific details and provide common functionalities and essential

services to the SDN network. The controller can add, remove, or revise flow entries in OpenFlow switches based on user commands or automated application outputs. It is also responsible for receiving network information and forwarded packets from switches for which no flow entries exist. Appendix B lists some of the most common open-source SDN controllers along with their supported interfaces and performance characteristics for reference [41].

2.2.2 Network Functions Virtualisation (NFV)

Like the SDN control plane, NFV uses virtualisation to decouple network functions from hardware so they can run as software on virtual machines (VMs) [41]. These virtualised network functions (VNFs) can then be remotely and dynamically created, configured, migrated, and replicated. The use of VMs therefore eliminates the need for physical, on-site installation or configuration of network functions on hardware. Examples of network functions that can be virtualised include network translation (Gateways), network address translation (NAT), firewalling, intrusion detection, domain name services (DNS) and caching. Several VNFs can be combined to create so-called Virtual Network Function Chains (VNFCs), which provide more complex functions by combining sets of network functions interacting among each other. VNFs operate on underlying infrastructure known as network functions virtualisation infrastructure (NFVI) that includes the underlying hardware resources for computation, storage, and networking. A virtualisation layer realised by VMs or container virtualization uses these hardware resources to provide virtual computation, storage, and networking resources to the VNFs. To manage the life cycles of VNFs, their allocated resources and to collect and forward performance measurements, NFV requires Management and Orchestration (MANO). A NFV MANO consists of three major components, the NFV Orchestrator, the VNF Manager and a Virtualized Infrastructure Manager. Figure 13 illustrates a high-level NFV architecture.

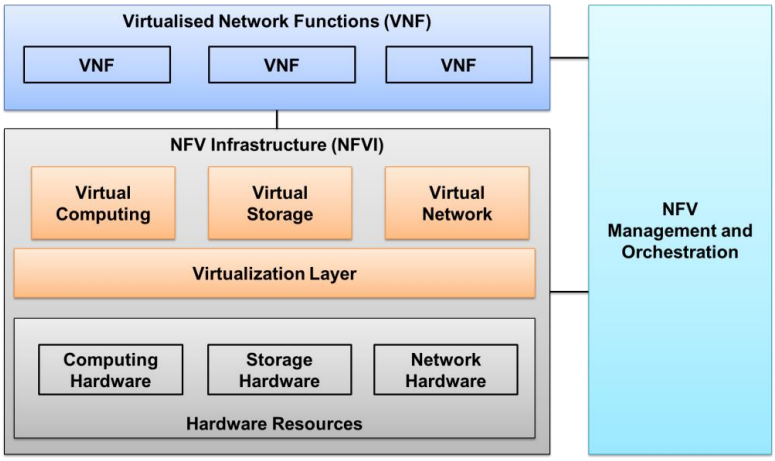


Figure 13: NFV reference architectural framework. Adapted from [42].

NFV complements SDN because both SDN and NFV enable the creation of communication networks that move network functionality from hardware to software. This allows network architects to use commodity computer and networking hardware instead of proprietary devices while taking advantage of the benefits of

application program interfaces (APIs). These software-based networks not only support more efficient network orchestration but also the automation of network services. Both approaches are mutually beneficial but are not dependent on one another.

2.3 Software-based Networking in Smart Grids

Inspired by successes in the telecommunications and information technology sectors, recent studies have focused on the problem of developing IoT based communication network architectures that aim to meet the stringent requirements of various smart grid implementations. As a starting point, the research survey performed by [19] provides a good overview of the current knowledge on SDN based smart grid communication (SDN-SG). A noteworthy advantage of SDN mentioned in this survey, that is of particular interest in this work, is its run-time configurability. This feature allows network architects and administrators to update the control plane of a live network quickly and effortlessly in response to changes with almost no network interruption. This feature may prove to be very beneficial during the design evaluations and when urgent changes to a communication network are required in response to changes in the distribution grid environment. This survey listed nine other potential advantages that have been identified for SDN-SG, most of which deal directly or indirectly with grid reliability and resilience. Many of these studies reviewed in this survey focused on improving communication reliability and quality of service using software-based networking principles and have shown success with this approach. However, most of this research focused on meeting specific network service level requirements and therefore only considered implementations and evaluations of their solutions in hypothetical grid designs. Only a few studies have taken into consideration how these software-based networking solutions may be implemented in real power grids and how these solutions may be scaled into vastly distributed communication networks such as those used in the electricity distribution domain. As a complementary approach to SDN, the benefits of NFV and M2M in smart grid communication network designs have also been highlighted in some of the studies reviewed by these authors.

Communication reliability seems to be a focus point in many smart grid communication studies, with network fault tolerance, service guarantees and communication recovery time receiving a lot of attention [43]-[45]. A common theme emerging in literature is that smart grid communication reliability can be broken down into three separate problems areas. These problem areas are: Reducing the probability of communication failures, reducing probability of communication delays, and reducing the mean time to repair communication problems. Research that aimed to solve these problems using SDN, NFV and M2M principles will now be reviewed in more detail with a specific focus on the contributions these studies made to this work. The focus will be on the network functions that studies implemented to improve communication reliability and the implementation methods they considered. This section presents an analysis of this research with the purpose of providing an overview of current knowledge, relevant theories, methods, and gaps in the existing research as it relates to the research problem that this address.

2.3.1 Machine-to-machine Communication (M2M)

M2M combines technologies that automate the communication between machines such as computers and smart devices, thus optimising the processes these machines support [46]. Central to this approach is the standardisation of the communication interfaces that can exist between these machines in a network, which is achieved by the introduction of a common middleware layer in the communication network architecture. This middleware layer supports standardised data models, encoding and serialisation of data for exchange between machines and is an improvement over conventional machine communication networks that are usually point-to-point, meaning they focus on the interactions of specific machines and their interactions with specific applications. This point-to-point approach often results in vertical applications with fixed associations where information is passed between devices and applications with the middleware not being aware of what is being communicated. This can lead to the formation of communication silos in networks that are difficult to maintain, support and update.

The M2M middleware shifts the focus from vertical applications to horizontal applications that consider communication among all participants in the network. If the middleware is aware of all the machines and applications in the network, it can allow applications to interact on a level that considers each with its own semantic descriptions [47]. The difference between these vertical and horizontal architectures is illustrated in Figure 14.

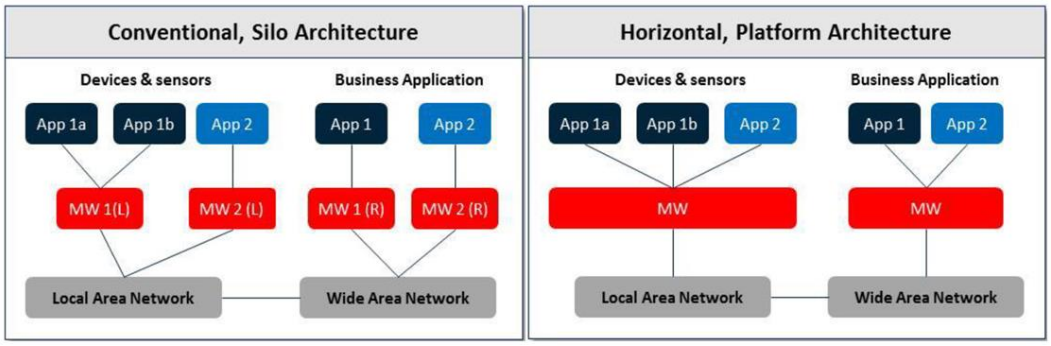


Figure 14: Conventional Silo Architecture vs. Horizontal Platform Architecture used in M2M [47].

ETSI published their first M2M standards in 2011. They focused on horizontal service platforms and related APIs that aimed to improve and maintain globally applicable, access independent technical specifications with an initial focus on the Service Layer [48]. The global OneM2M organisation followed with the release of a series of standard M2M specifications in 2014 [49]. These M2M standards complement a software-based networking approach using middleware to provide common services that support communication in networks that interconnect various devices and applications. Examples of these services include communication management, resource handling and data management. Devices and applications can utilise these services by subscription, which means that network designs must only consider publication of

the services in a standard format that is recognisable, thus eliminating the need for complex point-to-point integration each time a new machine is added to the network.

The release of M2M standards have led to the development of platforms that implement these standards for use in research and industrial applications. An example of one of these platforms is the open machine type communications (openMTC) initiative, initiated by the Fraunhofer Institute for Open Communications and the Next Generation Networks Chair [50]. This platform offers a reference implementation of the oneM2M standard, using a horizontal service approach that is independent of underlying hardware and network infrastructure. A high-level architecture of an M2M network based on the openMTC platform is shown in Figure 15. ETSI also emphasises M2M as an important research focus area for smart grids and identified various use cases for using M2M in smart grid systems [51].

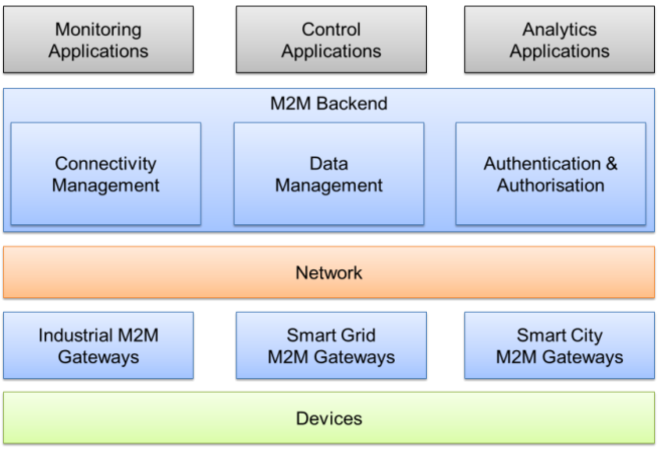


Figure 15: Architecture for an M2M platform based on oneM2M standards. Adapted from [50].

2.3.2 Reducing the Probability of Communication Failure

One of the key characteristics of a smart grid is its resilience to failures and malfunctions. Most modern power grids are protected by systems that can detect and rapidly respond to electrical faults with minimal disruption, while preventing these faults from propagating through the grid. As the systems that perform these protection functions become more dependent on co-ordination through communication, similar methods that will reduce the probability of communication failures that can lead to situations where the grid becomes unobservable and uncontrollable, need to be considered. A network's message success rate is a useful metric for determining how often communication failures occur in a network. The message success rate (r) can be expressed as the ratio of successful messages received (n_r) to total messages transmitted (n_s) over a period as shown in the following equation:

$$r = \frac{n_r}{n_s} \qquad (1)$$

By measuring the message success rate of a network retrospectively, the probability of communication failure in a network can be estimated and presented as a probability of future success. For most smart grid applications, the message success rate must never fall below 99 %, although some mission critical applications require message success rates higher than 99.9999 %. To reduce the probability of communication failure in smart grids, three approaches have been identified in recent literature. These approaches focus on solutions that automate the communication recovery after network links and nodes have failed, reduce the number of vulnerable points of potential failure in the network and provide critical functions closer in the network to the sources that utilise them.

Automated recovery from link and node failures

Most network failures occur because either a network link or a device acting as a node in the network fails. Although these failures cannot be totally avoided, it is possible to recover from them by diverting messages onto back-up flow paths after such a failure occurs. By automating these failovers, the speed of recovery can be improved substantially, which has led to the development of fast failover recovery (FFR) functions for communication networks. Reference [52] implemented their own FFR algorithm for a smart grid application that used the programmatic application interfaces to SDN controllers to reactively divert network traffic to alternate flow paths within tens of milliseconds after a network failure occurred. Their solution was implemented and evaluated in a single machine platform that made use of virtualisation to simplify network setup and testing. Although this study mentioned the possibility of implementing this architecture in the distribution domain, their reactive FFR solution would likely not be optimal for large-scale communication networks as it was dependent on communication with the SDN controller to function. The FFR function's performance would therefore be impacted by the size of the network and the number of network hops a message had to travel to the controller.

The proactive FFR solution implemented by [53] in their platform called *SDN4SmartGrids* did not suffer from this limitation. Their FFR algorithm allowed the SDN switches to reroute traffic without involving the SDN controller, since the backup paths were predetermined and stored in the flow tables of the SDN switch. This not only improved failover time, but also made the solution scalable to larger networks without compromising performance. The main objective of their study was to design a communication network that could meet the requirements of an IEC 61850 compliant substation. Their platform therefore implemented load balancing and traffic prioritisation functions in addition to FFR. To evaluate the ability of their *SDN4SmartGrids* design to meet these requirements, it was implemented in a small laboratory LAN setup based on a hypothetical substation network configuration. The main contribution taken from this study was that it presented an example of a software-based networking platform that combined multiple network functions to improve communication reliability, although the solution was mainly SDN-based. However, the proactive FFR approach proposed by these authors has one potential limitation for distribution grid implementation: limited flexibility: Because the flow paths had to be pre-configured by the SDN controller, they could not be adjusted easily without manually updating parameters that updated flow entries. This

meant that if a network topology changed and a failover path was not available any longer, the switch would not be able to correct the problem by itself and human intervention would be required to correct the flow paths. Because distribution grids are prone to frequent changes, this could introduce more communication reliability risks. An improved FFR approach proposed by some of the same developers of *SDN4SmartGrids*, was to implement a hybrid FFR solution that combined the benefits of both the reactive and proactive FFR approaches [54]. This hybrid approach, that they called 'enhanced fast failover', was also developed to meet communication requirements for IEC 61850 compliant substations and it was able to achieve very fast failover response times. In addition, it was not constrained by network scalability or flexibility. The enhanced FFR solution was implemented in a small laboratory LAN network that was based on a hypothetical substation network configuration.

Hybrid FFR algorithms are becoming the norm with more research into its application in geographically distributed networks and carrier grade networks [55]. The current version of the OpenFlow protocol now also supports a fast failover group entry in its tables [56] to enable proactive FFR with accelerated fault recovery time. Because OpenFlow is an open-source protocol, various applications can be developed to utilise these group tables via application interfaces with SDN controllers to enhance its performance. There is currently an active research community that contributes to its continual improvement through the development of open-source applications. The ONOS controller is an example of an SDN controller that ships with a hybrid FFR as a service application [57]. One of the big challenges of implementing network functions such as FFR in large-scale geographically distributed networks, is that its success depends on the availability of alternative flow paths in the network. Providing these back-up paths requires the introduction of additional network nodes and links. Implementing these flow paths in large-scale communication networks with network hardware is resource intensive and introduces additional points of potential failure in the network. If improved reliability is desired, methods need to be considered that can limit the amount of potential failure points introduced in a network. NFV may offer a means to achieve this.

Reducing potential points of network failure

When faced with the problem of many vulnerable and unreliable nodes in a typical AMI network, [58] considered the use of NFV to create and deploy VNFs with VMs dynamically at any point in a network as software. They hypothesised that by creating these network nodes as software, network development and deployment could not only be simplified, but the reliability of the network could also be improved due to a reduction in a dependence on vulnerable hardware. To investigate this, the authors used mathematical models to evaluate this using a design for an NFV-based AMI network. Their results showed a potential improvement in overall communication network reliability of 82.32 % on average. In addition, a cost model based on this architecture indicated that a potential decrease of financial requirements of 49.72 % was achievable if hardware components are replaced with VNFs. This study also highlighted the capabilities of NFV to reduce the mean time to repair (MTTR) an AMI network, which could potentially lead to an increase

of 4.2 % in network availability. This reduction in MTTR can contribute to reductions in operating costs with the ability to perform maintenance remotely using centralised VM rollouts over a network.

Although this solution focused mainly on the customer domain, the principles demonstrated in this study translate well into the distribution domain as well. AMI networks and distribution grid NANs can follow similar network topologies, and both can be unreliable and expensive to develop when using conventional methods. Because this solution was implemented through mathematical modelling, it lacked any examples of actual implementation and evaluation. Fortunately, other smart grid researchers have realised the potential benefits of virtualisation in communication network design and have subsequently implemented platforms that use NFV principles. The benefits of combining NFV with SDN have also been realised by researchers, especially to the provision of crucial network functions in a smart grid's edge networks.

Providing Critical Services in the Network Edge

Reference [59] studied the use of NFV with SDN to address the problem of dynamically providing Gateways to handle requests from heterogeneous devices connecting to critical networks as part of large-scale disaster management. Their proposed architecture, implemented on a locally distributed platform, included a provider application that could deploy network services, configured as per an application's needs, as VNFs on any available NFVI in the network. It also included virtual SDN-controllers and switches that were provisioned ad-hoc to create new networks and flow paths to support these services. Besides demonstrating the benefits of combining the features of NFV with SDN to provision M2M services in virtual platforms, this study also demonstrated a means to reduce the risk of functional and communication failures by locating critical functions closer to the devices that use them. Because VNFs and VMs can operate on almost any underlying computer infrastructure with sufficient resources, the ability is created to quickly repurpose existing networked infrastructure during emergencies to support specific smart grid functions. When these emergencies have passed, the infrastructure can be reassigned to its original functions. This will be useful for distribution grids operators who must maintain essential service provision during disasters. Virtualisation can therefore be considered one of the key enablers of mobile edge computing (MEC) and fog computing in smart grids.

Fog computing is a decentralised architecture that uses devices in the network edge to carry out a more substantial amount of computation, storage, and communication functions locally. A potential application of fog computing principles in a smart grid was reviewed by [60]. They developed a model that showed how fog servers could be used to create systems that separate private data from public data and how this separation can reduce network traffic into higher tier networks while providing better data security. Their solution was supported by a potential use case example, but unfortunately lacked any implementation examples. MEC, which is a more specific form of edge computing, has also been studied for its potential application in smart grid by [61], who proposed an architecture for enabling ubiquitous MEC in various smart applications, including smart grids. Based on the principles of SDSys and virtualisation, they argued

that their architecture could potentially support services for networking, storage, and security amongst others in edge networks. They supported this argument with an implementation of this architecture in platform that used virtualisation to create multiple SDSys controllers and virtual edge networks on a single computer. The researchers made use of Mininet which is a virtualisation-based network emulation system for this implementation. Tools such as Mininet will be very useful for designing the platform this work envisages, because it simplifies the process of implementing and evaluation designs for large-scale distribution networks.

When SDN and NFV is used in large-scale networks to provide functions and services in the network edge, the number of controllers and controller placement becomes a very important design consideration. Although the use of a single controller to control an entire SDN network is possible, it may increase the risk of controller overload and network congestion on the flow paths leading to the controller. If the only SDN controller in a network fails completely, flow tables will not be updated, and switches will continue to forward packets according to their last configurations until their flow entries expire. With no flow entries, switches will cease to function, resulting in network node failures. Reference [62] studied the impact of controller overload on the network performance for a smart grid automatic gain control (AGC) application. They found that the resulting processing delays led to communication delays that severely impacted the smart grid application's performance. To overcome this problem, multiple decentralised controllers may be introduced into network designs. This will allow controllers to share the network load and act as back-up devices for each other in case of failure. For controllers to have overlapping mastership roles over the same switches in the network, they need to be synchronised at regular intervals, which may result in additional synchronisation traffic on the network.

Reference [63] studied the optimal number of controllers for a NAN which supported a wireless sensor network (WSN), connected to 1000 smart meters in a neighbourhood. This was one of the few studies that based their research on an actual smart grid reference model when they considered a suburb in Melbourne, Australia for their design. Considering the QoS requirements for an AMI system, their simulations showed that one controller could handle around 30 switches in a NAN, which meant that four controllers would be needed to support this suburb. Besides the insight gained from this study on controller placement for smart grid NANs, the authors also provided another important contribution to this work which is their proposed reference architecture for SDN-based NANs. This architecture supports network topologies that divide the control load of switches amongst available controllers, ensures the availability of back-up controllers in case of failures, and supports uninterrupted functioning of the NAN even after it is isolated from the WAN. Implementing this architecture for a large-scale distribution grid network could be simplified with NFV, which may be a potential improvement to these authors' work. Instead of using simulations, network emulation systems such as Mininet may be considered by constructing the network topologies in a VM beforehand. The same analysis performed by these researchers on controller placement can then be performed for other network services that may be utilised in the network edge.

2.3.3 Reducing the Probability of Communication Delays

Most smart grid functions are not only impacted by messages that fail to reach their destinations, but also ones that reach their destinations too late. Smart grid applications have varying tolerances for delayed messages that can range from seconds to milliseconds. Smart grid applications have varying tolerances for delayed messages that can range from seconds to milliseconds. Smart grid communication networks must be designed to minimise communication delays and factors that contribute to communication latency must be considered when designing reliable distribution grids. Communication networks experience four types of delays: Delays in the propagation of the electrical, electromagnetic, or optical signals that carry information are called propagation delays (d_{prop}). These delays will increase in relation to the distance the signals have to travel. The time required to push a message onto a communication medium causes transmission delays (d_{trans}), while the time required to process a message and the time it awaits processing is called processing delays (d_{proc}) and queuing delays (d_{queue}) respectively. The latter three are all associated with the capabilities of network devices and the rate at which they can process data packets, push them on to the network and work through the packet queues that build up in their packet buffers. The total delay (D) a packet experiences while propagating through a network can therefore be expressed in the following formula:

$$D = d_{prop} + d_{trans} + d_{proc} + d_{queue} \quad (2)$$

Once a network has been constructed, further reductions in propagation delays are often difficult to achieve without replacing the communication mediums. Therefore, solutions that aim to reduce the probability of communication delays usually focus on improvements to the network devices. Upgrading the hardware of these devices can improve performance to a certain degree but may prove to be a very capital-intensive approach for large-scale networks. A lot of focus has therefore been on the optimisation and improvement of the network control plane functions that contribute to the major causes of communication delays. Three methods for reducing the probability of communication delays in smart grids have been identified in relevant literature. These methods are automated load balancing, automated traffic prioritisation and automated flow aggregation with traffic scheduling.

Automated load balancing

If network traffic flows are not constantly monitored and adjusted, communication bottlenecks can form at critical points in the network where resources are constrained. These bottlenecks will cause long packet queues to form that increase queuing delays and may even result in dropped packets when packet buffers are full. Network congestion can therefore become one of the main contributors to communication delays. To reduce congestion in a smart grid communication network, [64] developed a load balancing algorithm that alternated the path taken for each new flow at a switch, so that traffic was evenly distributed amongst all available flow paths between two points. Along with a FFR function, they implemented this load balancing function in an SDN-based WAN for evaluation against the QoS requirements of a demand response

management (DRM) application in the customer domain. What makes this implementation interesting, is the fact that it was done in a platform called the Global Environment for Network Innovations (GENI), which is a computational and network platform with resources that are geographically distributed across the United States of America. By using this platform, the authors were able to demonstrate the flexibility and rapid implementation of SDN-based network functions in a WAN subjected to real-world conditions. One of the shortcomings of their load balancing algorithm that might impact its usability in smart distribution grids, was the fact that it would likely encounter problems if uncongested alternative flow paths were not available in the network. In this situation, the load balancing algorithm might reroute traffic onto network flow paths that are already congested, thereby exacerbating the problem. It might also reroute traffic onto flow paths that have too many additional hops to a destination, which could increase the latency for time-sensitive messages. Enhancing this algorithm to become situationally aware might resolve this problem. This can be achieved with an application that evaluates the network traffic, flow path availability and travel distance proactively. The application can then update flow entries, accordingly, using the SDN controller interfaces to ensure network loads are balanced optimally without risk.

Automatic Traffic Prioritisation

Each smart grid application will have its own communication latency requirements even though they might share common network infrastructure. Problems can occur when high data volume applications overload network resources at the same time as time-sensitive applications require these resources to rapidly transfer their messages. The traffic prioritisation algorithm implemented by developers of *SDN4SmartGrids* was able to reduce the probability of delay for prioritised messages, but it was again dependent on preconfigured QoS class parameters. This reduced the function's flexibility and its ability to rapidly adapt to the dynamic environments. As further research, some of these authors performed another study, where they investigated the integration of a Multi-agent System (MAS) with a smart grid network traffic prioritisation function [65]. By associating agents with different smart grid applications and allowing these agents to advertise their QoS requirements, their prioritisation algorithm could now automate dynamic updates to the flow rules in the SDN switches via the SDN controller in response to these requirements. However, implementation of this solution was limited to a small hypothetical substation LAN in a laboratory. Besides demonstrating the benefits of implementing a dynamic automated traffic prioritisation function in a network, this study also provided insights into the benefits and means of providing network functions as flexible M2M services. Since this traffic prioritisation service was implemented as a software application, there is also an opportunity to implement it as a VNF and to make this a VNF-based service available in edge networks.

Automatic traffic scheduling and flow aggregation

Some smart grid applications in the distribution domain, e.g., data acquisition applications, do not have time-sensitive communication requirements, but they can generate a lot of big-data resulting in plenty of network traffic. To limit the additional burden that this traffic can place on constrained network resources, it

might be beneficial to temporarily store these data sets at nodes closer to their points of origin until network resources free up. This approach is referred to as in-network data buffering. Not to be confused with the packet buffers in network switches, this buffered data is usually stored in secondary storage devices and then scheduled for transfer at a time-of-the day when the amount of network traffic may be lower, e.g., midnight. These data transfers can also be triggered by network applications in response to events, e.g., user requests.

Reference [66] proposed a traffic scheduling and flow aggregation solution to prevent congestion-based data loss, while maintaining throughput fairness for non-time-sensitive data from smart meters. Supporting a meter data management (MDM) application in the customer domain, their application used an algorithm to schedule data for transfer based on a weighted value that ensured each meter received fair usage of the available network resources. Their SDN-based architecture was also implemented on a single computer using Mininet. A potential improvement to this design, as an alternative to forcing all meter data to flow to the aggregating host interacting with the SDN controller, could be to provide this flow aggregation as a M2M service for meter devices and applications. This will allow them to utilise the service when needed, using a standardised platform that may accommodate heterogeneous devices and applications. A NFV MANO that allows for ad-hoc provisioning of VMs and VNFs, can also be used to support dynamic scaling of the resources used in this design, allowing it to easily accommodate situations where additional buffer storage space or processing power might be required.

Reference [67] considered these approaches to not only support traffic scheduling and aggregation, but also to virtualise the functions performed by smart grid devices, i.e., Phasor Measurement Units (PMUs). Their proposed architecture made use of virtual PMU devices they called virtual objects (VOs) which could be made available as distributed resources with programmable interfaces to actual heterogeneous PMU devices in the field. It also consisted of a composition layer that contained virtual data aggregators which they called composite virtual objects (CVOs). These CVOs were responsible for parsing and aggregating data from semantically interoperable VOs. Data from the CVOs was then made accessible to different smart grid applications using either REST APIs for unicast or a publish-subscribe service for multicast scenarios. Their solution was implemented in a LAN connected to the internet. The LAN connected the VOs and some local CVOs and allowed them to access a cloud platform-as-a-service (PAAS) that hosted remote CVOs and applications that utilised PMU data. This study provided a useful architecture for deploying automated traffic scheduling and flow aggregation functions as M2M services using NFV. It also provided insight into another deployment option for software-based networking solutions for smart distribution grids, which is the use of PAAS in clouds. The use of PAAS could further reduce the implementation and maintenance costs for utilities as they would not have to burden themselves with infrastructure to host their local and potentially some of their remote VNFs and VMs. The design process for networks based on this architecture will also be simpler because an initial design can be performed on a single computer using virtualisation. The design can then be gradually rolled-out into the field by provisioning copies of VMs or VNFs to networked hosts.

2.3.4 Reducing the Mean Time to Repair Communication Problems

The communication reliability of a smart distribution grid is not only dependent on the ability to limit communication failures and delays, but also on the ability to quickly address these issues should they occur. Mean time to repair (MTTR) is a metric that is often used as an indication of a system or network's availability in terms of its recovery time after down time, over a specified period. The MTTR can be calculated by summing the downtime for each breakdown (t_d) and then dividing this by the total number of breakdowns (n) as expressed in the following formula:

$$MTTR = \frac{Total\ maintenance\ time}{Number\ of\ repairs} \qquad (3)$$

The MTTR is calculated from the moment a malfunction is detected until functionality is restored. It is an important communication reliability metric because each period in which a network malfunctions, is a period that the network operates with an increased probability of communication failures and delays. Reducing the MTTR in a network will therefore improve its overall communication reliability. The FFR, load balancing, traffic prioritisation and traffic scheduling functions discussed previously are all solutions that aim to reduce the MTTR of a network through the automation of network restoration functions. However, some corrective measures to network issues cannot be automated, either because they are too complex to resolve automatically or because they are unknown to network designers and administrators and must be identified and analysed first.

Remote network monitoring and control functions are vital for identifying these network problems that require urgent human intervention and for reducing the MTTR. OpenFlow makes use of flow monitoring messages to monitor in real-time, any changes to the subsets of flow tables. It can also make use of meters to measure and control the rate of packets assigned to these switches [56]. Using the programmatic interfaces to SDN controllers, applications have been developed to utilise this information to trigger alarms and to provide historical data perspectives for trend analysis. Application interfaces to SDN controllers can also provide direct control over the network by allowing users to update flow tables and flow rules directly. An example of one of these monitoring and control applications that was developed for the ONOS controller is the ONOS GUI [68].

Since distribution grids often make use of multi-vendor technologies in their networks, some networks in these grids may not make provision for SDN protocol-based communication. Challenges arise when SDN-based network applications must be used to monitor and control these heterogeneous networks. Reference [69] considered this problem for a SDN network that needed to interface, monitor, and control substation networks based on communication protocols related to the IEC 61850 set of standards. Their solution was to develop an architecture that used an SDN controller and the programmability features of OpenFlow to integrate OpenFlow with these IEC 61850 protocols. This allowed their network monitoring and

management applications to provide a global view of the SDN network as well as the non-SDN IEC 61850 substation networks. They implemented this architecture on a computer using Mininet. In addition, this study also proved that substation devices communicating with the GOOSE and SV messages specified by IEC 61850 could be modelled and implemented in network emulation systems such as Mininet.

These studies describe various methods other researchers considered for improving communication reliability in smart grids, while considering improvements to communication network development, evaluation, and deployment. The main aim of this work is to design and implement a software-based communication network that can improve the reliability of a city distribution grid, using a platform that streamlines the network development process. Therefore, the conclusions drawn from these studies may offer valuable insights into meeting this objective.

One of the first crucial components that must be included in a design for a communication network that aims to meet these objectives is a real-time network monitoring and control application that will provide grid operators with a global view of the network status. The reviewed SDN monitoring and control functions can provide this, as they are easy to incorporate in a software-based network design. The choice of SDN controllers and their associated interfaces therefore becomes an important design decision. Most of the platforms reviewed made use of SDN controllers based on OpenFlow, which might make it worth considering in this work as well. As was shown in [69], choosing a specific SDN protocol will not limit the network to that protocol, because most SDN networks can be integrated with other networks to support broader network monitoring and control. Software-based networking therefore has an inherent flexibility associated with it for accommodating the heterogeneous technologies that often exist in distribution grids.

To reduce the probability of communication failures with rapid automated recovery from link and node failures, network functions for proactive and reactive FFR as presented in [52] -[54] need to be considered next for this network. Furthermore, to reduce the probability communication delays caused by overloaded network resources, load balancing as in [53] and [64], as well as traffic prioritisation as in [53] and [65] also needs to be considered. Traffic scheduling and flow aggregation functions as in [60], [66], and [67] will be an important consideration for the EAM data acquisition functions that this network aims to enable, since these network functions can support the transfer of big-data that is not time-sensitive. Another important decision that emerges from the reviewed literature is the decision of implementing these functions directly as applications with interfaces to the SDN controllers, or as M2M services accessed ad-hoc by devices connected to the network. The use of M2M services may require a bit more initial configuration but will subsequently provide the network with more flexibility. The use of standardised service platforms will also offer more options to overcome the challenge presented by incompatible heterogeneous devices.

The network designs proposed in [52], [61], [66] and [69] were all implemented as VMs using virtual networking systems. These studies provide insight into how this work may achieve the objective of creating

a streamlined design platform running on a desktop or laptop computer. Creating these networks entirely as software will support rapid prototyping of different network designs where the scale of the networks is only limited by the availability of virtual resources allocated to the virtual machines. Since these systems make use of emulation rather than simulation, they also provide realistic results that are more representative of how these networks will react in real-world implementations. These systems also offer the ability for multiple VMs to execute different software applications simultaneously, supporting more extensive network testing of various functions and services. Various design alternatives can therefore be tested and improved with a few simple programmatic adjustments. Once a network design meets its specified requirements, these VMs can be rolled out to distributed computer infrastructure in the actual distribution grid to perform the same functions. Because SDN provides a decoupled network control plane with run-time configurability, the network will function the same way, whether it is implemented using VMs hosted on a single computer or multiple computers spread across an entire network. The only major differences that may be observable will be the impact of environmental factors that influence network performance, such as the physical distances messages must travel in the network. The results obtained from evaluations performed using these emulated networks should therefore be repeatable and scalable to larger network designs. By provisioning VNFs and VMs as software on commodity grid computer infrastructure such as substation computers or capable substation devices that may already be in place for existing distribution grid systems, the need for introducing additional network hardware may be reduced. This will not only reduce the amount of possible vulnerable network points, but also the MTTR malfunctions that may result in cost savings as was concluded in [58]. These savings may be applied to further grid enhancements, such as the systems that better protect and secure the grid's physical infrastructure. Multiple copies of these VMs and VNFs can also be created in the network to act as back-ups without the need for additional hardware.

There are many benefits to provisioning VNFs and VMs on NFVI located closer to devices that use them as was highlighted in [60] and [61]. The EAM system that will be supported by the network designed in this work will depend on large amounts of data generated in the network edge. Provisioning of services that support this data acquisition in these field networks therefore need to be considered. These functions may be implemented to improve network latency by managing network traffic and to limit the risk of data loss. Provisioning, updating and removal of these functions can occur ad-hoc over the network and may even be automated by applications interfacing with the NFV MANO. This adds another level of flexibility to the network. More importantly, VMs and VNFs providing services in the network edge can ensure continued operation of critical automated smart grid functions that are dependent on these services in situations where all connections with the core networks are lost. Applications can be configured to automatically fail over to alternative servers provided locally or in neighbouring grid NANs in case of local network problems, which will add another degree of grid reliability. Proper placement of these distributed services and functions however needs to be determined beforehand by simulating communication problems that the appropriate network functions will respond to. Network emulation systems may be used to simulate these failures without any risk on grid operations.

Although the results from these studies offer some insight into how software-based networking can be used to improve smart grid communication, further research is still required into how these approaches may be applied in the design of reliable distribution grids. As most of these reviewed studies focused on meeting specific communication requirements, they mainly considered implementations based on hypothetical network models that only focused on demonstrating the capabilities of certain networking approaches and functions. Even though they proved that they could meet their objectives, a question that remains to be answered is how these software-based networking approaches may be applied in real smart grid designs. Improving the overall reliability of a distribution grid will not only require the implementation and combination of multiple network functions in a flexible network architecture, but also a means to easily design and evaluate various design alternatives before they are implemented in real grid environments. These designs also require a more holistic consideration of the end-to-end smart grid implementation and should not be limited to the communication layer of the smart grid architecture. The reviewed approaches therefore need to be tested using models based on real distribution grids and more focus is required on the relationship between the grid's communication requirements and its infrastructure, data, and functional requirements. More research that considers big-data systems is also required. Bridging this research gap between the hypothetical grid and the real power grids of the world is something that, to the best of our knowledge, has not been sufficiently explored and we consider it necessary for greater adoption of smart grid in the distribution domain. Figure 16 classifies the reviewed methods that focus on improving communication in smart grids using IoT principles. A summary of these reviewed studies as it pertains to this work is also provided in Appendix C.

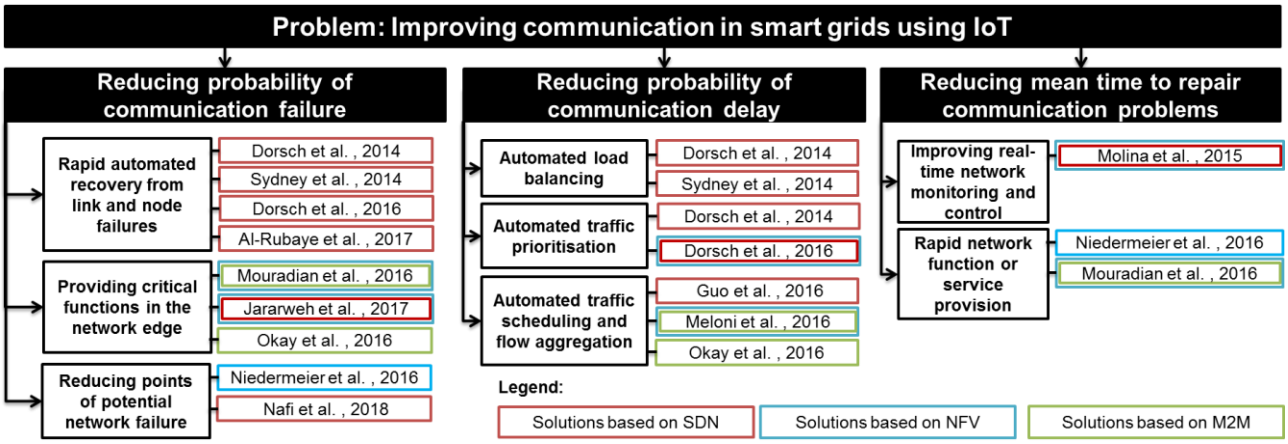


Figure 16: Taxonomy of related work.

2.4 Chapter Discussions

This chapter firstly presented an overview of smart grid communication in electricity distribution grids as background. Some background on software-based networking approaches was also presented. Three prominent frameworks that are related to the IoT paradigm, namely: SDN, NFV and M2M were also

discussed. Finally, related research that focused on improving smart grid communication using IoT frameworks was reviewed. This review highlighted various contributions that other research has made and that can support the development of the distribution grid communication network envisaged in this work. The review also revealed some of the gaps in these studies that this work aims to address.

The following chapter will introduce requirements and design considerations for an IoT platform that will be used to study the reliability improvements software-based networking can make in a city's distribution grid. Based on these requirements a new smart grid communication architecture will also be proposed, along with a process of how this architecture may be implemented. A communication network design for a section of the City of Cape Town's distribution grid, based on this proposed architecture, will then be presented.

Chapter 3

Requirements and Design

The previous section provided the background and reviewed the current work on software-based networking approaches that aimed to improve communication reliability in smart grids. In this chapter the design requirements for an SDN/NFV based IoT platform, which will be used to study the improvements in distribution grid reliability, will be established. With these requirements defined, a design will be presented for this platform. A proposed design for a reliable communication network that aims to support an EAM system in a section of the City of Cape Town's distribution grid will also be presented.

3.1 Requirements and Design Considerations

Overall, it should be noted that, since smart grids are integrated systems, each requirement in a specific smart grid operability layer described in the IEEE's smart grid architecture model covered in chapter 2, may impact requirements in other layers, domains, and zones. It is therefore important to consider an end-to-end view of a smart grid architecture when defining requirements for a grid communication network. Figure 17 illustrates this interoperability layer interdependency across the smart grid zones that are most prevalent in the distribution domain, highlighting the architecture elements that need to be considered.

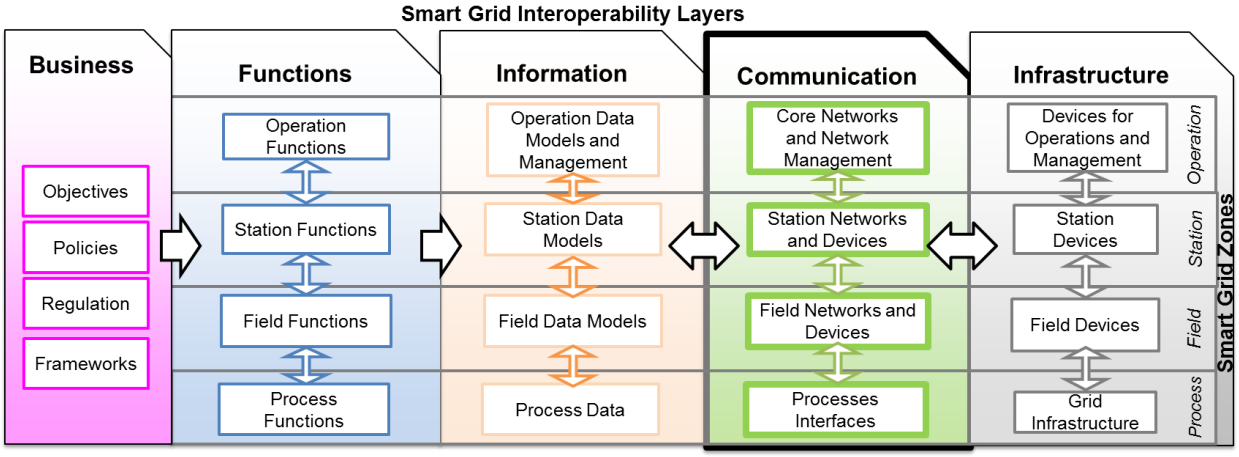


Figure 17: Interdependency between elements of the Smart Grid Architecture Model.

Starting with the business layer, the main purpose of a distribution grid is to distribute electricity to customers, reliably, safely, securely, and efficiently. The requirements specification for a smart distribution grid communication network therefore needs to start with the high-level requirements associated with the grid objectives, which must align to policies, regulation, and business frameworks. Integrated grid functions need to be defined to support these objectives, which will result in more detailed information, communication, and infrastructure requirements. The energy and information flows in smart distribution grids are bi-directional, and consequently these requirements need to consider how the location and flow of

information will impact the flow of electricity. Grid information will originate from different sources, located at different points in the grid. Information requirements for a smart distribution grid therefore should not only consider the type of information grid functions will utilise, but also the different formats this information will take on and its possible storage locations. Information that needs to be transferred between different locations in the grid will determine the distribution grid's communication requirements, which will result in requirements for infrastructure that can support communication. The grid's existing infrastructure, layout and operating environment will also influence communication requirements since these factors may limit the options available for ICT selection. The selection of specific grid infrastructure to support the distribution grid processes may deliver its own set of communication requirements and information requirements that is technology specific.

Considering this interdependency among smart grid interoperability layers, careful consideration needs to be made for all layers when establishing requirements for a communication network that aims to support improved distribution grid reliability. For this work two approaches have been considered to meet this objective. The first approach is to develop a communication network that can support a smart grid system developed with improved grid reliability in mind, i.e., an EAM system. This will place the focus of the communication layer design on supporting and enabling the functions layer to meet the grid objectives. The second approach is to implement network functions in the communication network design that will work together to improve the distribution grid's overall communication reliability. This will focus on addressing the grid objectives more directly with the communication layer design.

3.1.1 Development of Communication Networks that Enable EAM Systems

In response to the demand for modernisation of the world's power grids, various smart grid systems have emerged to address different grid challenges. Although many of these systems support the objective of improving grid reliability, EAM systems differentiate themselves by taking a long-term view of the grid asset lifecycle. These systems improve grid reliability by using analytics to discover trends in gathered grid asset data, and then using this information to identify potential threats in the short, medium, and long term. An example of this information is a steady increase in the average winding temperature of a transformer over a period of many years that eventually leads to failure.

With this information, maintenance interventions and replacement projects can be planned well in advance, ensuring that the needed resources will be available when required. Because EAM systems rely on data acquisition and data analytics, they can be very data intensive. They are however not as dependent on low network latency as most other smart grid systems as these data sets are usually not analysed in real-time. The recommended end-to-end (E2E) communication latency, bandwidth and reliability for the most common smart grid systems is summarised in Table 1. Note that the actual communication requirements for each

smart grid implementation will vary on a case-by-case basis and that these recommendations are only provided as a guideline for smart grid designers.

Systems	Examples	Bandwidth	Latency	Reliability
Mission-critical systems:	Substation Automation, Distribution Automation, Overhead Line Monitoring and Wide Area Situational Awareness.	9.6 – 1500 kbps	15 ms – 200 ms	99% - 99.9999%
Systems with lower data volumes:	Distribution Management, Home Energy Management, Distributed Energy Resources, AMI, and Outage Management.	9.6 – 100 kbps	100 ms – 2 sec	99% - 99.99%
Systems with higher data volumes:	Demand Response Management, Electrical Vehicles, EAM and Meter Data Management.	56 kbps and above	2 sec – hours	99% - 99.99%

Table 1: Recommended communication requirements for smart grid systems. Adapted from [70] and [71].

From these recommended communication requirements, it is noted that communication networks that support EAM systems are typically expected to offer communication reliability guarantees ranging from 99% up to 99.99% as a minimum. A communication network supporting an EAM system must therefore be designed to ensure that all data generated by the grid devices supporting the EAM system reaches its destination within a reasonable amount of time, at least 99% of the time. Most importantly, this network must ensure that no data is lost during any data transfers. Furthermore, according to these recommendations, EAM systems require networks with higher bandwidths to accommodate the transfer of data samples from hundreds of potential assets. In distribution grids, most of these assets will be in and around substations where existing systems such as SCADA may already be in place. The IEC 61850 standard, which is one of the most widely recognised standards for both inner-substation and inter-substation communication, specifies the message types of substations may utilise along with their required E2E latency and the sizes of typical messages [72]. These message types are listed in Table 2 and show that EAM systems, that typically make use of file transfers and low speed messages for continuous data acquisition, can expect messages ranging from 16 bits up to 200 kilobits per message.

Message Type	Required E2E Latency	Typical Message Size
Fast messages	< 3 ms - 100 ms	1 bit
Medium speed messages	< 100 ms	1 bit -16 bits
Low speed messages	< 500 ms	16 bits – 1024 bits
Raw data messages	< 3 ms - 10 ms	12 bits – 18 bits
File transfer functions	> 1000 ms	512 bits – 200 kilobits

Table 2: Message types used substation communication [72].

Since the data used by EAM systems is usually collected and transferred in batches, file transfer functions need to be considered for such a system. To manage the acquisition of batch data from many sensors placed in critical grid assets, EAM systems often use devices to temporarily store and aggregate data closer to their points of collection. These data sets will eventually reach centralised data repositories, where processing applications can access it. Figure 18 illustrates a typical example of data flow for an EAM system implemented in a smart distribution grid.

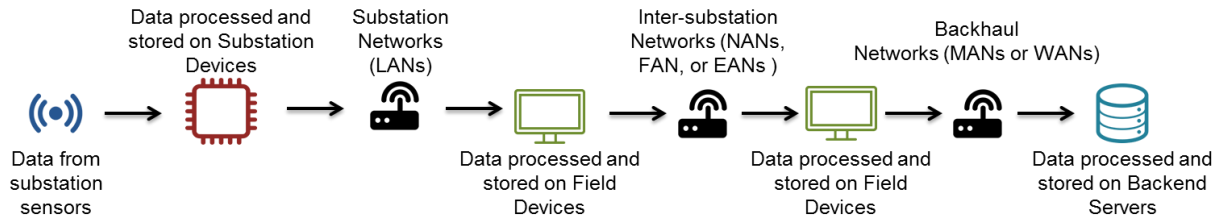


Figure 18: Data flow for substation data acquisition.

To establish the requirements needed to support this data acquisition, the communication network requirements need to be defined in relation to the requirements that exist in the other operability layers of the SGAM. Figure 19 shows a mapping of the elements in each layer that must be considered when establishing these requirements for an EAM system.

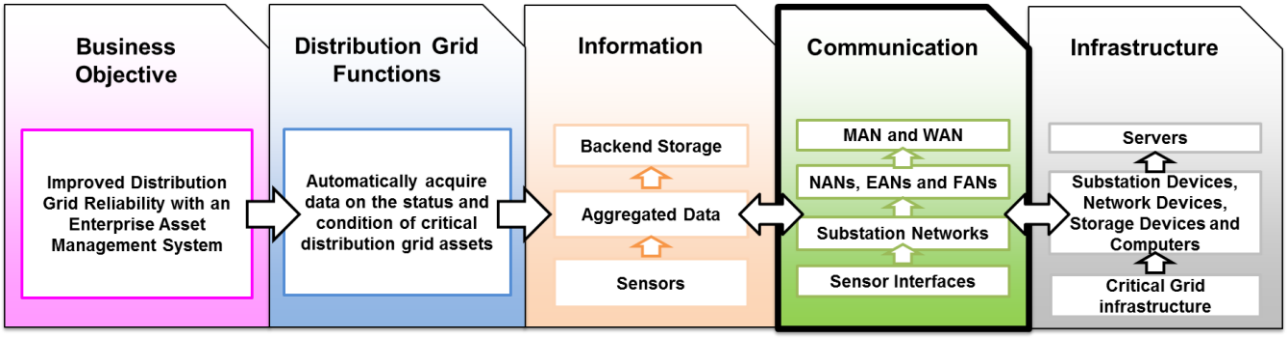


Figure 19: A smart grid architecture for an EAM system in a distribution grid.

Information Layer

The data for an EAM system can be stored in a variety of formats. A text file is one of the basic formats for capturing long strings of text data that is often supported by various substation technologies. Data models need to be defined to standardise the data representation in these text files. For the purposes of this work, a simple data model for storing substation sensor readings as text strings in text files is proposed and shown in with an example of a data string makeup in Table 3.

Substation Identifier	Device identifier	Data separating character	Sensor Identifier	Sensor Reading	Date stamp	Time stamp	String separating character
Green Point	Transformer	{	Primary Voltage	11.1 kV	1/1/2020	12:15:29.45	}

Table 3: Proposed data model and example for storing sensor readings as text strings in text files.

Assuming text files are chosen as the storage format for data acquisition, file transfers will be required to move these data sets between grid devices. If these text files can be transferred directly without any data manipulation in the field, it will help to reduce the processing overhead of field devices. The File Transfer Protocol (FTP) is an option to consider for these file transfers. FTP was developed with the upload of large

files from clients to servers in mind and is widely supported because it has been around for many years [73]. FTP uses the transfer carrier protocol (TCP) to facilitate its communication and it divides files into smaller sections during transfer to limit the size of each message. The transfer of larger files will therefore require more messages. Although FTP is not optimised for very low latency transmissions, it is a protocol commonly used for large batch data transfers that are not very time-sensitive, such as those required by EAM systems. Furthermore, enhancements of FTP such as FTP over Secure Sockets Layer/ Transport Layer Security (FTPS) and FTP over Secure Shell (SFTP) offer ways to transfer these files more securely.

The devices supporting the data acquisition functions will likely also require remote management. The communication requirements of the related management applications subsequently also need consideration. To accommodate the downstream and upstream messages used by these management applications alongside the file transfers, this communication network design needs to accommodate two-way communication flows with consideration of the appropriate communication protocols.

Infrastructure Layer

For this work the City of Cape Town's distribution grid was chosen as a reference for a large-scale grid in a metropolitan area that could benefit from a smart grid EAM system implementation. A description of the Green Point grid section in the Cape Town distribution grid that was considered for this design implementation is provided in Appendix C. Assuming a planned smart grid implementation for this distribution grid supporting an EAM system, a network needs to be designed that will provide the needed communication interfaces to various devices receiving readings from sensors connected to grid infrastructure. With the large variety of sensor technologies available today, information about almost any aspect of a distribution grid may be obtainable with the right sensors, networks, and applications. Careful consideration must go into exactly which elements of the grid to monitor for an EAM system, as too much data from too many sources can easily lead to an information overload. A substation layout with a proposed sensor network for an EAM system is shown in Figure 20.

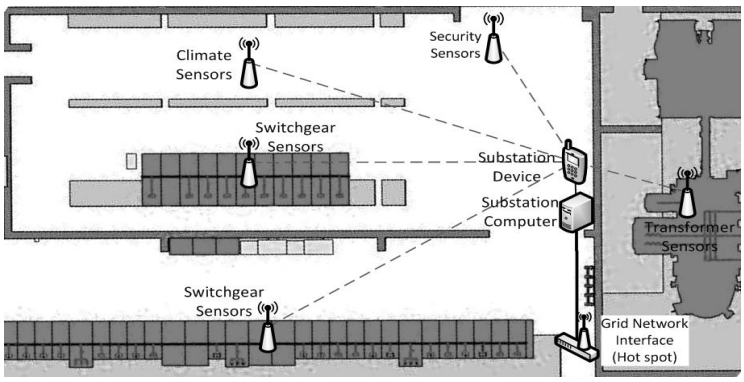


Figure 20: Proposed substation sensor network supporting an EAM system.

For this design, we considered sensors that can provide data to help address three common causes of critical grid asset failure in Cape Town's distribution grid namely: theft, vandalism and grid overloads caused by illegal connections [74]. The sensors used by security systems and those used to monitor the climate inside substations can provide valuable information to the EAM system to help detect potentially vandalised assets. Voltage and current sensors placed inside transformers and switchgear can provide data that can help with the detection of anomalous load changes which may aid in the identification of illegal electricity connections. Substation computers that are often used to support other substation functions may also be integrated into the network design to support the data acquisition process.

Communication Layer

Network topologies need to be designed to connect these substation sensor networks to the Green Point NANs and the city's MAN. The network topology for every smart distribution grid will be unique to its grid environment and would therefore be subject to different design constraints such as the geographical grid layout and the available options for communication mediums. These topologies must be designed within physical design constraints, while ensuring reliable communication with the provision of adequate back-up flow paths and back-up communication infrastructure. A proposal for the Green Point NAN topology that considers the layout of the Green Point distribution grid is shown in Figure 21. In this diagram each switch is named after the primary or main substation in which it will be located.

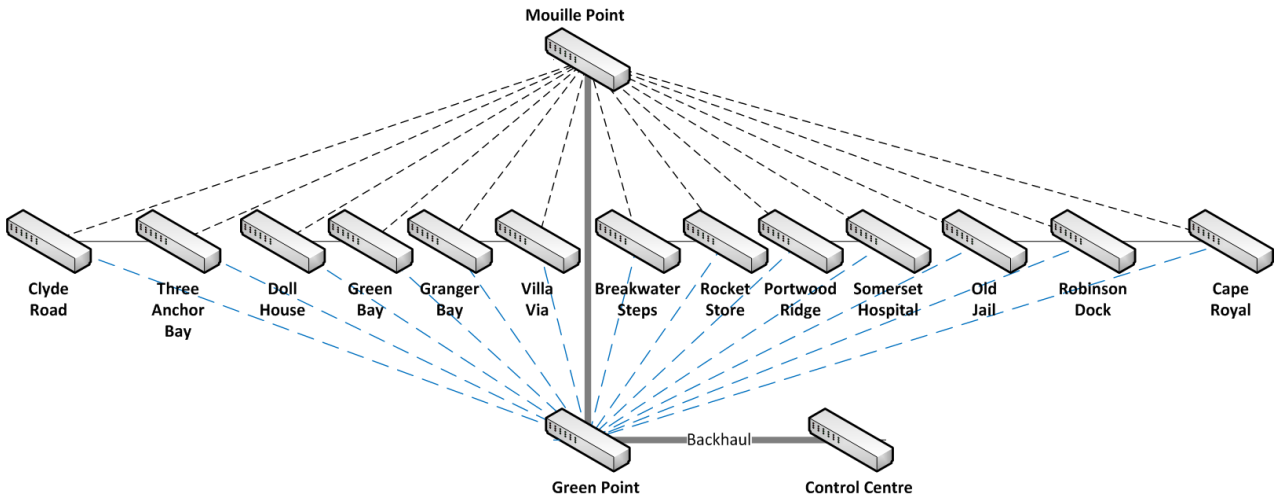


Figure 21: Proposed network topology for the Green Point Distribution Grid section.

A star topology was chosen for this NAN. All switches are connected to a switch in one of the neighbourhood's main substations, called Mouille Point substation. They are also connected to a switch in the neighbouring Green Point main substation to form a back-up NAN network for the grid section. Each switch in the NAN also has a direct link to a switch in a nearby substation. This topology ensures that all switches in the network have at least three separate connections to the network, with direct connections between the substations that will most likely communicate with each other the most. Connections to a

backhaul network connecting the two main substation switches to the MAN is also shown in this topology. These must all be TCP/IP based Ethernet networks that can support the required FTP file transfers, as well as the control and monitoring messages used by management applications.

Considering that this communication network will only cover a section of the distribution grid, which may be prone to frequent changes as the grid expands to serve new customers, its incorporation into a larger city-wide distribution grid communication network will require a large measure of flexibility. Current hardware-based methods for developing distribution grid communication networks lack this flexibility and require extensive capital investment each time a communication network is developed or changed. The use of network hardware infrastructure also drives up the operating cost required to maintain the network and can reduce the network's communication reliability. Another important requirement for this communication network will therefore be a flexible development platform that will streamline its design, evaluation and deployment while minimising the need for physical network setup and configuration in the field.

3.1.2 Improving the Communication Reliability of Smart Distribution Grids

Smart grids must support bi-directional energy and information flows to achieve the objectives of the modern power grid. Because most of the systems in a smart grid depend on automated and intelligent grid management systems, the reliable flow of power will in many cases depend on the reliable flow of information. This interdependence between the power supply reliability and communication reliability in a smart grid is explained in the scenario illustrated in Figure 22.

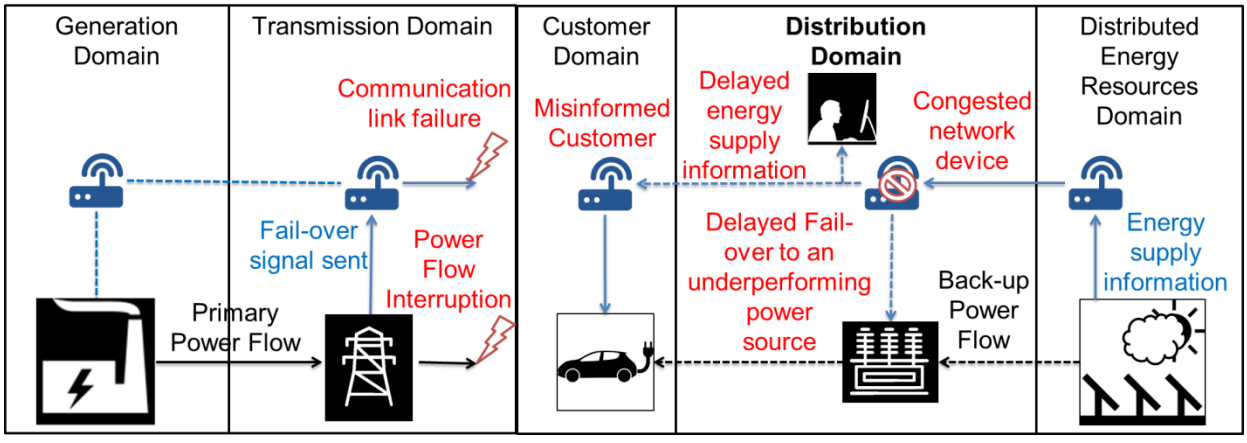


Figure 22: Interdependence between power supply reliability and communication reliability.

In this scenario, a power supply disruption detected by a sensor in a transmission substation triggers a failover signal to another substation in a distribution grid, which will allow it to switch over to a local solar PV plant. A damaged fibre-optic cable in the communication network will prevent this failover signal from reaching the other substation, thus leading to a supply disruption. An operator notices the power supply disruption and manually switches over supply to the solar PV plant. However, the operator does not know that this plant is not currently generating electricity, because a malfunctioning network switch is delaying the

transfer of status information. This results in a switch-over to an unavailable power source. Delayed information about the supply disruption, caused by the malfunctioning switch may also leave angry customers uninformed. Furthermore, any grid communication devices that depend on the disrupted electricity supply will also be impacted, which can lead to further communication problems. Designing smart distribution grids with functions that improve communication reliability is therefore vital to improving the overall reliability of the power grid. To improve a smart grid's communication reliability, mechanisms need to be put in place that reduce the probability of communication failures, reduce the probability of communication delays, and reduce the mean time to repair communication problems.

Reducing the Probability of Communication Failure

The first set of requirements for networks that aim to improve the communication reliability in smart distribution grids must be to implement functions that reduce the probability of failures at the communication source, destination and in the communication channel. The main cause for communication failure at a message's source or destination in distribution grid networks considered in this work is malfunctioning communication hardware or software on devices that transmit or receive data. Considering the described scenario, an example of a source failure would be the transmission substation attempting to send the failover, but finding that its network interface is malfunctioning, thereby preventing it from communicating. Because this failover signal is time-sensitive, the grid failover function would fail. If, on the other hand, the PV power plant transmitting its status information was disconnected from the network, its accumulating status data could still be useful for some smart grid applications. The data transmitting device connected to this PV power plant should therefore be configured to store these data sets until its connection to the communication network is re-established. It should then reattempt to transmit these data sets. While any device that is supposed to receive status information from the PV power plant is disconnected from the network, transmissions will fail and the data from the transmitting device must be stored as well. Since many of these devices rely on limited amounts of embedded memory for storage, there is a risk that these devices might run out of storage space, which will result in data loss. To overcome this problem, in-network data buffering functions must be included in the network design to provide additional storage capacity. These functions will act as temporary data repositories that take over the data transmission responsibilities from the source devices. In network data buffering will also only work for smart grid functions that make use of non-time-sensitive messages. For time-sensitive grid functions it may be advisable to automatically switch over data transmissions to back-up devices as soon as a communication problem is detected. If these switchovers can be performed proactively, it can reduce the risk of destination device unavailability during critical moments of data communication while simultaneously reducing the dependence on human intervention for these functions to execute.

A link failure in the transmission domain's communication network is illustrated in the scenario depicted in Figure 22. The scale and the distributed, often hazardous, environments smart grids operate in can increase the risk of communication link and node failures [75]. Failure of network devices such as switches, called

node failures, will impact multiple network links simultaneously. When a network link or node fails, its associated communication flow paths become unavailable for use, which means alternative flow paths need to be identified and provisioned. This needs to happen as quickly as possible to minimise the impact on communication latency. Network functions are therefore required to provide network administrators with the ability to identify and remotely repair network link and node failures or allow them to switch over traffic flows to back-up flow paths. Functions that can automate some of these repair tasks are also required to limit the impact on time-sensitive grid functions.

Most single link, double link and node failures can be repaired easily with fast failover recovery functions, but when multiple important links or nodes fail simultaneously in a communication network, recovery can become more complex. Although not very common, multiple link and node failures can be caused by natural disasters or as part of attacks on the network, resulting in subsections of the communication network becoming isolated from the core networks. In the depicted scenario, if the communication networks in the distribution domain were to be isolated from the smart grid's core networks, it would prevent the devices in this domain from exchanging any messages with other domains. This would disable any inter-domain failover messages from being exchanged. Because the networks within the distribution domain may still be unaffected by these failures, a reduced measure of localised grid functions may still be performed. Distribution grid communication networks must be designed to maximise these limited communication capabilities to retain some level of grid functionality. Therefore, they require functions that will support automated failover mechanisms to locally provisioned services that can be utilised to sustain grid operation until full network connectivity can be re-established. Overall, the communication network functions implemented in this design should aim to limit the impact of any communication failures on the execution of grid functions. To evaluate this requirement for the communication network design envisaged in this work, interruptions to the implemented EAM data acquisition functions must be measured while the network is subjected to various simulated communication failure scenarios.

Reducing the Probability of Communication Delays

A message transfer in a distribution grid communication network can only be considered successful if the message reaches its destination in time to execute the required grid function before it is too late. According to Table 1, some messages such as the depicted failover message from the substation in transmission domain will require end-to-end latencies as low as 15 ms, while others such as the PV power plant's status messages may be able to tolerate message delays up to 2 seconds. A smart grid communication network must be designed to minimise the chances of communication delays as far as possible. To reduce the probability of communication delays that may cause these messages to exceed their maximum tolerable latency, two factors that influence the E2E latencies of messages in smart distribution grids are considered in this work. Firstly, the distance a packet must travel in terms of the number of hops through the network's nodes. Each hop in the network will contribute to the packet's overall processing, queuing and transmission time. Hence, network functions that will ensure the use of the shortest flow paths between sources and destinations are

required. This must also hold true for situations where the grid topology changes frequently. It is thus important for the network's routing paths to be monitored and updated regularly, considering a global view of the network topology.

Secondly, the amount of time each packet spends at a node before it is transferred, needs to be optimised for efficiency. One of the main contributors leading to increases in the time that a packet spends at a node is underperforming network devices. Network devices may underperform if their underlying hardware malfunctions or if their software control plane is not optimally configured. Real-time communication network monitoring and historical network traffic trend analysis may be used to identify underperforming network devices, which may then be repaired or replaced. Another contributor to packet transfer delays in distribution grid communication networks, especially those that handle large data volumes, is the overloading of network devices. All communication packets received by a network device will enter a packet queue for processing. Packet prioritisation functions may ensure that critical packets will be placed at the front of the queue, but in some circumstances the amount of network traffic to a network device may exceed the device's maximum buffer capacity. In these circumstances, the network device will not be able to accept any more packets until it has worked off its backlog, which may result in packets being dropped. Dropped packets will require retransmission, that will contribute significantly to the amount of time it takes to get a message across. Network functions that can control the amount of traffic on overloaded portions of a communication network are required to address this issue. These functions need to be flexible so that they can adapt to changing network circumstances. They also need to be automated for rapid execution and configured in a manner that would not cause them to disrupt any smart grid functions.

Reducing the Mean Time to Repair Communication Problems

When a malfunctioning network device or broken network link is identified, communication needs to be restored by diverting network traffic away from the problem nodes and links. For this reason, alternative flow paths need to be identified and established before communication problems occur and these flow paths need to be updated regularly. If no alternative flow paths are available, steps need to be taken to ensure no data earmarked for transmission gets lost. The problematic devices and channels then need to be analysed, repaired, and tested before data exchange can resume on them. Although automated network functions can be used to address many of these communication problems, network administrators still require a global view of the network configuration and status. With detailed reporting and alarm mechanisms administrators can identify network problems that cannot be repaired by automated functions. Remote network control functions are required to allow them to perform repair tasks remotely, thereby eliminating the need to travel to field devices for repairs. In large-scale smart distribution grid communication networks, the scalability of a repair process is of particular interest. The difficulty of locating a problem increases with the size of the network, as does the time to repair the problem when multiple network devices and links are impacted by it. Network functions that aim to improve communication reliability must therefore be scalable for deployment in massive, distributed network environments. Appendix D summarises the three communication reliability

requirements established in this section and maps each of these requirements to the smart grid communication layer and the functional layer objectives they aim to address.

3.2 Design

The results from the studies reviewed in chapter 2 demonstrated various benefits that communication architectures based on SDN, NFV and M2M can offer to improve smart grids. Based on the conclusions drawn from other research, Appendix E provides a summary of the arguments made in this work for the use of a communication network design based on the pillars of SDN, NFV and M2M in distribution grids. Given these arguments, a strong case can be made for designing reliability focused communication networks for large-scale distribution grids supporting cities and large metropolitan areas, using a software-based approach. Therefore, several features of the reviewed communication architectures are incorporated into a proposal for a new communication network architecture that focuses specifically on supporting distribution grids. This proposed architecture will be presented in this section, along with a proposed process to be used when considering the implementation of this architecture using a virtual networking platform. Finally, a distribution grid communication network design based on this architecture and process will be described for the presented Cape Town grid section model.

3.2.1 Proposed Platform Architecture

The functional architecture proposed in this work is presented in terms of the four smart grid zones that are most relevant to the distribution domain. Note that, although this architecture is designed with the electricity distribution domain in mind, it may also be adapted for use in other smart grid domains. The elements of each grid zone will now be described in a bottom-up approach. Firstly, the process zone represents the distribution grid infrastructure that supports the electricity distribution processes. This includes electricity infrastructure, machines, and buildings. In distribution grids this layer will describe the distribution substations and other distribution field equipment. The sensors and actuators connected to the distribution infrastructure are also included in the process zone. These sensors and actuators are connected to devices in the field zone. In most cases these connections are physical copper connections that transfer electrical signals.

The field zone will contain the devices that transform these signals into data for transfer using the distribution grid's communication networks. Smart grid devices in this grid zone of a distribution grid may include stand-alone computers, laptops, mobile devices and in many cases, devices that are integrated directly with the grid infrastructure as part of a "smart" unit. Devices located in and around substations will be hardened to withstand environmental hazards and they will be equipped with hardware for data processing, storage, and communication. Many of the functions performed by these devices can be virtualised, which is why NFVI and VMs are also included in the field zone of this architecture. The devices

in the field zone will communicate among themselves and with elements in the station zone using internal substation networks (LANs), NANs, EANs and FANs.

The communication nodes in these networks, i.e., the network switches, are contained in the station zone. By implication, the devices in higher tier substations and other facilities acting as NFVI for virtual switches and SDN controllers, will become the distribution grid's edge network "hot spots". The NFVI in the station zone will then connect the distribution grid's field components to the grid's MANs or WANs by providing interfaces to the grid's backhaul networks. The NFVI in the station zone will also include instances of M2M Internetworking Proxies (IPEs) that act as interfaces for non-M2M devices, and in some cases M2M Gateways that provide edge networks service needed at specific points in the distribution grid.

The operation zone contains the NFVI that hosts the domain level SDN switches and controllers for the distribution grid's core communication networks. VMs hosted on this NFVI also contain the M2M Backend services. The distribution grid's smart grid applications, network management applications and M2M management applications are also located in the operation zone with interfaces to the relevant NFVI and VMs. This includes the SDN applications for network monitoring and automated network control functions. The NFV MANO also operates in the operation zone, where it manages the provisioning, updating, resourcing and termination of all VMs on available NFVI in the network. The proposed architecture is shown in Figure 23.

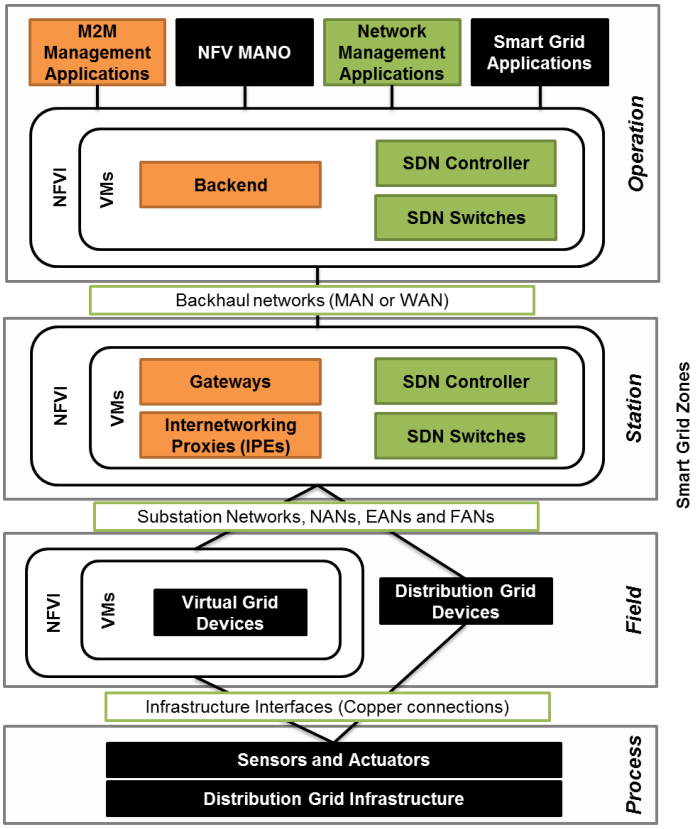


Figure 23: Architecture of the proposed solution.

3.2.2 Proposed Process for Network Implementation, Evaluation and Deployment

Given the proposed platform architecture, a process is proposed for applying this architecture in support of the design, evaluation and deployment of communication networks using platforms based on virtualisation. The proposed process is illustrated in Figure 24.

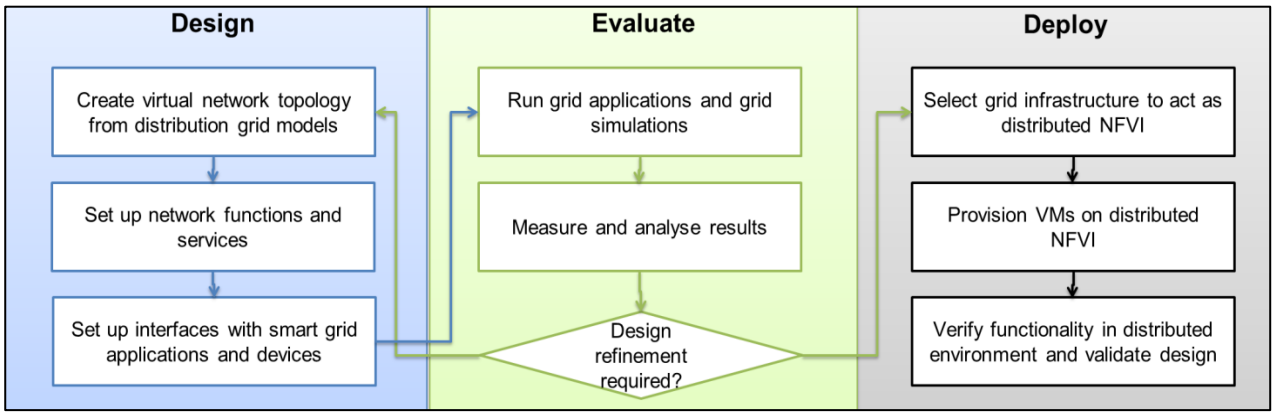


Figure 24: Process for designing, evaluating, and deploying networks using virtualisation-based platforms.

The process starts with the creation of a topology design using a virtual networking system. This will allow any network topology design to be created as a virtual network hosted on a single computer, provided it has sufficient resources to run multiple VMs simultaneously. Network functions and services that support monitoring, control and automation can then be set up along with the required interfaces to smart grid applications and devices. With an initial network topology design set up in a virtual network, these applications may be executed to evaluate the network's capabilities, as well as its ability to support specific smart grid functions. Various network problems should be simulated to evaluate their impact on communication resilience, grid functionality and to test the capabilities of the automated network recovery functions. Applications that monitor and report network activity should also be tested in these virtual platforms. After the evaluation has been completed, a decision can be made on whether the network design has met the requirements that will justify its implementation in an actual grid environment or whether these designs need to be improved. The flexibility offered by the virtual networking platforms makes it easy to change designs on the fly and to re-evaluate these designs with minimal effort. When an evaluated virtual network design is considered ready for deployment in a real distribution grid environment, copies of the VMs and VNFs created in the design platform can be provisioned to designated grid NFVI. Substation computers, servers and even capable substation devices may be selected to act as this NFVI, if they are equipped with the required hardware resources. A capable NFV MANO will be needed to provision these VMs and to manage them in a large-scale distributed environment. Once deployed a final verification of the network can then be performed, followed by validation that the deployed communication network in its distributed configuration can still meet its specified smart grid communication requirements. Any subsequent

changes can be made and tested in the design platform first, before the updated network designs can then be deployed as new VMs.

The virtual network topologies created and evaluated with this process will include virtual network devices that are software representations of the components that perform network-related functions e.g., network switches and network controllers. They will also include virtual hosts that are VMs representing the machines that exchange messages on the virtual network. These virtual hosts will generate, process, store and push data onto the virtual network which will require routing by the virtual network devices to other hosts. Because they are VMs, the virtual hosts can be configured to perform almost any function, which include the running of specific smart grid applications and the providing of M2M services. Virtual network links that are direct, port-to-port communication interfaces that can be setup on a local VM to serve as communication channels between virtual devices on a single machine will also be used in these virtual networks. These virtual links function exactly like real network channels, with the exception that they are not subject to environmental effects such as interference and propagation delays, although some applications can simulate these effects if needed. When designing these network topologies, a good rule of thumb for distribution grid communication networks is to ensure that a minimum of two alternative flow paths are maintained from each switch to the core network. This will improve communication reliability with multiple back-up flow paths, noting that EANs and FANs located in rural areas will likely have fewer switches and communication links available to them, which may limit the options for creating back-up flow paths in these networks.

Two other important topology considerations are the placement of SDN controllers and the provisioning of interfaces to communication networks in other smart grid domains. We offer the reference model for communication network topology design shown in Figure 25, as a guideline for placing SDN controllers in distribution grid communication networks and for identifying potential network interfaces. This reference model is based on the SDN based network architecture of heterogeneous communication networks for smart grid reviewed in chapter 2 [63].

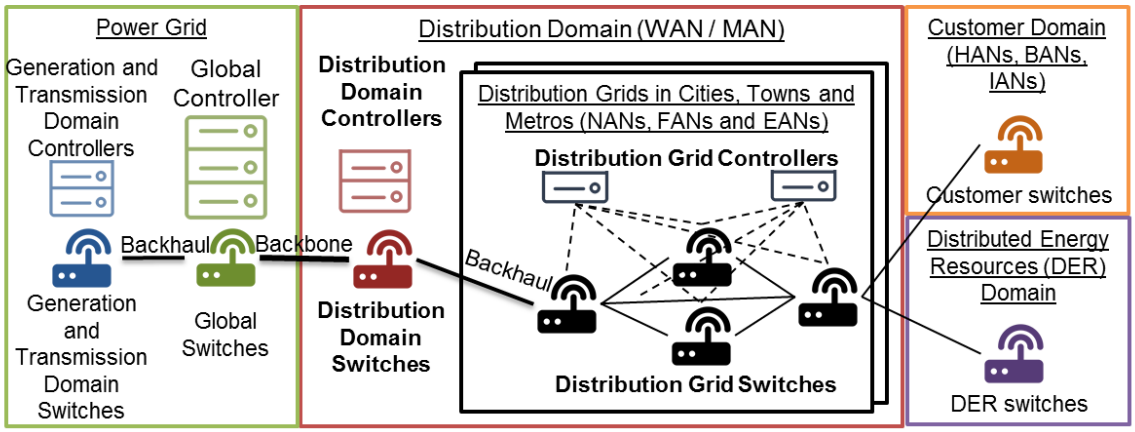


Figure 25: Reference model for designing network topologies for smart distribution grids.

All distribution grid communication networks should start with the placement of domain level switches and domain level controllers. The domain level switches will support the distribution grid's core network, which means it will function as the interface between the backhaul network links and the grid's backbone networks. The distribution domain controller should be provisioned close to the domain switches to support communication control in the core networks and to act as the master controller for the domain. The backhaul network links will connect the domain level switches to distribution grid switches that connect to other switches in the surrounding area to form the NANs, EANs and FANs in the distribution grid's edge networks. Distribution grid controllers should be placed in these networks to provide control functions closer to edge devices and to reduce the risk of SDN switches losing all connectivity to a SDN controller when network failures occur. Multiple SDN controllers in neighbouring networks can share control of their assigned switches to add another degree of reliability to the network design.

Distribution grid switches may act as interfaces for networks located in the customer and distributed energy resources domains, providing smart grid access for customer devices and smaller renewable energy plants. The distribution grid communication network's domain switches may also interface with the other large smart grid domain communication networks, e.g., the generation or transmission domains, via the smart grid's backbone networks. These backbone networks will be supported by global network switches and global SDN controllers that ensure controller synchronisation for the entire smart grid communication network. Overall, a final evaluation of a distribution grid communication network topology design based on this reference topology should verify that each network switch has multiple flow paths available for it to use to reach the core networks, that each network device has multiple controllers capable of controlling it, and that each host is reachable by all other hosts in the network.

3.2.3 Proposed Design for a Software-based Network that Supports an EAM System

Using this architecture, a design for a software-based communication network that aims to improve the reliability of a section of the City of Cape Town's distribution grid can be developed. As the first step in the proposed process, this design can then be implemented in a virtual design platform for evaluation. For the proposed communication network design, a substation computer will be placed in each primary and main substation to act as the station and field zone NFVI. These substation computers will then function as the network "hot-spots" for the distribution grid devices near them, offering network connectivity via the SDN switches that they will host as VMs.

Each substation will also have at least one substation device that will have direct interfaces to the sensors that collect data about the critical grid assets for the EAM system. The primary substations in the distribution grid will be the main network interfaces for devices in smaller secondary substations, mini-substations and other distribution grid field infrastructure that do not have their own substation computers. In addition to hosting an SDN switch, the primary substation computers will also host M2M IPE services that will

transform all data received from the distribution grid devices to a common data model used by M2M data acquisition services. The IPE will also manage the low-level data aggregation and storage of received sensor data. The substation computers located in the main substations will host the M2M Gateway services, which will provide another level of data aggregation in the edge networks, along with other network service functions. Distribution grid SDN controllers will also be hosted on these main substation computers to provide edge network control in the NANs.

Servers located in the distribution grid's control centre will serve as the NFVI in the operation zone. These servers will host VMs for the domain SDN controllers and the Backend M2M services, which will be the central repository for the acquired grid asset data. Running alongside these VMs will be the EAM applications themselves that will access this stored data for analysis. Applications that manage the distribution grid devices and remote storage will also run on these servers, along with the NFV MANO that will provision all the required VMs and manage their resources. Most important for managing the functions that will improve the distribution grid's communication reliability will be the network management applications that will also run on these servers.

Considering the established requirements, network functions that automatically discover the entire network topology and update the topology information when the network configuration changes, will be included in this design. These functions will also ensure that the shortest routing paths will be maintained for each discovered network switch. Applications that support automatic FFR and traffic scheduling with flow aggregation will be included in this design, with the potential to add more of the reviewed network functions that support communication reliability at a later stage. FFR functions will be the main recovery mechanism for communication link and node failures. The benefits and downsides of both proactive and reactive FFR functions have been reviewed in the previous chapter and it was concluded that a hybrid FFR approach offers the best option in terms of flexibility and performance for distribution grid applications. Automated traffic scheduling functions will be implemented as M2M services and configured to support automated failover to back-up devices to manage destination device unavailability.

Lastly, applications that support real-time network monitoring and run-time network control will also be included in the design, with a GUI that provides an easy-to-use presentation of network information to users. Real-time network monitoring and control functions will play a crucial role in improving the communication reliability for functions that cannot be automated. These applications will be used to provide live monitoring metrics of the global network status to users and will be recorded for historical trends analysis that could be used to predict future failures.

The architecture of this proposed design is shown in Figure 26.

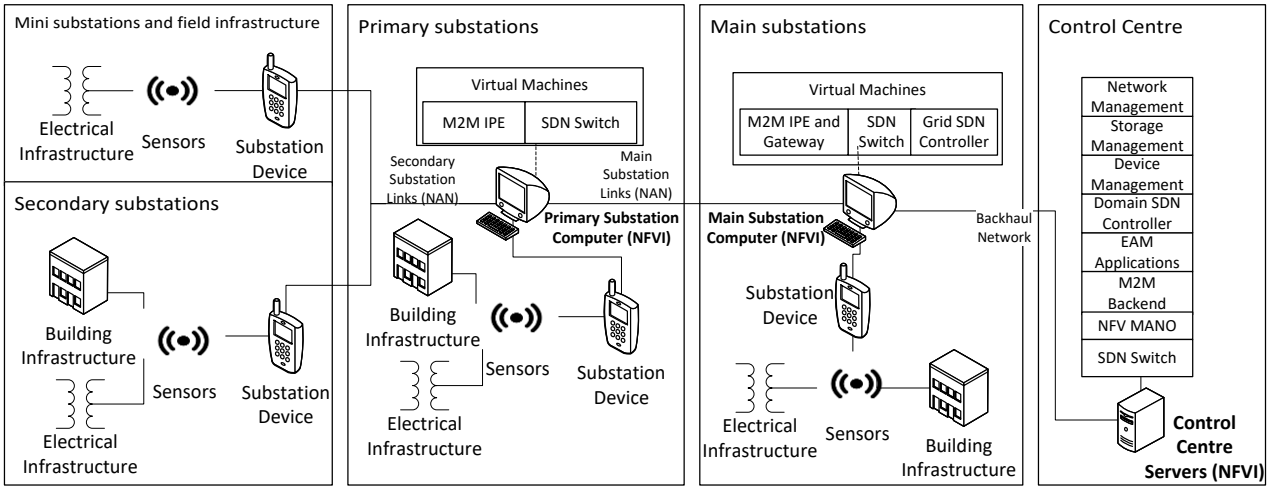


Figure 26: Architecture for a software-based network supporting an EAM system in a distribution grid.

3.3 Chapter Discussions

This chapter first introduced the requirements and design considerations for communication networks that aim to improve the reliability of distribution grids. Two methods, namely improving distribution grid reliability with a communication network that enables the functions required by an EAM system and improving the overall communication reliability of the network, were considered. Based on these requirements an architecture for a software-based networking platform that aims to improve the reliability of distribution grids was proposed. A process that will support the implementation of this architecture using a virtual networking system was also proposed along with a reference model for topology designs. These were then used to develop a proposed design for a software-based network that aims to support an EAM system supporting a section of the City of Cape Town's electricity distribution grid.

The following chapter will provide the implementation details of the proposed software-based network design. Details about the implementation platform, evaluation tools and testing procedures will also be discussed.

Chapter 4

Implementation and Evaluation Platform

The previous chapter presented the requirements and design considerations for communication networks that aim to improve the reliability of distribution grids along with a design for a software-based networking architecture that aims to address these requirements. A design was also proposed for a reliability focused communication network that aims to enable an EAM system in a section of the City of Cape Town's electricity distribution grid.

This chapter describes the implementation of the proposed communication network design that is based on the proposed architecture, using a virtual networking system. The network functions that were implemented to enable and support the required smart grid data acquisition functions and to improve communication reliability will also be described. These network functions were fast failover recovery, automated SDN controller failover, automated host failover, data buffering and traffic scheduling. Finally, the evaluation platform, evaluation tools and the testing procedures that will be used to evaluate the proposed design will be discussed.

4.1 Implementation

To evaluate the proposed design, a virtual network was implemented using a network emulation system hosted on a single desktop computer. The scope of this implementation was limited to the elements illustrated in the architecture diagram shown in Figure 27 and the elements as listed in Table 4.

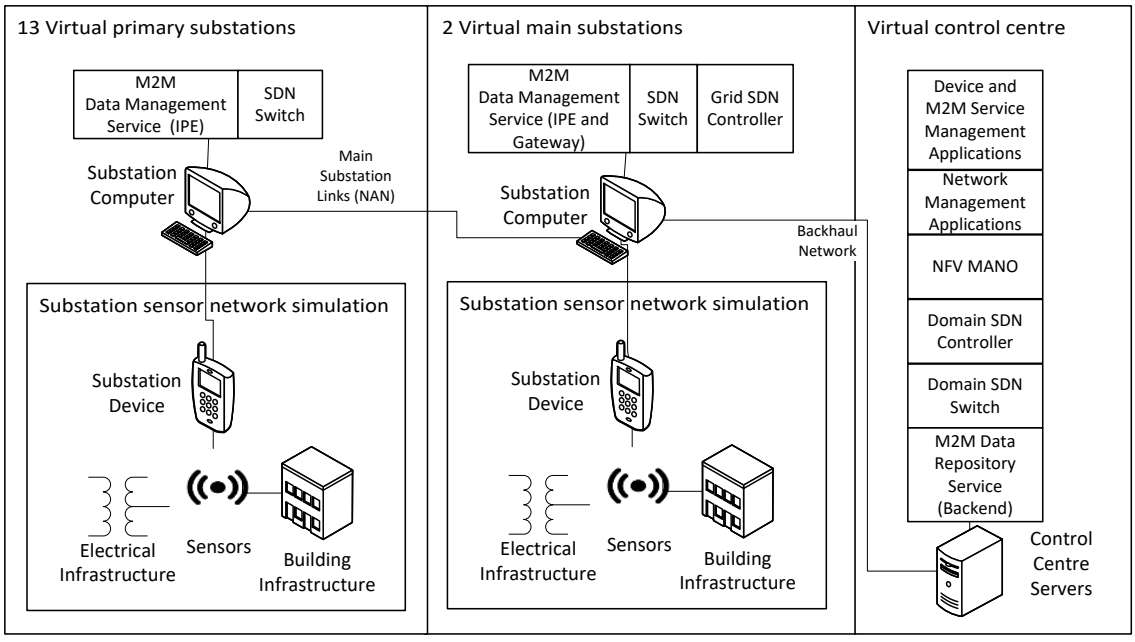


Figure 27: Distribution grid communication network architecture for implementation.

Layer	Scope
Infrastructure	Two main substations and 13 primary substations in the Green Point grid section with sensors fitted to their switchgear, transformers, and buildings.
Devices	13 sensors per substation connected to a smart substation device and a substation computer.
Virtual Devices	16 virtual network switches, three virtual SDN controllers and 16 virtual hosts providing network functions and services.
Network	Virtual wired Ethernet networks from substation devices and computers to control centre servers forming one NAN connected to a MAN.
Control	Network functions for fast failover recovery, automated SDN controller failover, automated host failover, data buffering and traffic scheduling implemented using SDN application interfaces and M2M services.
Applications	Substation data acquisition and monitoring implemented as functional applications. Network monitoring and control, VM management and M2M service management implemented as non-functional applications.

Table 4: Implementation scope.

As the focus of the design was to improve grid reliability, the implemented software-based communication network was set up with the intention of achieving the two design objectives highlighted in chapter 3. Firstly, the network had to enable and support the big-data acquisition functions required for an EAM system and secondly, the network had to utilise various implemented network functions to improve communication reliability. To evaluate the implemented network's ability to meet these objectives, initial tests were conducted to determine its ability to enable the automated transfer of simulated substation sensor data to a central data repository without any data loss or operational interruption. Further tests were then conducted to determine if the implemented network functions could automatically detect and recover from various simulated network faults and what effects this would have on the executing data acquisition functions.

The virtual network used in this implementation was created in a testbed on a single desktop computer using Mininet [76]. Mininet is a network emulation system that uses lightweight virtualization to create networks in a Linux VM. Mininet is therefore considered a virtual network orchestrator where a topology is programmatically defined in a Python script and implemented using the Mininet API. As part of the topology, Mininet creates instances of virtual SDN switches and connects them with virtual network links. Each switch instance is separately controlled by SDN controllers that integrate with Mininet and use the emulated network interfaces to exchange control messages. Mininet also uses process-based virtualisation to create multiple instances of virtual hosts in isolated network namespaces. These hosts have access to the underlying VMs applications and directory structures which makes it easy to use them to execute a variety of applications for network evaluation purposes.

The ONOS Tutorial VM Version 1.15.0 [77] was chosen for this testbed because it came preconfigured with Mininet, ONOS SDN controllers and various other applications needed for creating virtual SDN networks. A high-level architecture of the implemented testbed is shown in Figure 28.

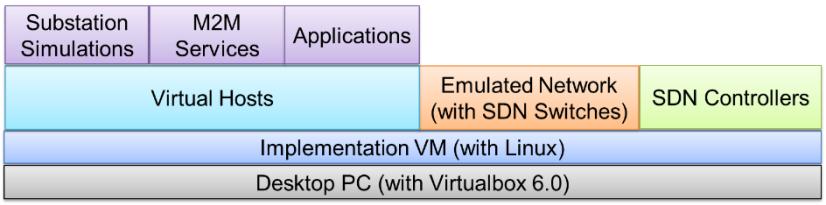


Figure 28: Implementation testbed architecture.

4.1.1 Distribution Grid Infrastructure and Devices

An assumption was made that the grid infrastructure and devices used for each substation in this grid section would be standardised, meaning each substation would consist of identical sensors, substation devices and substation computers that used the same communication protocols. This assumption allowed for the development of a Substation Simulation script which could easily be replicated each time a new substation was needed in the network design. The use of a Substation Simulation script eliminated the need to set up multiple physical interfaces to real devices, which streamlined the testbed set up and any subsequent adjustments to the implemented design. Since the focus of the evaluation was on supporting data acquisition functions and on improving communication reliability, the use of substation simulation scripts to generate the data needed for the planned tests would not impact the test results. Using the OpenMTC IPE-sensors demo application examples [78] as a reference, the Substation Simulation script was developed in Python to generate sensor data, acting as smart grid devices connected to various substation sensors. The simulation script was developed to execute the procedure shown in Figure 29.

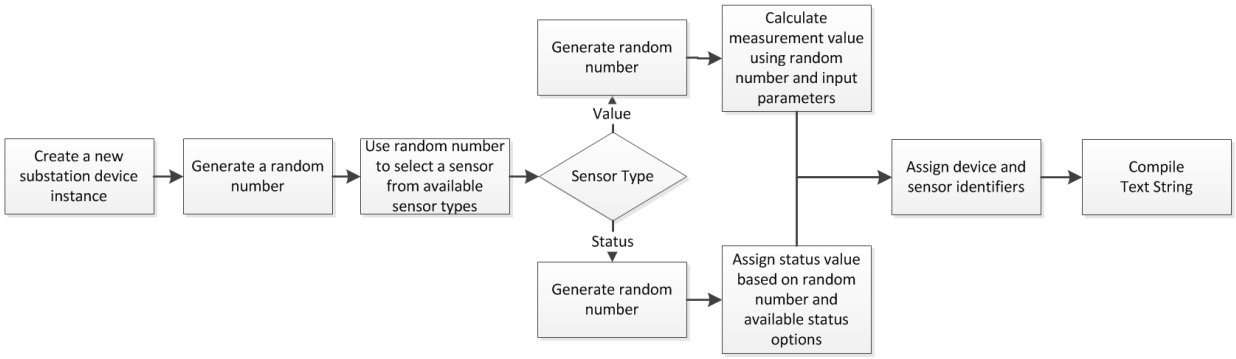


Figure 29: Procedure executed by the Substation Simulation script.

The script starts with creation of a new simulated substation device instance for which a random sensor is selected using a function generated random number and a predefined sensor list. This list reflects the types of sensors that were chosen, based on the information needed to combat the biggest risks to Cape Town's distribution grid reliability. The thirteen different sensors that are available to the simulation script are listed in Appendix F. Based on whether the selected sensor provides a value or status measurement, a second random number is then used to generate a new sensor reading using predefined parameters as inputs to a data generation function. These input parameters can be adjusted to support different scenarios. The generated

sensor reading is then added to a text string based on the data model presented in chapter 3, which includes the device's unique identifier, the sensor's unique identifier and a date- and timestamp. Each text string is stored temporarily in the simulated device's embedded memory.

The automated aggregation and transfer of these data sets to a central data repository was implemented using M2M service platforms. These services were also developed as Python scripts that included functions that performed data storage and organisation, data transfer using FTP communication, traffic scheduling and automated failovers to back-up devices. Two separate Python scripts were developed to implement these M2M services. The first script, called the Primary Substation Data Acquisition script was integrated with the Substation Simulation scripts and executed on 13 virtual hosts representing the primary substations in the Green Point distribution grid. Performing the functions of an M2M IPE, this script accesses the text string outputs generated by the substation simulation scripts directly as soon as it receives an indication that a text string is available. The IPE script then extracts the device's unique identifier from the text string and checks to see if the device that provided the string appears in a list of recognised devices. If the data are received from a new device, the script adds the new device to its list of recognised devices. It then uses the file management services of a virtual host representing a substation Gateway to request the creation of a new directory structure for the newly identified device. Next, the script differentiates between the value and status measurements and adds the received text string to one of two separate text files for future differentiation. These text strings are then continually added to text files until a preconfigured file size threshold is reached. Using event-based scheduling, reaching the threshold will trigger another service request to the substation Gateway service. This second request is for copies of its locally stored text files to be created in the relevant directory structures on the Gateway substation computer. After confirmation is received that the copies have been created successfully, the IPE service will delete its local copies to free up space for the next set of sensor readings. The file threshold counter will also be reset, and the data aggregation process will repeat.

The second Python script, called the Main Substation Data Acquisition script, is executed on the hosts representing the main substation computers with the purpose of scheduling and transferring any acquired data to a Backend repository hosted on the control centre servers where it can be accessed by EAM applications. In this script, time-based scheduling is used, which means a timer function is configured to wait a specific amount of time before each data transfer. Each time the timer reaches zero, the script establishes an FTP connection between the virtual hosts representing the M2M Gateway network and M2M Backend. The script then uses the file services of the Backend to compare its own directory structures to those of the Backend and then requests the creation of new structures where it identifies differences, thereby synchronising the two data sources. The script then requests that copies of the newly acquired text files on the Gateway be created on the Backend in its relevant directory structures. Upon confirmation from the Backend service that these copies have been created successfully, the script then uses its own file management functions to delete the local text file copies. The timer is then reset, and the data acquisition

steps are repeated. The procedures that these Python scripts are based on are shown in Figure 30 and Figure 31. An end-to-end procedural flow of all three integrated scripts is also provided in Appendix G.

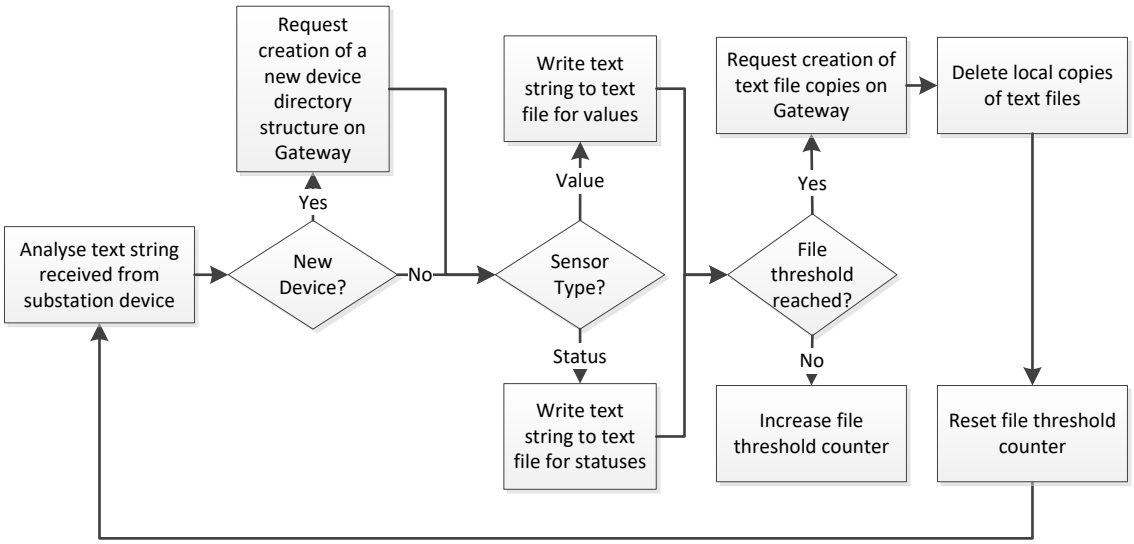


Figure 30: Procedure by the Primary Substation Data Acquisition script.

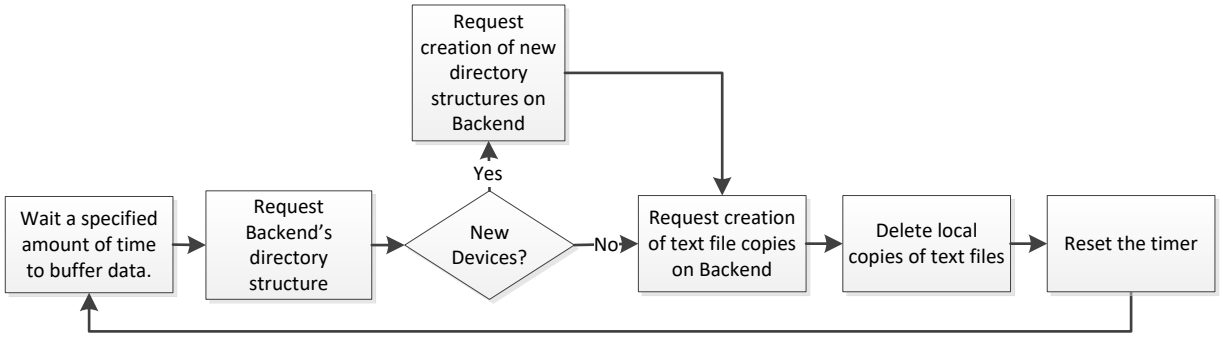


Figure 31: Procedure executed by the Main Substation Data Acquisition script.

In addition to the procedural steps described in these diagrams, each of the Python scripts include exception handling functions. One of these functions provides automated communication failover to a back-up host whenever a designated primary host is not available. These functions will ensure continual operation of the substation data acquisition functions in situations where destinations hosts were unreachable.

4.1.2 Networks

Mininet's emulated network links and virtual network devices were used to set up a network topology for the Green Point NAN and its backhaul connection to the control centre. The proposed network topology shown Figure 21 was created to connect 16 virtual hosts representing the NFVI in the 13 primary substations, the two main substations and the control centre. Sixteen virtual SDN switch instances were used, one for each substation and one for the control centre. Open-V-Switch (OVS) was chosen for these network

switches as one of the most popular open source SDN switches available that uses the OpenFlow protocol. Mininet allows emulated network links to be programmatically set up between any VMs created in the virtual environment. These links can also be configured programmatically to simulate real-world constraints such as bandwidth limitations. Considering the available network infrastructure in the Green Point area, a bandwidth limitation of 10 Mbps was set on all links in the Green Point NAN. The backhaul network links were restricted to 100 Mbps, considering a realistic bandwidth limit for available fibre networks. No latency restrictions were imposed on any network links because propagation delays were assumed to be negligible for the short distances between substations in the chosen grid section. As a more simplified means of setting up virtual networks in Mininet, a GUI based add-on program for Mininet called MiniEdit, was used for this topology setup. MiniEdit provides a user interface that simplifies topology creation with drag-and-drop interactions, drop-down menus, and pop-up windows, as an alternative to topology creation through Python programming. MiniEdit was developed in Python and works by generating its own dynamic Python scripts based on the user's created topology configuration. MiniEdit also provides an interface to the Mininet API that allows users to interact with the live networks which simplifies live network testing and failures simulation.

Three SDN controller interfaces were also added to the network topology, with the two main substations each receiving a distribution grid controller. The domain controller interface was linked to the control centre's domain level switch. These SDN controller interfaces were used to establish connections to SDN controller instances running as three separate VMs. Each SDN controller was set up so that it could share control of the network switches with the other two controllers. The open source ONOS SDN-controller was chosen for this implementation because ONOS was specifically developed for WAN applications [79]. ONOS controllers have demonstrated high performance, control plane resilience and scalability making them suitable for smart grid applications. ONOS also uses OpenFlow for southbound communication and REST APIs for northbound interfaces. Another reason why the ONOS controller was selected is because of its active development community and its library of community developed applications that support a variety of network functions [41].

4.1.3 Control, Management and Orchestration

Control of the communication network was implemented through application interfaces to the SDN controllers. Because multiple SDN controllers were used in this design, all three SDN controllers had to be synchronised for which a co-ordination and orchestration application was needed. Atomix, which is an event-driven framework for co-ordinating fault-tolerant distributed systems, was used for this function [80]. Atomix uses APIs for sharing mission critical states and coordinating state changes in distributed systems, making it ideal for coordination between stand-alone SDN controllers. For each of the SDN controllers, applications that would automate network discovery, the set-up and updating of the shortest flow paths between all discovered network elements, the reactive and proactive fast failover recovery and the packet flow monitoring were executed. Applications that were developed by the ONOS community were used for

these functions as they easily integrated with the ONOS controller. Two applications were also implemented to allow users to manually control the network via instructions to the SDN controllers and to present the SDN controller's monitoring information to users for analysis. The ONOS command line interface (CLI) was one of these applications that allowed users to issue instructions to SDN controllers as text commands. These commands can be used to update flow tables or to issue network queries. The ONOS GUI was implemented as a more user-friendly application that compliments the ONOS CLI. The ONOS GUI can present real-time information about the discovered network topology and traffic flows as animated graphical topologies. It also provides a variety of network reports, including detailed lists of discovered network elements, active services, and flow statistics.

In addition to Mininet, which was used as a NFV MANO, a Python software development environment (SDE) was added to the testbed. The Python SDE was used to develop and execute the scripts used for the substation simulations of M2M data management services. An FTP client-server application was also added for use by the virtual hosts. EAM applications that would utilise the acquired substation data were not implemented in this testbed as this was out of scope for this work. An overview of the implementation platform configuration is shown in Figure 32 and a mapping of the platform components to the proposed architecture components presented in chapter 3 is provided in Table 5. Details about accessing an online repository that contains the files required to implement this testbed, are provided in Appendix H.

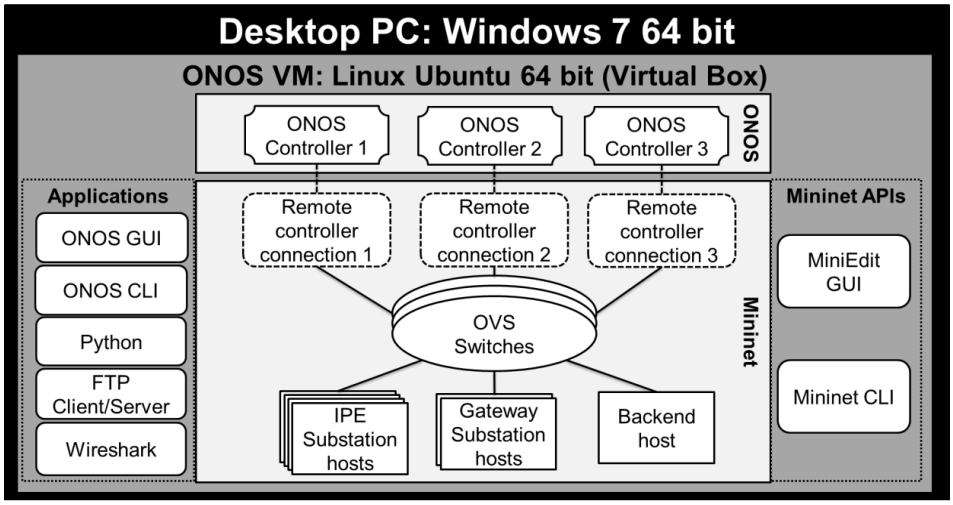


Figure 32: Testbed setup.

Software-based networking architecture component	**Platform implementation**
Distribution grid infrastructure, sensors, and devices	VMs running Python scripts for substation simulations
M2M IPEs, Gateways and Backend	VMs running Python scripts for traffic scheduling and data aggregation
SDN Switches	VMs hosting OpenVswitch instances
SDN Controllers	VMs hosting ONOS controller instances
Network Management Applications	VM hosting ONOS GUI, ONOS CLI and Wireshark
Device and M2M Service Applications	VM hosting a Python SDE
NFV MANO	VM hosting Mininet and Atomix

Table 5: Mapping of proposed architecture to testbed implementation.

4.2 Evaluation Tools and Test Plan

To evaluate the implemented communication network, tools were needed that could monitor communication during normal operating conditions and during fault conditions that threaten the reliability in the distribution grid. The ONOS GUI was chosen as the main evaluation tool for overall communication monitoring because the real-time information on the network status, presented in a user-friendly topology layout, provides a good high-level overview of the network. The network packet analyser Wireshark [81] was also installed in the testbed and was used for detailed packet monitoring and packet dissection. To observe the effects of simulated communication problems on the data management Python scripts that enabled the data acquisition functions, these scripts were set up with status logging messages that informed users of each function the script executed and alerted them to any problems with the execution of these functions. The Mininet API was used to simulate various communication failures by administratively disabling specific network interfaces. SDN controller failures were simulated with commands entered via their ONOS CLI. To simulate large amounts of network traffic that resulted in network resource overloading, the parameters of the Substation Simulation scripts were adjusted to increase the size of the text files beyond the maximum available bandwidth of the network. A stopwatch was also used to measure the time required to perform certain manual activities.

Using these tools, several tests were performed to evaluate both the implementation platform that aims to streamline the network development process and the network that aimed to improve grid reliability. The results of each test were recorded for analysis. The first test focused on evaluating whether the proposed Green Point communication network design could be implemented on a single desktop computer and how long it would take to prepare such a network. The second test evaluated the implemented network's ability to support automated big-data acquisition functions required by an EAM system and the capability of the implemented M2M data management service to enable these data acquisition functions. The last five tests evaluated the functionality of the implemented network functions and M2M services that were implemented with the objective of improving communication reliability. The testing procedures that were followed for each of these tests will now be discussed.

4.2.1 Streamlined Network Implementation

In the first test, the time required to perform the network preparation tasks was measured to get a general idea of the effort involved in setting up a software-based network for a section of a distribution grid. These tasks were: setting up the network topology, preparing the SDN controllers, instantiating the network, discovering the entire network topology, calculating, and installing the optimal network flow rules and activating the required network functions and services. The network preparation was broken down into three stages. Stage one focused on the core network consisting of three virtual hosts, three SDN switches and three

remote SDN controller interfaces which were set up in MiniEdit. Three ONOS controller instances were also instantiated with the launch of the ONOS CLI and ONOS GUI applications as well as the required ONOS services. Stage two focused on the development of the Greenpoint NAN consisting of 13 switches and their respective network links, while stage three focused on setting up the 13 edge network hosts. For each stage, the developed network was instantiated in Mininet and the ONOS GUI was used to monitor the automated network discovery and flow rule updates. Console windows were also launched for the execution of the developed data acquisition services. The time measurements for each stage and task were summed and the total network preparation time was calculated. The criterion for success in this test was a software-based network prepared on a desktop computer to a point where it was ready to support a smart grid application and for this network to be monitored in real-time.

4.2.2 Enabling Smart Grid Functions

For the second test, the goal was to use the implemented network to transfer the data generated by the substation simulations, ensuring that these data sets ended up in the correct directory structures in the grid's central data repository. These transfers had to make use of the implemented M2M data management services and had to occur without any human intervention or any unplanned interruptions. This test was initiated with the execution of the Substation Simulation and the Primary Substation Data Aggregation script on the 13 primary substation hosts. The file threshold parameters for these scripts were set to 500 readings per file. The Main Substation Data Aggregation script was then executed on the main substation hosts with their timer parameters set to delay data transfers for 5 seconds each time new files were acquired. As these scripts were executing, their feedback messages were monitored on terminal windows that were launched for each substation host. The file management activities performed by the M2M services were also monitored using the Linux file explorer and the network activity was monitored using the ONOS GUI.

4.2.3 Improving Communication Reliability

The criteria for success for the last five tests were as follows: After a specific communication problem was simulated, each of the implemented network functions had to trigger automatically, based on the respective problems they were implemented to respond to. The network functions then had to perform the required actions that would limit the impact of these problems on communication and execution of the data acquisition applications. Simulated failure scenarios included: a link failure, a node failure, a SDN controller failure, a Gateway host disconnection, a substation disconnection, and a NAN isolation. Execution of these functions also had to occur without any data loss or interruption to the data acquisition process. In testing the data buffering network functions, some unavoidable delays in data transmission would be allowed, if communication resumed automatically, once the network problem was removed. Execution of the traffic scheduling functions had to result in improvements in round trip latency (RTL) for packets transferred on overloaded network flow paths.

For each of these tests a primary substation host was configured to execute the Substation Simulation script as well as the Primary Substation Data Aggregation script. This would result in data transfers between the primary substation host and its designated main substation Gateway host at regular intervals. The Main Substation Data Aggregation script was also executed to transfer acquired data to the Backend host at scheduled intervals. To simulate a link failure, the link between the two switches in an active flow path was disabled. A node failure was simulated by disabling a specific network switch in an active flow path. Recovery from both the link and node failure had to be managed by the implemented FFR functions. The disconnection of specific hosts was simulated by disabling the links to these hosts. The host failover functions implemented in the Python scripts had to manage recovery from a disconnected Gateway host and the data buffering functions had to ensure no data was lost while an IPE host was disconnected from the network. The buffering functions also had to prevent data loss during a simulated NAN isolation while the backhaul network link was disabled. All these failures were simulated using the Mininet API. A SDN controller failure was simulated by issuing the "shutdown" command to one of the ONOS controllers using the ONOS CLI and the remaining back-up SDN controllers had to ensure this failure was managed automatically. For each simulated failure, network activity was monitored using the ONOS GUI and the effect on the running data acquisition scripts was monitored through feedback messages from the executing Python scripts in the host's terminal windows. Figure 33 illustrates the various points in the network where failures were introduced to test these network functions.

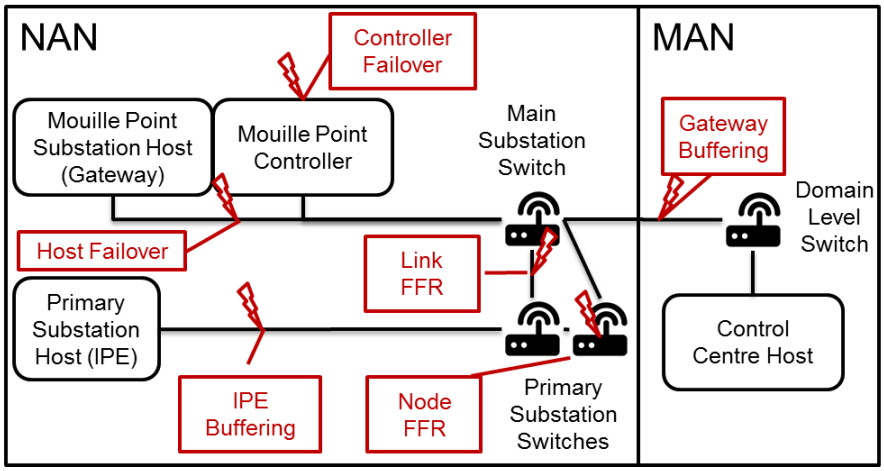


Figure 33: Simulated failure points in the implemented network.

Overloading of the network resources in the active flow paths was simulated by adjusting the parameters of the Substation Simulation scripts to allow storage of 100 000 sensor readings per text file before each transfer. The large number of readings would result in text files that were each over 10 MB in size which meant that, with the 10 Mbps bandwidth limit on the NAN links, network congestion would quickly build up in these flow paths. The scheduling parameter was then adjusted to delay transmissions for 10 seconds. To observe the effect of this congestion on the RTL of the transferred packets, a terminal window was launched on another substation host and a ping command was executed on it. This commanded the host to send a

series of short messages to another host using the shared and congested network resources. As each message required a reply from the other host, it allowed this host to act as another smart grid application that utilised short, time-sensitive messages. The recorded time stamps for each sent ping message and each received reply was used to calculate the RTL for each ping message.

A summary of the test plan followed in the evaluation of this implemented design is provided in Table 6.

<table>
<tr><th>Test</th><th>Objective</th><th>Expected Outcome</th><th>Metrics</th></tr>
<tr><td>1</td><td>Set up a software-based communication network, modelled after the Green Point distribution grid and measure the amount of time required to do so.</td><td>A software-based network capable of supporting smart grid functions is prepared on a desktop computer in less than an hour.</td><td>- Network setup time.</td></tr>
<tr><td>2</td><td>Use the implemented software-based network and its implemented M2M services to transfer simulated sensor data from substation hosts to a host representing a central data repository.</td><td>All generated data reach the correct directory structures in the grid’s central data repository without the need for user intervention in the acquisition process.</td><td rowspan="8">- Percentage of measurements not reaching the data repository.

- Amount of data acquisition process interruptions.

- Total avoidable delay time in the data acquisition process.</td></tr>
<tr><td>3</td><td>Recover from network link and node failures using FFR functions.</td><td rowspan="5">Automatically detect and recover from the failure without interruptions and with no avoidable delays in the data acquisition process.</td></tr>
<tr><td>4</td><td>Recover from a SDN controller failure using automated controller failover functions.</td></tr>
<tr><td>5</td><td>Recover from a Gateway disconnection using automated host failover functions.</td></tr>
<tr><td>6</td><td>Prevent data loss during a substation disconnection using data buffering functions.</td></tr>
<tr><td>7</td><td>Prevent data loss during a NAN isolation using data buffering functions.</td></tr>
<tr><td>8</td><td>Reduce delays caused by overloaded network resources by using traffic scheduling functions.</td><td>Improve the RTL of other packets sent on the overloaded flow paths.</td><td>- Round-trip latency (RTL).</td></tr>
</table>

Table 6: Summarised test plan.

4.3 Chapter Discussions

This chapter provided a description of the implementation of the design proposed in chapter 3, in a platform consisting of a network emulation system. The implementation platform was described along with some of the computer scripts that were developed as part of this work. The evaluation tools and test plan that were used to evaluate the communication network were also described.

The next chapter will present the results obtained from these tests. The implemented communication network will be evaluated in terms of its functionality and its ability to meet the listed test objectives. The network’s ability to meet the requirements specified in chapter 3 will also be discussed.

Chapter 5

Results, Analysis and Verification

The previous chapter presented the implementation of a software-based network, based on a model of a section of the Cape Town distribution grid. The tools and procedures that were used to evaluate the implemented design were also discussed. This section provides the evaluation results, an analysis of these results and verification of the implemented design's ability to meet the requirements of a communication network that supports distribution grid reliability.

5.1 Streamlined Network Implementation

The first test focused on answering the question of whether a software-based communication network based on the design proposed in this dissertation could be implemented on a single desktop computer. It also considered how long it would typically take to set up such a network. A stopwatch was therefore used to measure the time required to perform each of the network preparation steps. Firstly, the network topology shown in Figure 34 was created in MiniEdit, starting with the backhaul network.

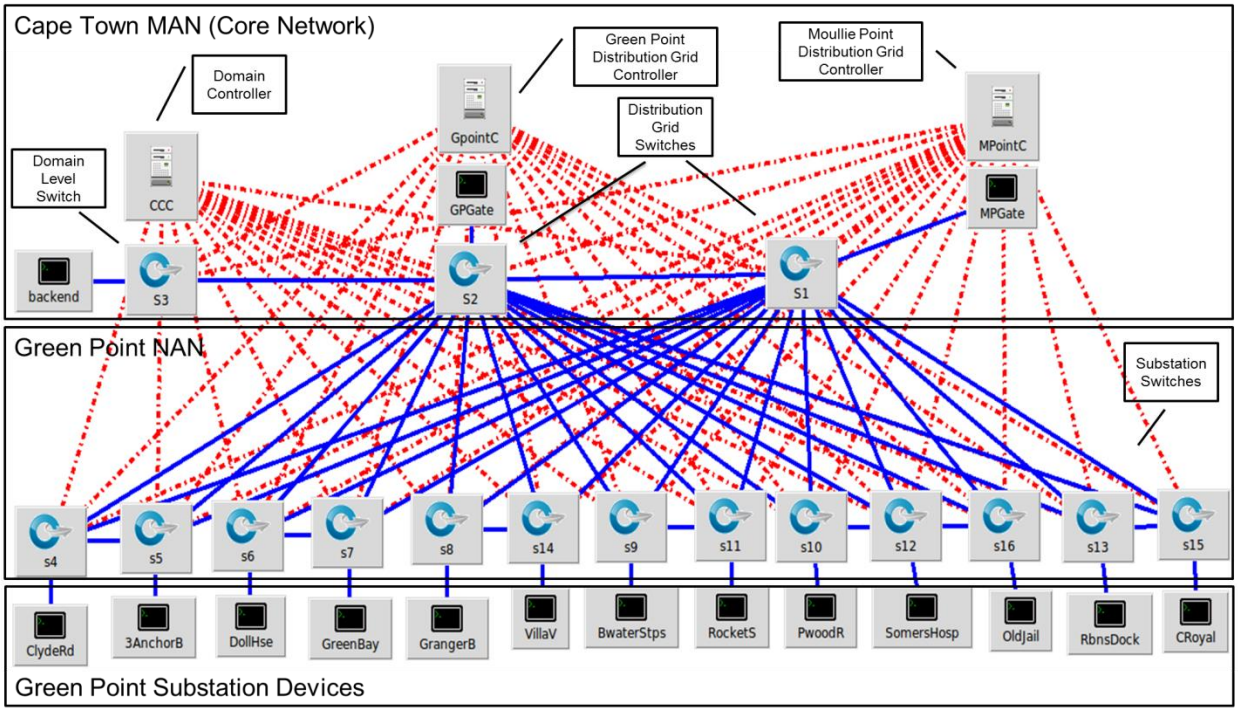


Figure 34: Network topology created in MiniEdit and instantiated in Mininet.

The Green Point NAN was mapped-out next, followed by the substation hosts that were placed and connected to their respective substation switches. Finally, three ONOS controller instances were initiated and the required controller functions were launched. The ONOS GUI, which was executed after the ONOS

controllers reported successful activation, then reported that it was monitoring the three controllers. It also reported each controller's IP address, which was then used in MiniEdit to set up the controller interfaces for the network.

As soon as the "run" command was issued, MiniEdit compiled and executed the Python script needed to instantiate the designed network in Mininet. The ONOS GUI then reported the discovery of the created network switch instances, the network links and identified flow paths. The switches were presented in a graphical topology view with boxes of different colours indicating the mastership of each switch by one of the three controller instances. Network links between these switches were depicted as grey lines between these boxes that changed colour depending on the amount network traffic present on a link at any given time. The amount of network traffic on each link was also reported in real-time. Using the Mininet CLI, each host was commanded to make itself known to the network with a ping message. With each ping message, the ONOS GUI reported an increase in the number of discovered hosts and the number of discovered flow paths. These hosts were depicted on the topology as a circular icon with the host's IP address below it. The full discovered network topology reported in the ONOS GUI, is shown in Figure 35.

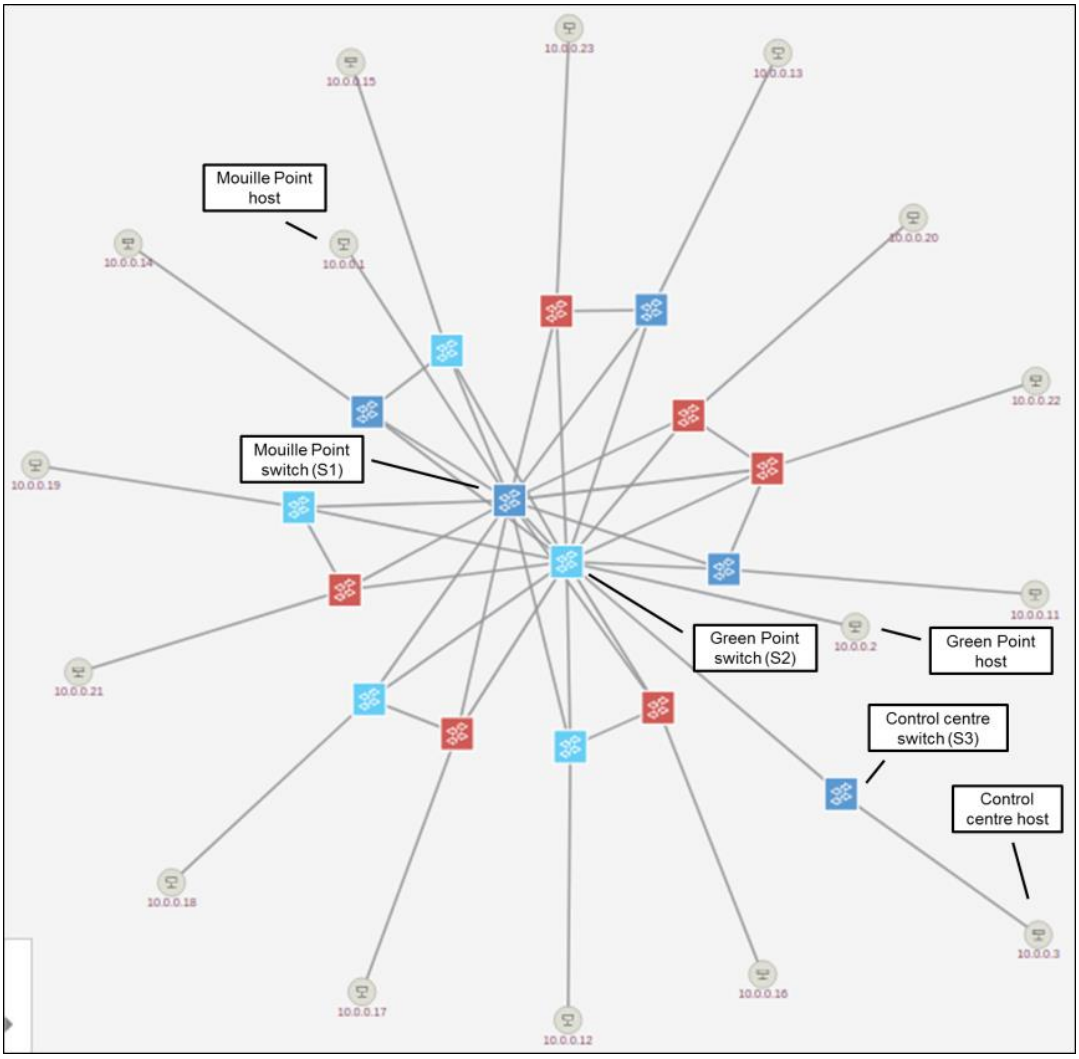


Figure 35: ONOS GUI reporting discovered network topology.

To activate the developed M2M services, a console window for each host was launched and the respective Python scripts for each host were executed in these windows. The implemented network was now ready to support automated data acquisition. The recorded time measurements were then summed for each of the network setup phases, as well as each setup step. These values were then summed to calculate the total network implementation time which is shown in Table 7.

Network setup phase	**Network set up and controller preparation**	**Network instantiation (automated)**	**Discovery and flow path optimisation (automated)**	**M2M service activation**	**Total**
2 Main substations and control centre	7 min 31 sec	1 sec	3 sec	2 min 22 sec	9 min 57 sec
13 Primary substation switches	4 min 42 sec	3 sec	3 sec	N/A	4 min 48 sec
13 Primary substation devices	3 min 55 sec	4 sec	4 sec	4 min 40 sec	8 min 43 sec
Total	16 min 8 sec	8 sec	10 sec	7 min 2 sec	**23 min 28 sec**

Table 7: Time measurements for setting up a communication network on a desktop computer.

From these measurements, the biggest contributors to the total network implementation time were the manual activities of setting up the network in MiniEdit, instantiating the SDN controllers and preparing the required network functions and services. The use of a GUI application that automatically compiled and executed the code needed to instantiate the network in Mininet, and a script that automatically instantiated the three SDN controllers with their integration to Atomix streamlined these steps to a certain degree. The network setup, controller preparation and network instantiation therefore only contributed 16 minutes and 16 seconds to the total development time. This could have taken longer, had each of these tasks been performed manually. Activation of the developed M2M services took a total of 7 minutes and 2 seconds, mainly since a total of 16 console windows had to be launched with the execution command entered manually for each host. Automation using a script could probably reduce the time required to perform this step, but these sorts of improvements will be left for future work. Network discovery and flow path optimisation using the built-in ONOS functions only contributed 10 seconds to the total implementation time.

The time required to set up a software-based communication network for a smart distribution grid will depend on a lot of variables, especially since all steps in the process cannot be automated. The 23 minutes and 28 seconds it took to set up this network is however a good general indication of the time required to set up a distribution grid communication network consisting of three core network connections and 13 NAN network connections. These results also support the findings of other research that motivates the use of software-based networking to improve communication network development over conventional hardware-based networking. Implementing a similar network using conventional approaches would likely have taken hours, if not days due to reliance on many manual steps and the need for hardware procurement. Most importantly, these results demonstrated that a software-based network, designed according to a model of a

real city distribution grid section, can be implemented in a virtual environment hosted on a desktop computer.

5.2 Enabling Smart Grid Functions

The second test focused on answering the question of whether a software-based network can provide the communication functions and services needed to support an EAM system that would improve the reliability of a distribution grid. To answer this question, the ability of the implemented network and its M2M service platforms to enable automated data acquisition functions was tested. Using the network developed in the previous test, this test was initiated after the Substation Simulation script and the Primary Substation data aggregation script was executed on the 13 primary substation hosts. As soon as these scripts were executed, the creation of text files could be observed in the respective directories of each of these hosts using the Linux file explorer. The creation of copies of these text files in the directory structures for the Mouille Point Gateway substation was also observed. If a directory for a specific substation did not exist, it was created automatically by the M2M service platforms. A snapshot of the transferred text files appearing in the Clyde Road substation's directories is shown in Figure 36.

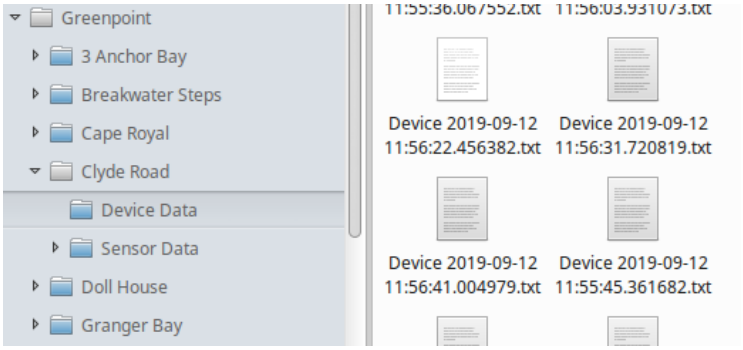


Figure 36: Example of transferred text files in the Gateway directories.

Opening one of the text files showed a series of text strings that contained the data generated by the Substation Simulation script. An example of one of these text strings created for a primary substation called Rocket Store is shown in Figure 37.

```
Rocket Store: VoltsB: {'timest': '2019-08-29 11:34:38.298937', 'type': 'Voltage', 'unit': 'V', 'value': 403}
Rocket Store: VoltsW: {'timest': '2019-08-29 11:34:38.319683', 'type': 'Voltage', 'unit': 'V', 'value': 414}
Rocket Store: VoltsR: {'timest': '2019-08-29 11:34:38.340802', 'type': 'Voltage', 'unit': 'V', 'value': 415}
Rocket Store: Humid: {'timest': '2019-08-29 11:34:38.361566', 'type': 'Hmudity', 'unit': 'g/m3', 'value': 30}
Rocket Store: AmpsB: {'timest': '2019-08-29 11:34:38.382559', 'type': 'Current', 'unit': 'A', 'value': 10}
Rocket Store: VoltsR: {'timest': '2019-08-29 11:34:38.403427', 'type': 'Voltage', 'unit': 'V', 'value': 415}
```

Figure 37: Example of a generated value measurement text strings in a transferred text file.

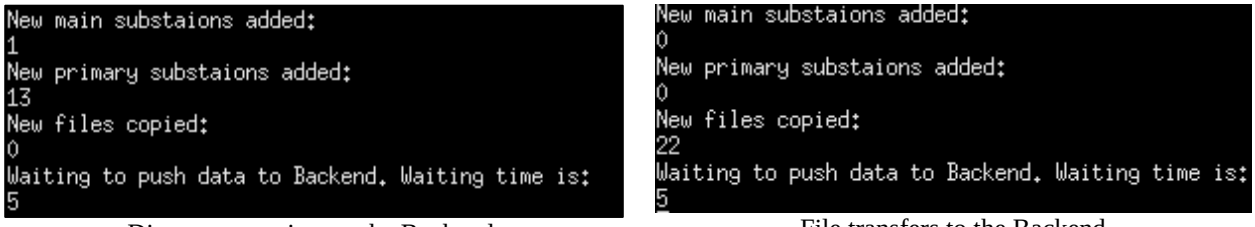
Messages that were displayed in the console windows that were used to execute the Python scripts also provided feedback on successful execution of these operations. Three examples of these messages are shown in Figure 38. They include messages indicating the generation of data by the Substation Simulation script,

directory creation on the Mouille Point host and successful data transfers followed by deletion of the local text files by the Primary Substation Data Aggregation script.

Data generation | Directory creation on Gateway | Data Transfer to Gateway

Figure 38: Messages displayed by Python scripts executing on primary substation hosts.

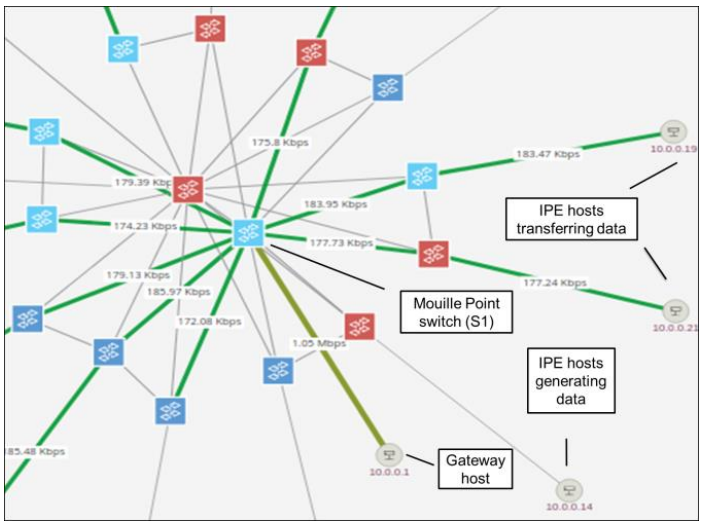
The data acquisition process required the transfer of all generated data, aggregating in the main substation hosts' directory structures, to the control centre host via the backhaul network links. To enable these transfers, the Main Substation Data Aggregation script was executed on the Mouille Point substation host with the scheduling parameter set to transfer files every 5 seconds. After this Python script was executed, feedback messages on the console window indicated that one new main substation and 13 new primary substations were identified from the Mouille Point substation host's directory structures. The script also confirmed that the relevant directories were created in the control centre host's directories after a successful FTP connection was established. After 5 seconds, the console window reported that the text files stored in this Mouille Point substation host's directories were now copied to the directories created on the control centre host. Further inspection confirmed that the Mouille Point substation's directories were now empty and that all copies of these files now resided in the control centre host's directories. The feedback messages displayed in the Mouille Point Gateway host's terminal window are shown in Figure 39.



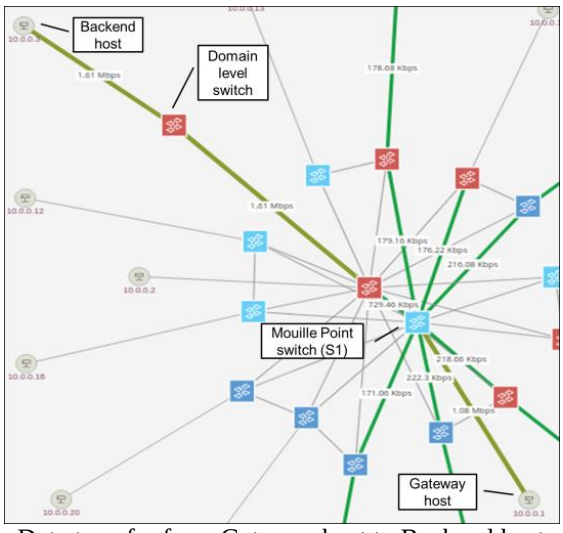
Directory creation on the Backend | File transfers to the Backend

Figure 39: Feedback messages displayed as a Gateway host transferred its data to the Backend host.

Further verification of the FTP communication used by the implemented M2M services was provided by the real-time monitoring functions offered by the ONOS GUI. Figure 40 shows two snapshots of the network traffic during the execution of these Python scripts. In the first, several hosts were generating new measurements while the others were transferring their text files to the Gateway host with IP address 10.0.0.1. This led to an increase in network traffic on the link between the Gateway host and its switch. The second snapshot shows data being transferred from the Gateway host to the Backend host with IP address 10.0.0.3. This transfer resulted in another big increase in network traffic on the links between these two hosts.



Data transfer from IPE hosts to Gateway host



Data transfer from Gateway host to Backend host

Figure 40: Snapshots of the network traffic during data acquisition process.

The data acquisition process executed for a period of 5 minutes before the Python scripts were stopped. After this test several new directories were created in the Mouille Point Gateway and the control centre host's directory structures while the control centre host's directories contained all the data generated by the substation simulation scripts. This data acquisition occurred without the need for any human intervention and no exceptions or errors were reported in any of the executing scripts during this test. The results of this test therefore demonstrated that the implemented software-based network can provide the communication functions and services needed to support and enable an EAM system.

5.3 Improving Communication Reliability

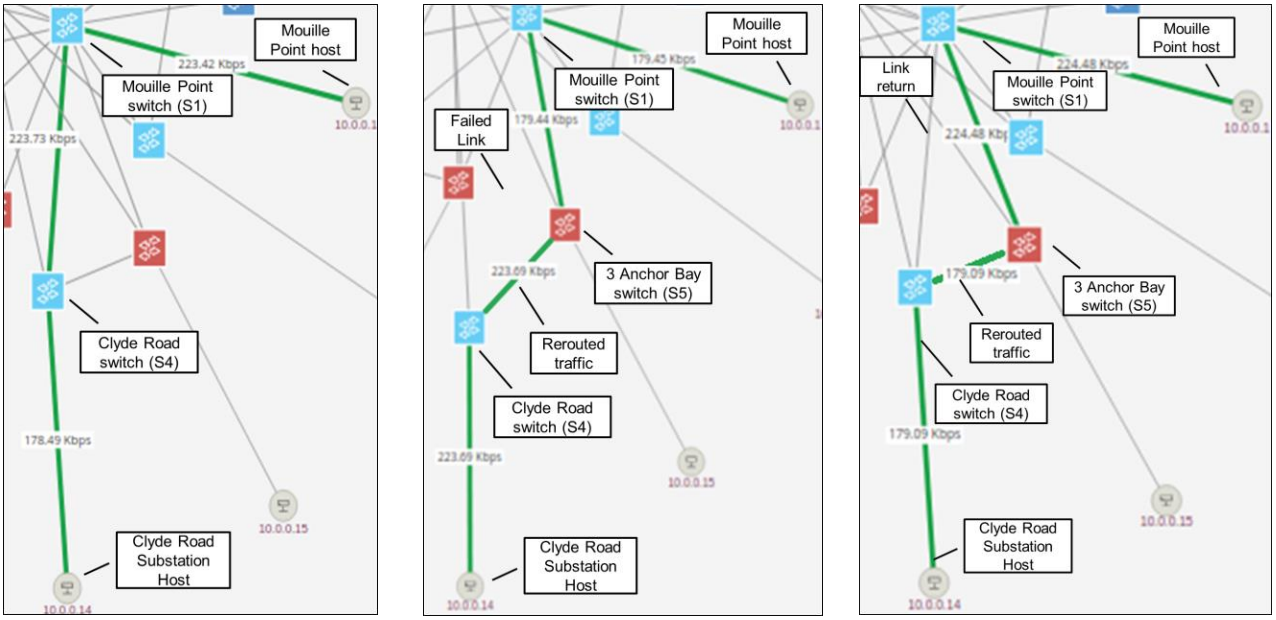
The final set of tests aimed to answer the question if the network functions and M2M services implemented in this design can improve the communication reliability of the modelled distribution grid. These tests focused on evaluating the ability of these network functions and services to reduce the probability of communication failures and communication delays, while reducing the mean time to repair communication problems. The functions that were tested included FFR, automated SDN controller failover, automated host failover, data buffering and traffic scheduling. To test these network functions, a network was subjected to a series of failure simulations from which it had to recover automatically without interrupting the data acquisition process. All these tests were conducted with the implemented network used in the previous two tests.

5.3.1 Fast Failover Recovery (FFR)

A link failure was simulated between two switches in an active flow path between two communicating hosts using Mininet "link down" command. As soon as the link was disabled, the ONOS GUI reported that an ONOS controller detected the link failure and that all network traffic on the flow path with the failed link had been rerouted to an alternative flow path. The failed link and rerouted traffic were depicted on the

network topology with an animation of the failing link turning red and disappearing, followed by an increase in network traffic on an alternative link. During and after this failure and rerouting, no errors or interruptions were reported by the executing Python scripts.

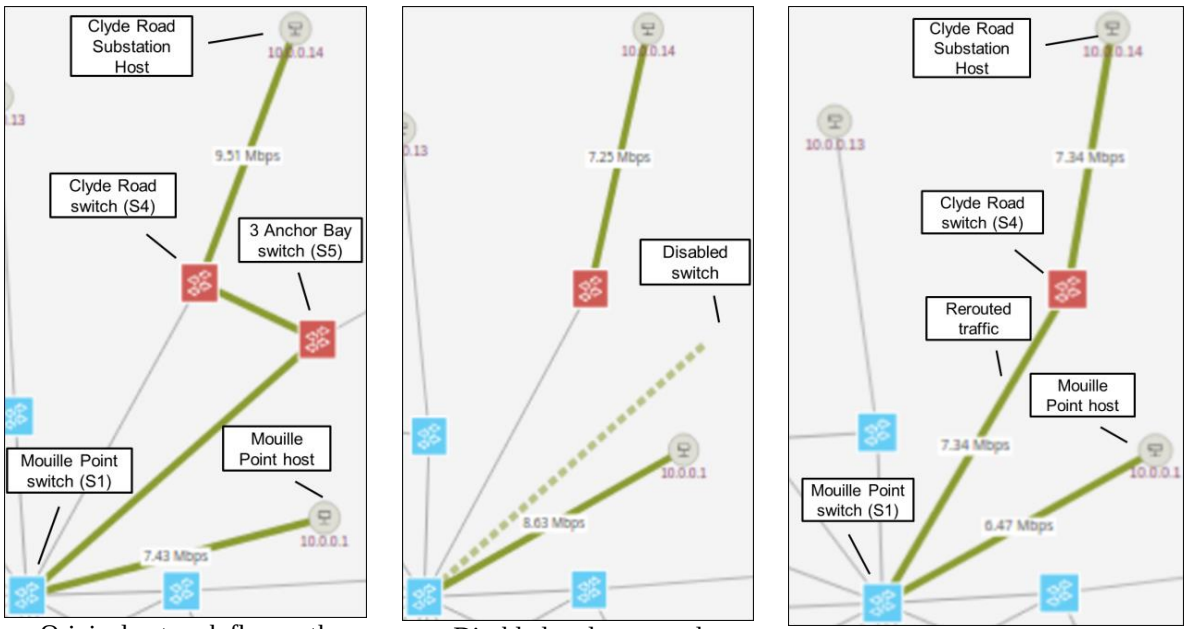
After a period of two minutes the disabled network link was enabled again. The ONOS GUI reported discovery of the re-established link, but traffic flow did not revert to the original flow path immediately. Only after a couple of seconds, did the flow correct itself to the shortest path again. Figure 41 shows three snapshots of network traffic on the original flow path, the rerouted traffic flow after a link failure and traffic flow after the failed link was re-established.



Original network flow path | Alternative flow path after link failure | Retained traffic flow with link enabled again

Figure 41: Snapshots of monitored link failover recovery.

Similar results were observed when a node failure was simulated by disabling a network switch. After the switch was disabled, the ONOS GUI reported the node failure and the automated rerouting of impacted traffic to an alternative flow path. The rerouting also occurred rapidly with no errors or interruptions in the executing Python scripts. When the node was re-enabled after two minutes, the ONOS GUI reported a topology change caused by the returning node. Again, traffic flow did not revert to the original flow path but stayed on the shortest though its local switch. The snapshots in Figure 42 show the initial traffic flows, the detection of the node failure and the rerouting of traffic reported by the ONOS GUI during this test.



Original network flow path | Disabled node reported | Rerouting on alternative flow path

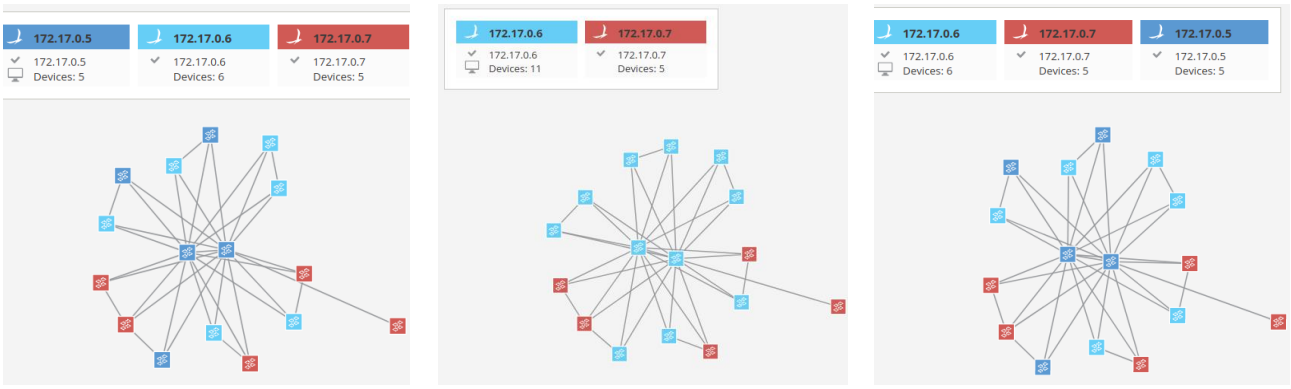
Figure 42: Snapshots of monitored node failover recovery.

Both FFR tests were repeated 3 times and each time similar results were obtained. Because proactive FFR functions were implemented in this network, these failovers occurred without the need for the switches to communicate with the SDN controllers. This meant that the failovers executed within a few milliseconds each time. When a switch detected that one of the links connected to one of its ports was not reachable, it utilised its group tables to look up the relevant alternative flow and then diverted traffic to another available port. Because there were no functions implemented that forced the SDN controllers to automatically optimise the network's flow paths at regular intervals, these switches had to wait for their flow entries to expire. This triggered a request to the SDN controller for new flow entries, which it then provided based on the calculated shortest paths that diverted traffic flows if needed. With each node failure, there was a potential risk of data loss since these switches could be disabled while they were buffering and processing packets. This would result in the packets not reaching their destination. Fortunately, the communication protocols used in this network provided for retransmission requests when packets were not delivered within a specific time. Since the data transmissions for the implemented functions were not time-sensitive, these retransmissions would not cause the data acquisition applications to fail.

5.3.2 Automated SDN Controller Failover

The implemented network's ability to handle SDN controller failures was tested next by simulating such a failure with a "shutdown" command issued through the ONOS CLI. This command instructed the SDN controller with IP address 172.17.0.5 to switch itself off. As soon as the command was issued the ONOS GUI reported the missing controller and that the mastership of five impacted switches had changed to the controller with IP address 172.17.0.6. This failover to an alternative SDN controller was indicated on the topology with colour changes on the icons of the impacted switches. About 15 seconds after the ONOS controller was disabled, the ONOS GUI reported the automatic reactivation on the controller and its

reconnection to the network. The controller now had no SDN switches assigned to it to control. By issuing a mastership normalisation command, mastership roles for all switches were reallocated fairly across all available switches. Throughout this test, the executing data acquisition scripts continued to function without any interruption. Three snapshots of the ONOS GUI topology taken during this test are shown in Figure 43. They show the controller mastership allocation before the failure, during the failover and after the controller was rebooted and the mastership normalisation command was issued.



Original ONOS controller configuration

Reassigned mastership roles after controller failure

Rebooted controller with normalised mastership roles

Figure 43: Snapshots of monitored controller failure recovery.

These results show that the implemented network was capable of automatically recovering from a SDN controller failure without impacting the execution of data acquisition functions. This can be attributed mainly to the fact that the switches in this network had their flow entries preconfigured by the SDN controllers. For the brief instance where a SDN controller was not assigned to some of the switches in the network, they simply continued to forward their packets as per their current flow entries. The failover to an alternative ONOS controller witnessed in this test was managed by the Atomix application when it detected the offline controller and it automatically triggered the remaining SDN controllers to take over mastership roles of the unassigned switches. This resulted in updates to the switches that caused their control and monitoring messages to flow to their newly allocated controller. When the disabled ONOS controller came back online, it reconnected with the emulated network automatically through the created Mininet interface that was still active. The backup controllers however retained mastership roles of these switches since no automated controller normalisation functions were implemented, which is why a manual normalisation command had to be issued.

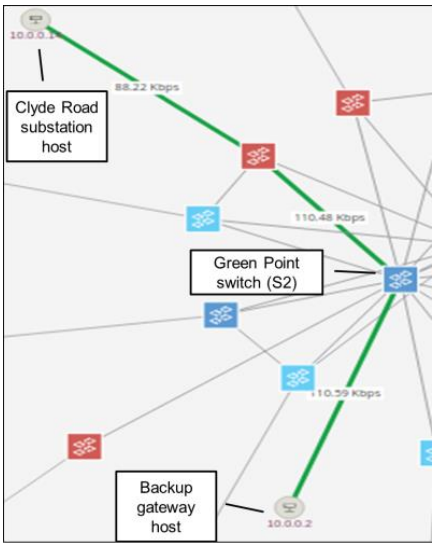
These results show that the risk for communication failure and communication delays caused by SDN controller failures in the implemented network is very small. In one considered scenario for a communication problem to realise, a SDN controller would have to fail at the exact moment when a switch requires flow entry updates. Delays in the failover to the alternative controller would then probably lead to delays in these required flow entry updates and subsequently some communication delay. However, the probability of this is very low and these delays will not impact the implemented data acquisition functions since they do not depend on any time-sensitive messages. Another scenario would require all three SDN controllers to fail

simultaneously which would mean no back-up controllers would be available to control any switches. Since the architecture proposed in this work is based on deployment of all three SDN controller instances at three physically separate locations, the likelihood of this scenario occurring is also very low.

5.3.3 Automated Host Failover Functions

To test the host failover functions that were implemented in the Primary Substation Data Acquisition scripts, all links to the Mouille Point Gateway host were disabled while this script was running. As soon as these links were disabled the ONOS GUI reported the disconnection of the host with IP address 10.0.0.1 from the network topology. A warning message then appeared on the Clyde Road substation host's terminal window as it attempted to transfer its next set of text files. The message informed users that the IPE host was unable to establish a connection to its designated Gateway as it was currently unreachable. This was followed by another message reporting the script's attempt to connect to an alternative Gateway host. A successful connection to the Green Point host with IP address 10.0.0.2 was also reported. As soon as the alternative Gateway accepted the login credentials required by the file management service, the ONOS GUI showed an increase in network traffic between the Clyde Road host and the Green Point host, indicating a transfer of text files. After a few successful transmissions to the backup Gateway host, the links to the Mouille Point Gateway host were enabled again. The Clyde Road host's terminal window now reported no warnings and confirmed a successful connection to the host with IP address 10.0.0.1. Although the data from the Clyde Road substation was temporarily split across two Gateway hosts, an inspection of the Backend directories confirmed that the subsequent transfers of these files to the central repository ensured that they ended up in the same central location. Snapshots taken during this test of the IPE host's terminal window and this network traffic to the alternative Gateway observed in the ONOS GUI is shown in Figure 44.

Messages observed on the Clyde Road terminal window



Network traffic on links to alternative Gateway

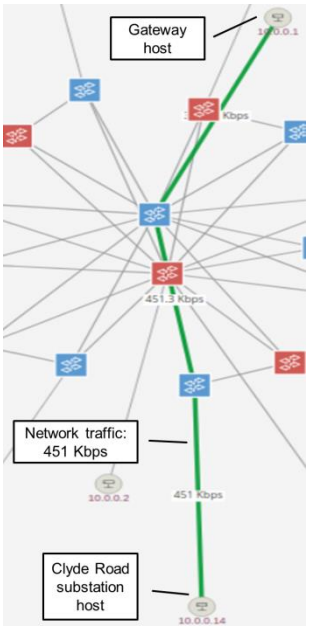
Figure 44: Snapshot of the reports and monitored topology during an automated host failover.

These results demonstrated the effectiveness of the implemented host failover function, but since this function was scripted to attempt a connection to the substation's primary Gateway host before it switched

over to the backup, a delay of a few seconds was introduced each time the primary host was unreachable for a file transfer attempt. This delay did not impact the overall data acquisition process, but it raises the risk that a Gateway host might fail while an IPE host is busy transferring its data. The use of the FTP protocol addresses this risk by ensuring that any failed file transfers would be detected and reported as an exception. As a potential future enhancement, these Python scripts can be configured to automate the handling of these sorts of exceptions with similar failovers functions if needed.

5.3.4 Automated Data Buffering

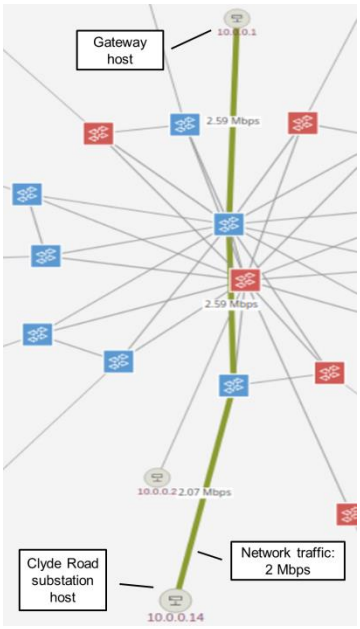
If a substation lost its connection to the NAN, it would be impossible for its IPE host to establish a connection to any Gateway hosts. The only way to avoid data loss in this scenario would be to store the data locally and to reattempt the data transfers as soon as the network connections can be re-established. Next, the ability of the implemented M2M services to perform this buffering function was tested, by disabling the link between the Clyde Road host and its switch. Figure 45 shows the network traffic reported before the host was disconnected, the messages displayed on the host's terminal window during the period of disconnection and the network traffic reported after the host was reconnected to the network for this test.



Network traffic before substation isolation event

```
connecting to ftp server
Cannot connect: [Errno 101] Network is unreachable
Connecting to alternate Gateway
Cannot connect: [Errno 101] Network is unreachable
No Gateways available, buffering data.
connecting to ftp server
Cannot connect: [Errno 101] Network is unreachable
Connecting to alternate Gateway
Cannot connect: [Errno 101] Network is unreachable
```

Messages observed on the Clyde Road terminal window



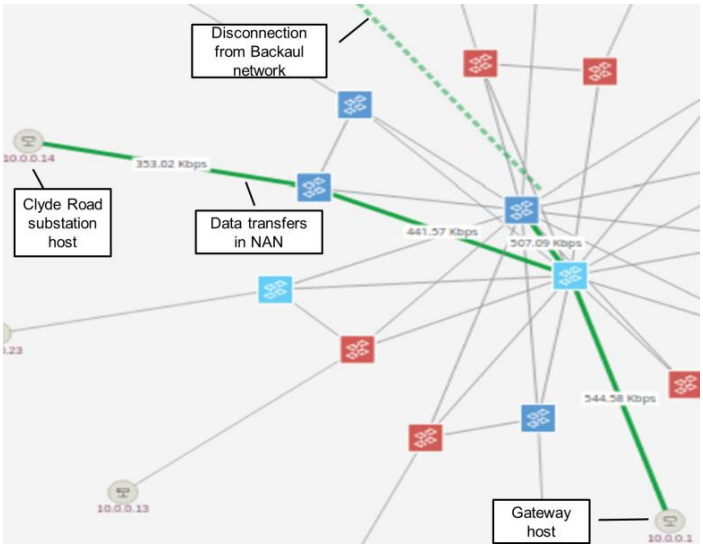
Network traffic after substation isolation event was resolved

Figure 45: Snapshots of network traffic and messages reported for substation isolation test.

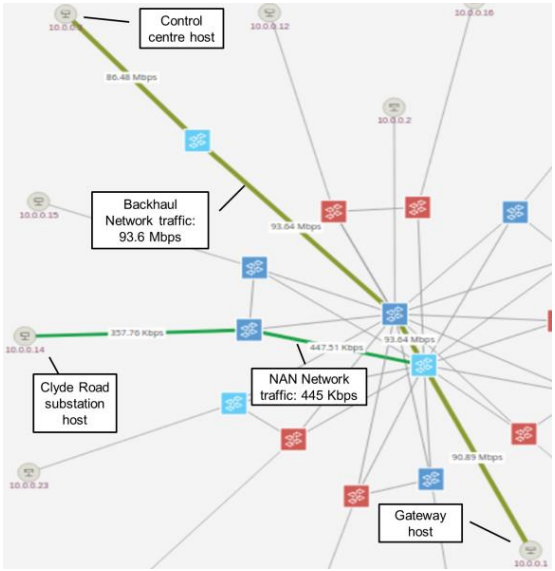
As soon as the Clyde Road host attempted to transfer its next set of files across the disabled link, its terminal window reported that the Gateway host was unreachable. It also displayed a similar message for the subsequent attempt to connect to the backup Gateway host that was also now unreachable. The terminal window then displayed a message that informed users that the IPE service would now buffer these data sets since no Gateway hosts were reachable at this point. After four failed attempts to transfer the data were reported, the network link to the Clyde Road host was re-enabled. This time, the host reported no warnings

and data transfers resumed. Interestingly, the amount of network traffic reported by the ONOS GUI on the active links was more than 2 Mbps, which was substantially higher than hundreds of kilobits per second that was previously observed.

The capability of the network to buffer data on Gateway hosts was tested next. A Gateway would need to buffer data in scenarios where an entire NAN might get isolated from the distribution grid MAN. For this test, the backhaul network connection between the Mouille Point host and the control centre host were disabled. The console windows for the Clyde Road IPE host and the Mouille Point Gateway host were then monitored as they attempted to perform their data transfer functions. Figure 46 shows two snapshots taken during this test. The first shows the reported disconnection of the backhaul link to the control centre host while the IPE host was transferring data to its Gateway host. The second shows the network traffic right after the backhaul link was enabled again.



Backhaul failure and maintained data transfers in the edge network during isolation.



Increase in network traffic to address backlog after edge network isolation fault was resolved.

Figure 46: Snapshots of network traffic and messages reported during an edge network isolation test.

During the simulated disconnection, the Gateway host reported the unreachability of the Backend with a warning message which was followed by a message that the data would be buffered on the Gateway host. Meanwhile, the IPE host continued to transfer its data to the Gateway without any interruption. After many unsuccessful attempts by the Gateway to communicate with the Backend host, the backhaul links were enabled again. With the Backend now reachable the Gateway started transferring all its acquired text files and as was observed with the previous test, the initial data transfers from the Gateway to the Backend required a lot more bandwidth, in this case more than 90 Mbps. This also normalised again with subsequent transmissions. Inspection of the Backend directories confirmed that all buffered data eventually reached the intended central storage repository which also confirmed the effectiveness of both data buffering functions. However, extensive delays between the moments the data was generated and the moment it reached the Backend host were observed in both scenarios. These delays were expected and were unavoidable.

The increase in network traffic in both test cases can be attributed to the build-up of data on the buffering hosts while they were unable to transfer these data sets. As soon as the connections were re-established, these hosts attempted to push all their buffered text files to their intended hosts to clear their backlogs. This resulted in more FTP packets being passed between the hosts that resulted in more network traffic. This is an important consideration for smart grid communication network designers, as this can impact the network's communication reliability. Depending on the amount of buffered data and the amount of simultaneously transmitting devices, the use of these data buffering functions can cause a large amount of data to flood the network after network repairs have been completed. If not properly managed, this may overload the network resources leading to further communication problems.

5.3.5 Traffic Scheduling

In addition to being fault tolerant, a reliable communication network also should limit the probability of avoidable communication delays that are usually caused by overloaded network resources. The traffic scheduling functions implemented in this network were tested by adjusting the parameters of the substation simulation scripts so that very large text files would be created for transfer over the network. The file threshold parameter was changed to record 100 000 readings before each file transfer which resulted in the creation of text files larger than 10 Megabytes in size. The scheduling parameter was also adjusted to delay transmissions for 10 seconds. The 10 Mbps constraint implemented on the NAN's network links therefore resulted in overloading of the network each time file transfers were performed. These overloaded network resources were clearly observable in the ONOS GUI with the overloaded links in the topology turning red, while reporting the maximum 10 Mbps of available bandwidth being utilised with each file transfer.

To observe the effect of this congestion on the RTL of the transferred packets, a terminal window was launched on Granger Bay substation host and a ping command was executed on it. Wireshark was used to record the details about these packet exchanges on the network and to identify the different types of network traffic present during these exchanges. Using Wireshark, the RTL for each of these ping messages was recorded as this test ran for ten file transfer cycles. Plotting these data sets on a graph showed that each cycle provided a similar trend for the ping message RTL measurements.

The trend for each cycle showed a steady increase in RTL for the pings up to a maximum of approximately 0.6 seconds. A sudden decrease in RTL was then observed with measurements under 0.05 milliseconds. The trend for one of these cycles is shown in the graph in Figure 47.

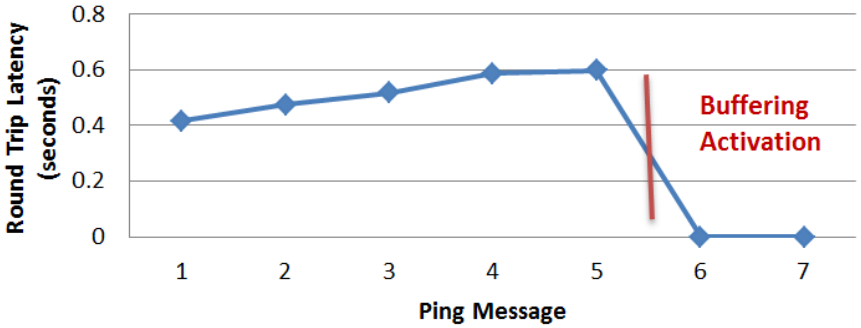


Figure 47: Improvement in round trip latency due to traffic scheduling functions.

These sudden decreases in RTL coincided with the triggering of the traffic scheduling function that stopped the text file transfers. Each time the traffic scheduling function's timer expired, it stopped the data from buffering on the host and allowed the transfer of the text files to proceed, resulting in a large amount of FTP packet exchanges on the network. This influx of FTP packets filled the network buffers of the switches which delayed the processing of packets on the network, including the ping messages. After the file transfers were completed, the timer was reset, and the scheduling function prevented any further file transfers to proceed for the next ten seconds. This removed the presence of the FTP packets from the network and reduced the amount of network traffic substantially, allowing faster transfers of the ping messages. By manually adjusting the scheduling parameters, the amount of congestion free time on the network could be increased or decreased as needed. Increasing the amount of buffering time however resulted in larger backlogs forming on the buffering hosts. More time was therefore required to work off these backlogs. For systems that rely on the transfer of very large amounts of data, there is a risk that the time required to work off these backlogs might exceed the amount of time that the hosts will be in a buffering state. This will lead to a situation where the host's storage resources gradually fill up until it runs out of storage space. In these scenarios, upgrades to the network's maximum bandwidth will be required.

Overall, the results from these tests verified the capability of the implemented network to recover from the following: link and node failures using FFR functions, SDN controller failures using automated controller failover, host disconnections using automated host failover functions and substation or network isolation events using data buffering functions. These results also demonstrated the ability of the implemented design to reduce the risk of overloading network resources with the transfer of bigdata, by using traffic scheduling functions. These were all features that will result in an overall increase in the communication reliability of the network. The execution of these functions did not interrupt any of the data acquisition functions, but instead it supported them by ensuring that they could be performed - even under circumstances that would normally lead to functional failures. Even in scenarios where communication failures and delays were unavoidable, these functions provided automated recovery mechanisms that resulted in no data being lost during any of the conducted tests. A summary of the outcomes of the conducted communication reliability tests is provided in Table 8.

Tested functions	Simulated failure scenarios	Outcomes	Impact on data acquisition functions
Fast Failover Recovery	Link Failure in a distribution grid NAN	Failure detected and traffic diverted to an alternative flow path.	Functions executed without interruptions or any data loss.
	Node Failure in a distribution grid NAN		
Automated SDN controller failover	SDN Controller Failure in a distribution grid NAN	Failure detected and mastership of impacted switches reassigned to alternative controller. Failed controller automatically restored.	
Automated host failover	Disconnected Gateway host in a distribution grid NAN	Data transfers rerouted to a backup Gateway host.	Functions executed without interruptions or any data loss. Slight delay in data transfers.
Automated data buffering	Disconnected substation in a distribution grid NAN	Data buffered on an IPE host and buffered data transferred after connection was re-established.	Functions executed without interruptions or any data loss. Delay in data transfers during buffering.
	Disconnection of a NAN from a distribution grid MAN	Data buffered on a Gateway host without impacting IPE host transfers. Buffered data transferred after connection was re-established.	Functions executed without interruptions or any data loss. Delay in data transfers from Gateway to Backend during buffering.
Traffic Scheduling	Overloaded network resources caused by large file transfers	Amount of network traffic reduced with scheduled data buffering, resulting in improved RTL for other network traffic.	Functions executed without interruptions or any data loss. Delay in data transfers during buffering.

Table 8: Summary of communication reliability tests.

5.4 Chapter Discussions

This chapter presented and analysed the results for the communication network development, smart grid functional enablement and communication reliability improvement tests.

A software-based communication network based on the proposed design for a section of the City of Cape Town's distribution grid was successfully implemented on a desktop computer, using a networking emulation system. The ability of the implemented SDN controllers to automatically discover the network topology, provide optimised network flow paths and report on network activity was also verified. The time required to set up such a network was measured and found to be around 23 minutes. Although the setup time for such a network is dependent on many variables, these measurements give a good indication of the streamlining software-based networking can offer to the grid development process over conventional hardware-based networking. These results may also be extrapolated to get a general estimation of the time that would be needed to set up larger smart grid networks.

Next, the ability of the implemented network to support the data acquisition functions required by an EAM system was evaluated. Using text files and FTP communication, sensor data from simulated substations were successfully transferred across the network to a host representing the grid's control centre.

These data sets were also temporarily stored and managed in the NAN using M2M Gateway and IPE services. This substation data acquisition process was entirely automated, and the transferred data was made available to EAM applications in files that were organised in directory structures managed by the developed M2M service platforms.

Finally, five network functions that were implemented to improve the communication reliability of the distribution grid were tested. The results from these tests demonstrated the network's ability to automatically recover from link failures, node failures, SDN controller failures, disconnected Gateway services and isolated distribution grid edge networks or substations. These results also demonstrated the effectiveness of traffic scheduling functions to improve the RTL of packets passed on congested communication networks. Each of these functions executed automatically in response to simulated failure events without interrupting the data acquisition process. Although some unavoidable data transfer delays were observed in situations where data buffering was required, no data losses were recorded with the execution of any of these network functions.

The next chapter will present the conclusion to this dissertation. A summary of this work will be presented, followed by a concise discussion of the conclusions drawn from this research. Recommendations for future work will also be discussed.

Chapter 6

Conclusions and Recommendations

The previous chapters presented the background, implementation and evaluation of a software-based network that aimed to improve the reliability of a smart distribution grid. This chapter will conclude this work by answering the research questions posed. It will also include a summary and reflection of the research, as well as recommendations for future work on the topic.

6.1 Research Summary

Given that communication is the cornerstone of the next generation power grids of the world, a lot of recent research has focused on improving the communication networks that support modern power grids. Studies that focused on the use of IoT-based networking in smart grids, specifically SDN, NFV and M2M, were reviewed in Chapter 2. This review identified research contributions that support the design of software-based communication architectures that aimed at improving the reliability of smart distribution grids. Some existing research gaps were also highlighted, with one of the main gaps being limited research focus on architectures that consider real grid implementations. Overall, the literature reviewed showed a growing interest in research that aims to overcome the challenges of smart grid communication with software-based networking, covering various smart grid domains and grid zones. This justifies the need for continual research in better communication architectures for smart grids, especially for those focusing on the distribution domain.

Chapter 3 established the requirements and design considerations for communication networks and architectures that aim to improve the reliability of distribution grids. This chapter stressed the interdependence between the two-way power and communication flows that exist in smart grids and the impact communication problems can have on grid reliability. It is also emphasised that the reliability of a distribution grid can be improved through communication functions and services that enable reliability centred smart grid systems that depend on data exchange and by improving the grid's communication reliability. Design considerations for a communication network that could support an EAM system, while reducing the probability of communication failure and delay were also discussed. A communication architecture, based on SDN, NFV and M2M that would allow this network to be created on a desktop computer for evaluation was then proposed. Using this architecture as a basis a software-based, distribution grid communication network design for the Green Point neighbourhood in Cape Town was also described that showed how these IoT frameworks could be implemented in real distribution grid designs.

Chapter 4 presented the streamlined implementation of the proposed communication network design using a network emulated system called Mininet. Two Python scripts that were developed to enable EAM data acquisition functions using M2M services were also described, along with a script that simulated data being generated by sensors in substations. The entire implementation was done on a desktop computer and the use of VMs simplified the implementation process as each VM could be replicated multiple times with minimal effort. The reusability of the open-source ONOS controller, OVS switches and various other open-source applications also meant that the implementation platform could be set up with a limited need for software development. This implementation platform was used to determine how simple and efficient it would be to prepare a network based on the proposed architecture and to evaluate whether the network could support smart grid data acquisition functions without interruption while it was subjected to various simulated communication problems. The test plan used to perform these evaluations was also presented.

Finally, chapter 5 presented the results obtained from executing this test plan along with an analysis of the results. The results showed that the network could be prepared in under 30 minutes, at which point it could support the automated acquisition of simulated substation data. When six different communication problems were simulated, the results showed that the implemented network functions and services performed as expected by enabling the network to recover in a reasonable amount of time without any resulting data loss or functional interruptions. Favourable results were also obtained when the network's ability to automatically reduce communication delays, caused by overloaded network resources, was tested. For each test, the network activity was monitored and controlled using applications that interfaced with the SDN controllers using APIs. This not only allowed the network to be reconfigured at run-time, but also provided users with a graphical representation of the network based on real-time information.

6.2 Conclusions and Research Contributions

This research aimed to determine if software-based communication networks can improve the reliability of smart grids in the distribution domain. Based on the results obtained, it can be concluded that this is indeed possible with the implementation of a communication architecture built on the pillars of SDN, NFV and M2M. Firstly, the results obtained in this research supports other studies that concluded that communication networks based on these frameworks can be used to improve networks with the implementation of various network functions and services that can contribute to reducing the probability of communication failures and communication delays. Through automation and real-time monitoring of the network, these network functions can also execute rapid control mechanisms, thereby minimising restoration time after failures occur. Most of the reviewed literature however considered implementations using modelling or prototypes based on hypothetical network models that focused on meeting the requirements of specific time-sensitive smart grid functions. The results of this work expand the benefits of software-based networking to communication networks based on real distribution grid models and smart grid systems that rely on non-time-sensitive big-data transfers.

Two key design requirements for communication networks that aim to improve grid reliability were established in this work. They were the ability to support reliability centred smart grid systems that rely on communication and the ability to improve communication reliability. By demonstrating the ability of a software-based network to support and enable the functions required by EAM systems, the research showed how the proposed communication architecture can be implemented to meet the first requirement. Furthermore, the results showed that a software-based network could improve communication reliability through the implementation of various automated network functions and serveries that monitored the network in real time. This demonstrated the proposed design's capability to meet the second requirement - the requirement for a design platform that can deliver more reliable network designs through improved network development and evaluation techniques was also highlighted. This study addressed this need when it demonstrated that a design platform based on software-based networking principles could simplify and streamline network development. A communication network based on a model of a real city's distribution grid was implemented on a desktop computer in under 30 minutes and through the simulation of various communication problems, a comprehensive evaluation of the design could be performed relatively easily. This evaluation offered valuable insights into the capabilities of the design and since the design could be easily reconfigured, tweaks and adjustment could also be made until the design performed as required. The results obtained in this study offer many advantages over conventional hardware-based network developments that usually take considerably longer to set up, evaluate and reconfigure. The use of a decoupled control plane and VMs not only improves the network development process, but also makes it much easier to maintain a network. Opportunities to integrate customised network applications into the designs for software-based networks were also demonstrated in this research. This feature will add another degree of flexibility to network implementation which may assist network designers in overcoming various smart grid challenges, such as those that utilities face with limitations posed by supplier specific solutions and technologies.

Besides offering a means to improve grid reliability, one of the main contributions of this research is the insights it provides into how software-based networking can reduce the cost of smart grid implementations for utilities. Excluding the costs associated to the procurement of the NFVI and the man-hours required for software development and deployment, this research showed that virtually no cost is associated with the setup of a software-based network. Although not implemented in this work, related literature supports the fact that the VMs used in the proposed design platform can be deployed directly in a distributed platform of networked devices. This will not only simplify the deployment of network designs in actual grid environments but will also allow utilities to repurpose existing commodity grid infrastructure for network development and improvement, thereby reducing the overall hardware cost associated with smart grid implementations. Subsequent changes to the network configuration triggered by changes in the grid environment can also be accommodated more easily using the proposed process of centralised design with modular roll-out. Compared to the tedious manual work often involved with reconfiguring hardware-based networks, software-based networks will be much easier to maintain and improve, which can reduce the

operating costs of a smart grid implementation. By developing networks in a separate design and evaluation platform before they are deployed in actual grid environments, the risk of grid malfunctions during implementation can also be reduced drastically which in turn will enable utilities to meet their regulatory compliance requirements with more confidence.

In contribution to literature that focuses on smart grid research, this study addressed three research gaps. Firstly, this research demonstrated how SDN, NFV and M2M can be combined to improve the large-scale communication networks required to support smart distribution grids. This study then showed how these networks can be implemented to support smart grid applications that rely on non-time-sensitive, big-data communication. Lastly, instead of relying on hypothetical network models, this study considered a real city distribution grid model for implementation using a network emulation system. Since many distribution grids around the world follow similar architectures, topology design approaches and grid technologies, the results of this research may be applied in other smart grid designs and may even be expanded to larger grid network models and smart grid implementations in other domains. The measurements recorded in this study may also serve as a useful reference for scheduling and planning network development projects.

Overall, the opportunities created by this research to improve a distribution grid's reliability and streamline its development may offer utilities a means to develop better electricity distribution grids that are also more economically viable to implement. These benefits will likely encourage the uptake of smart grid by utilities and the promotion of the use of more technologies that can integrate with smart grid networks by various grid users. In the long term, this may also result in the uptake of more cost effective and energy efficient technologies that may help to reduce the cost of electricity and the world's carbon footprint. By making modern electricity distribution grids more reliable and easier to develop, control and maintain, these network architectures may contribute to an important objective of many utilities around the world, which is to make electricity more abundantly available to meet the world's changing energy demands.

6.3 Recommendations

This dissertation presented a software-based communication network design and implementation that aimed to improve distribution grid reliability. As the scope of this work was limited, opportunities exist for this design to be adapted, improved, or expanded with consideration of the implications of the findings for theory and practice.

Firstly, because network security is just as important for modern power grids as reliability, further research is needed to determine if the architecture proposed in this work can be adapted to improve communication security in smart distribution grids. Together with this work, the work of [61] that considered SDSec may serve as a reference for such a study. The provisioning of M2M services for certain security functions such as intrusion management and encryption should also be considered. Complimentary to this,

future research should consider the use of the architecture proposed in this work to support other smart grid capabilities such managing grid data storage or virtualising and managing smart grid devices that interface with grid infrastructure.

Further research should also consider improvement of the software applications that were used in this work. Because open-source applications were used, they were not specifically developed for use in smart grids and opportunities exist to extend and improve these applications to make them more efficient and effective. Opportunities for improvement in the Substation Simulation script and the data acquisition scripts that were developed as part of this work also exist. An example of a potential enhancement may be to extend these scripts to allow them to accommodate the rapid transfer of short messages used by other smart grid systems. As an alternative to transferring data in batches using text files and FTP, these enhancements may consider using other protocols such as HTTP to exchange messages through service platforms. The OpenMTC platform [47] may be considered for such an implementation. Alternatively, the use of database services may also be considered for managing big-data sets.

As this work only considered implementation of an Ethernet network, future research should consider enhancing this design to accommodate multiple communication protocols. This will be necessitated by the fact that different grid stakeholders will likely share the same smart grid communication infrastructure which will require integration between various heterogeneous networks and devices. The software-based networking architecture proposed in this work may be extended to support this integration, by providing functions and services that perform network and data model translation. In reference to the reviewed study in [69], this architecture may also be extended to leverage the benefits of monitoring and control using SDN applications across grid domains that rely on different communication standards. The extension of this architecture to provide dedicated virtual private networks (VPN) for specific smart grid systems on shared network infrastructure also requires further investigation.

The EAM system considered in the implementation of this design is not the only smart grid system available that can improve grid reliability. Future work should consider how the proposed communication architecture can support and enable other smart grid systems as well. Based on the benefits and opportunities of software-based networking highlighted in this work, one of the potential focus areas for these studies should be network functions and services that can reduce the end-to-end latency of time-sensitive messages exchanged by systems that monitor, control, and enable automated grid protection functions. Examples of the systems applicable to the distribution domain include substation automation and distribution management. Considerations for systems that rely on communication with mobile devices, such as workforce management systems should also be considered as this design scope was limited to stationary grid devices.

The scope of the design implemented in this work was limited to a NAN and its backhaul links to a distribution grid control centre. Research into the expansion of this design to include other communication

networks in the distribution domain should be considered as future work. More focus needs to be placed on internal substation networks that depend on specific substation communication standards that may be bound by supplier specific technologies in legacy systems. Software-based networking may be used to facilitate the breakdown of the silos that usually form in these legacy systems. Because the design scope in this work was also limited to a section of an electricity distribution grid, the potential to extend this design to multiple grid sections or possibly an entire city distribution grid also requires further investigation. Such a study should however consider the increase in network traffic that will result from an increase in the amount of data generating grid devices connected to these networks. Using [63] as a reference, further research should also aim to determine the optimal amount of SDN switches and SDN controllers needed for larger distribution grid communication networks and their optimal placement. Implementation of such a network will likely be very resource intensive. Although a high-performance computer should be able to meet the resource requirements for implementing such a network to a certain degree, one may not be readily available for this purpose. A possible alternative worth considering is to use multiple networked computers in a distributed virtual network. Maxinet [82], which is an enhancement of the Mininet system for distributed implementations, should be considered for such an implementation.

Finally, to better understand the implications of implementing the proposed architecture in a distributed platform using the process proposed in this work, future research should consider network implementations in distributed testbed environments. Some utilities and institutions have testbeds that may be considered for such a study, some of which were highlighted in the survey performed by W. Tushar et al [83]. Other examples include testbeds developed for academic purposes such as GENI which was reviewed in [64], and testbeds for industry research such as OneLab developed by Eskom and Huawei [84]. Such a study should aim to confirm that the VMs and VNFs created in the proposed design platform can indeed be deployed in distributed environments. Following such a confirmation these studies should investigate any significant differences between the results obtained from evaluations conducted with these distributed platforms and the results obtained in the design platforms. When considering these results, attention should be placed on factors such as the physical distance between NFVI, hardware constraints on different platforms and the impact of hazards in the physical environment. As part of this research, interfaces to actual smart grid devices such as IEDs or PMUs should also be considered. This may provide further insights into the integration of these software-based networks with physical grid infrastructure. Following the success of such an implementation, future work should consider a case study on implementing this communication architecture in an actual distribution grid. A comprehensive understanding of the distribution grid chosen for this case study will be required, as there will be more risks associated when working with a real grid environment. This case study should aim to limit the need for procuring and installing new infrastructure by repurposing existing substation infrastructure as NFVI as far as possible. Once completed, a comparison between the deployment time required for the real grid implementation and the results obtained in this research should be considered and these results may be used as an input for planning other smart grid implementation projects.

References

[1] X. Fang, S. Misra, G. Xue and D. Yang, "Smart Grid - The New and Improved Power Grid: A Survey," in IEEE Communications Surveys & Tutorials, vol. 14, no. 4, pp. 944-980, Fourth Quarter 2012.

[2] K. Folly, “Challenges in implementing smart grid technologies in Africa,” presented at Africa Utility Week, 13 May 2013.

[3] R. Kappagantu and S. Arul Daniel, “Challenges and Issues of Smart Grid Implementation: A Case of Indian Scenario,” in Journal of Electrical Systems and Information Technology vol. 5, no. 3, pp. 453-467, 2018.

[4] H. Mashad Nemati, A. Sant’Anna and S. Nowaczyk, “Overview of Smart Grid Challenges in Sweden.” in the Swedish Artificial Intelligence Society Workshop (SAIS) 2014 Proceedings, pp. 155-164, 2014.

[5] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges," in IEEE Communications Surveys & Tutorials, vol. 15, no. 1, pp. 5-20, First Quarter 2013.

[6] Z. Fan et al., "Smart Grid Communications: Overview of Research Challenges, Solutions, and Standardization Activities," in IEEE Communications Surveys & Tutorials, vol. 15, no. 1, pp. 21-38, First Quarter 2013.

[7] M. Uslar, et al. “Applying the Smart Grid Architecture Model for Designing and Validating System-of-Systems in the Power and Energy Domain: A European Perspective,” in Energies, vol. 12, no. 2, pp. 258, Jan. 2019.

[8] J. L. Chen, Y. W. Ma, H. Y. Kuo, C. S. Yang and W. C. Hung, “Software-defined network virtualization platform for enterprise network resource management,” in IEEE Transactions on Emerging Topics in Computing, vol. 4, no. 2, pp. 179–186, April 2016.

[9] Y. Cui, S. Xiao, C. Liao, I. Stojmenovic and M. Li, “Data centers as software defined networks: Traffic redundancy elimination with wireless cards at routers,” in IEEE Journal on Selected Areas in Communications, vol. 31, no. 12, pp. 2658–2672, December 2013.

[10] R. Ahmed and R. Boutaba, “Design considerations for managing wide area software defined networks,” in IEEE Communications Magazine, vol. 52, no. 7, pp. 116–123, July 2014.

[11] A. Hakiri and P. Berthou, “Leveraging SDN for the 5G Networks: Trends, Prospects and Challenges,” June 2015. [Online]. Available: http://search.proquest.com/docview/2082681892/.

[12] ITU-T, “Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks - Framework of software-defined networking,” Recommendation ITU-T Y.3300, 2014.

[13] The IEEE Software Defined Networks website. [Online]. Available: https://www.sdn.ieee.org.

[14] The Cisco SDN webpage, [Online] Available: https://www.cisco.com/c/en/us/solutions/software-defined-networking/overview.html.

[15] The Nokia Nuage Networks website. [Online]. Available: https://www.nuagenetworks.net.

[16] Green Building Council South Africa, “Green policies slowly help market growth,” 29 October 2019. [Online]. Available: https://gbcsa.org.za/green-policies-slowly-help-market-growth/.

[17] National Electricity Regulator of South Africa, “Electricity supply quality of service Part 1: Minimum standards,” Specification NRS 047: -1:2002, 2002. [Online]. Available: http://www.nersa.org.za/Admin/Document/Editor/file/Electricity/IndustryStandards/NRS047%20part%201.pdf.

[18] W. Wang, Y. Xu, and M. Khanna, “A survey on the communication architectures in smart grid,” in Computer Networks, vol. 55, no. 15, pp. 3604-3629, 2011.

[19] M. H. Rehmani, A. Davy, B. Jennings, and C. Assi,” Software Defined Networks based Smart Grid Communication: A Comprehensive Survey,” in IEEE Communications Surveys & Tutorials, vol. 21, no.3, pp 2637-2670, 2019.

[20] M. Emmanuel and R. Rayudu, “Communication Technologies for Smart Grid Applications: A Survey,” in Journal of Network and Computer Applications vol. 74, pp. 133-148, October 2016.

[21] CEN-CENELEC-ETSI Smart Grid Coordination Group, “Smart grid reference architecture,” Technical Report, 2012. [Online]. Available: https://ec.europa.eu/energy/sites/ener/files/documents/xpert_group1_reference_architecture.pdf.

[22] IEEE 2030 Work Group, “IEEE 2030-2011: Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-User Applications, and Loads”, Technical Standard, 2011.

[23] The IEC Smart Grid Website. [Online]. Available: https://www.iec.ch/smartgrid/.

[24] The NIST Smart Grid Website. [Online]. Available: https://www.nist.gov/engineering-laboratory/smart-grid.

[25] S. Roy, D. Nordell, and S. S. Venkata, “Lines of Communication,” in IEEE Power and Energy Magazine vol. 9.5, pp. 64–73, 2011.

[26] P. P. Parikh, M. G. Kanabar and T. S. Sidhu, "Opportunities and challenges of wireless communication technologies for smart grid applications," in IEEE PES General Meeting, Providence, RI, pp. 1-7, 2010.

[27] L. Chhaya, P. Sharma, G. Bhagwatikar and A. Kumar, “Wireless Sensor Network Based Smart Grid Communications: Cyber Attacks, Intrusion Detection System and Topology Control,” in Electronics, vol. 6, no. 1, p. 5, Jan. 2017.

[28] T. Khan, N. Ramchunder and Y. Brijmohan, “The effectiveness of a wireless mesh communication network for distribution automation in electric utilities”, in Proceedings of the 25th Southern African Universities Power Engineering Conference, pp. 488-495, 2017.

[29] The Solarwinds website. [Online]. Available: https://www.solarwinds.com.

[30] R. Minerva, A. Biru and D. Rotondi, “Towards a definition of the Internet of Things (IoT),” May 2015. [Online]. Available: https://iot.ieee.org/definition.html.

[31] ITU-T, “SERIES Y: Global information infrastructure, internet protocol aspects and next-generation networks, Next Generation Networks – Frameworks and functional architecture models, Overview of the Internet of Things,” ITU-T Recommendation ITU-T Y.2060, 2012.

[32] Y. Simmhan, A. G. Kumbhare, B. Cao and V. Prasanna, "An Analysis of Security and Privacy Issues in Smart Grid Software Architectures on Clouds," IEEE 4th International Conference on Cloud Computing, Washington DC, pp. 582-589. 2011.

[33] ITU-T, “Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks - Framework of software-defined networking,” Recommendation ITU-T Y.3300, 2014.

[34] The Cisco Nexus 3000 network switch webpage. [Online]. Available: https://www.cisco.com/c/en/us/products/switches/nexus-3000-series-switches/.

[35] J. Biswas et al., "The IEEE P1520 standards initiative for programmable network interfaces," in IEEE Communications Magazine, vol. 36, no. 10, pp. 64-70, October 1998.

[36] The Open Datapath specification webpage. [Online]. Available: https://www.opennetworking.org/technical-communities/areas/specification/open-datapath/.

[37] E. Haleplidis et al., "Network Programmability with ForCES," in IEEE Communications Surveys & Tutorials, vol. 17, no. 3, pp. 1423-1440, third quarter 2015.

[38] T. Luo and S. Yu, "Control and communication mechanisms of a SoftRouter," in Proceedings of the Fourth International Conference on Communications and Networking, China, pp. 1-6, 2009.

[39] F. Hu, Q. Hao, and K. Bao, "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation," in IEEE Communications Surveys & Tutorials, vol. 16, no. 4, pp. 2181-2206, Fourth quarter 2014.

[40] The OpenVSwitch website. [Online]. Available: http://www.openvswitch.org.

[41] O. Salman, I. H. Elhajj, A. Kayssi and A. Chehab, “SDN Controllers: A Comparative Study,” 18th Mediterranean Electrotechnical Conference, Limasson, Cyprus, 2016.

[42] ETSI, “Network Functions Virtualisation (NFV); Infrastructure Overview,” Group Specification ETSI GS NFV-INF 001 V1.1.1, 2015.

[43] Y. Xue, M. Ni and W. Yu, "Approach for studying the impact of communication failures on power grid," 2016 IEEE Power and Energy Society General Meeting (PESGM), Boston, MA, pp. 1-5, 2016.

[44] Q. Wang, M. Pipattanasomporn, M. Kuzlu, Y. Tang, Y. Li and S. Rahman, "Framework for vulnerability assessment of communication systems for electric power grids," in IET Generation, Transmission & Distribution, vol. 10, no. 2, pp. 477-486, 2016.

[45] S. Al-Rubaye, E. Kadhum, Q. Ni and A. Anpalagan, "Industrial Internet of Things Driven by SDN Platform for Smart Grid Resiliency," in IEEE Internet of Things Journal, vol. 6, no. 1, pp. 267-277, Feb. 2019.

[46] M. Chen, J. Wan and F.Li, “Machine-to-machine communications: architectures, standards and applications,” in KSII Transactions on Internet and Information Systems, vol. 6, no. 2, pp. 480–497, Feb 2012.

[47] oneM2M, “Solving the IoT Platform Challenge”, November 2015. [Online]. Available: https://onem2m.org/images/files/onem2m-executive-briefing_A4.pdf.

[48] ETSI, “Machine-to-Machine communications (M2M); functional architecture,” Technical Standard TS-102 690 V2.1.1, 2013.

[49] oneM2M, “Functional architecture for the oneM2M services platform,” Technical Standard TS-0001-V3.15.1, 2019.

[50] The open machine type communication (openMTC) website. [Online]. Available: https://www.openmtc.org.

[51] ETSI, “Machine-to-Machine communications (M2M); Applicability of M2M architecture to Smart Grid Networks; Impact of Smart Grids on M2M platform,” Technical Report TR-102 935-V2.1.1, 2012.

[52] N. Dorsch, F. Kurtz and C. Wietfeld, "Communications in distributed smart grid control: Software-defined vs. legacy networks," 2017 IEEE Conference on Energy Internet and Energy System Integration (EI2) in Beijing, pp. 1-6, 2017.

[53] N. Dorsch, F. Kurtz, H. Georg, C. Hägerling and C. Wietfeld, "Software-defined networking for Smart Grid communications: Applications, challenges and advantages," 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm), Venice, pp. 422-427, 2014.

[54] N. Dorsch, F. Kurtz, F. Girke, and C. Wietfeld, “Enhanced fast failover for software-defined smart grid communication networks,” in IEEE Global Communications Conference (GLOBECOM), pp. 1–6, Dec 2016.

[55] J. Lewis “Carrier Grade Resilience in Geographically Distributed Software Defined Networks”, Research Dissertation, University of Cape Town, December 2016.

[56] Open Networking Foundation (ONF) OpenFlow switch specification version 1.5.1, March 2015.

[57] The Open Networking Operating System (ONOS) website. [Online]. Available: https://www.opennetworking.org/onos/.

[58] M. Niedermeier and H. de Meer, “Constructing Dependable Smart Grid Networks using Network Functions Virtualization,” in Journal of Network and Systems Management, vol. 24, issue 3, pp. 449–469, July 2016.

[59] C. Mouradian, N. T. Jahromi and R. H. Glitho, "NFV and SDN-Based Distributed IoT Gateway for Large-Scale Disaster Management," in IEEE Internet of Things Journal, vol. 5, no. 5, pp. 4119-4131, Oct. 2018.

[60] F. Y. Okay and S. Ozdemir, "A fog computing based smart grid model," 2016 International Symposium on Networks, Computers and Communications (ISNCC), Yasmine Hammamet, pp. 1-6, 2016.

[61] Y. Jararweh et al., “Software-Defined System Support for Enabling Ubiquitous Mobile Edge Computing,” in The Computer Journal, vol. 60, issue. 10, pp. 1–15. October 2017.

[62] E. G. d. Silva et al., “A one-class NIDS for SDN-based SCADA systems,” in IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), vol. 1, pp. 303–312, June 2016.

[63] N. Nafi, K. Ahmed, M. Gregory, and M. Datta, “Software defined neighbourhood area network for smart grid applications,” in Future Generation Computer Systems, vol. 79, part 2, pp. 500–513, 2018.

[64] A. Sydney, D. Ochs, C. Scoglio, D. Gruenbacher and R Miller, “Using GENI for experimental evaluation of Software Defined Networking in smart grids,” in Computer Networks, vol. 63, pp. 5-16, 2014.

[65] N. Dorsch, F. Kurtz, S. Dalhues, L. Robitzky, U. Häger and C. Wietfeld, "Intertwined: Software-defined communication networks for multi-agent system-based Smart Grid control," 2016 IEEE International Conference on Smart Grid Communications (SmartGridComm), Sydney, NSW, pp. 254-259, 2016.

[66] W. Guo, V. Mahendran, and S. Radhakrishnan, "Achieving throughput fairness in smart grid using SDN-based flow aggregation and scheduling," 2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), New York, NY, pp. 1-7, 2016.

[67] A. Meloni, P. A. Pegoraro, L. Atzori and S. Sulis, "An IoT architecture for wide area measurement systems: A virtualized PMU based approach," 2016 IEEE International Energy Conference (ENERGYCON), Leuven, pp. 1-6, 2016.

[68] The ONOS web GUI webpage. [Online]. Available: https://wiki.onosproject.org/display/ONOS/The+ONOS+Web+GUI/.

[69] E. Molina, E. Jacob, J. Matias, N. Moreira, and A. Astarloa, "Using software defined networking to manage and control IEC 61850-based systems," in Computers and Electrical Engineering, vol. 43, pp. 142–154, April 2015.

[70] V. C. Gungor et al., "A Survey on Smart Grid Potential Applications and Communication Requirements," in IEEE Transactions on Industrial Informatics, vol. 9, no. 1, pp. 28-42, Feb. 2013.

[71] US Department of Energy, "Communications Requirements of Smart Grid Technologies,", 2010. [Online]. Available: https://www.energy.gov/sites/prod/files/gcprod/documents/ Smart_Grid_Communications_Requirements_Report_10-05-2010.pdf.

[72] IEC, "Communication networks and systems in substations – Part 5: Communication requirements for functions and device models," IEC standard 61850-5, 2003.

[73] RFC, "File Transfer Protocol," Technical Standard, October 1985. [Online]. Available: https://tools.ietf.org/html/rfc959/

[74] City of Cape Town, "City of Cape Risk Report for the Energy and Climate Change Directorate," August 2020.

[75] J. Zhang, B. Seet, T. Lie and C.H. Foh, "Opportunities for Software-Defined Networking in Smart Grid," 2013 9th International Conference on Information, Communications & Signal Processing, Tainan, pp. 1-5, 2013.

[76] The Mininet website. [Online]. Available: http://www.mininet.org.

[77] The ONOS Tutorial web page. [Online]. Available: https://wiki.onosproject.org/display/ONOS/Basic+ONOS+Tutorial/.

[78] Fraunhofer Fokus, "OpenMTC IPE-sensors demo app". [Online]. Available: https://www.openmtc.org/doc/training/training-ipe-sensors.html.

[79] The Open Networking Foundation web page. [Online]. Available: https://www.opennetworking.org/onos/.

[80] The Atomix website. [Online]. Available: https://www.atomix.io.

[81] The Wireshark website. [Online]. Available: https://www.wireshark.org.

[82] The Maxinet website. [Online]. Available: https://maxinet.github.io.

[83] W. Tushar et al., "Smart Grid Testbed for Demand Focused Energy Management in End User Environments," in IEEE Wireless Communications, vol. 23, no. 6, pp. 70-80, December 2016.

[84] Eskom, "Eskom and Huawei partner to advance Smart Grid innovation", [Online]. Available: http://www.eskom.co.za/news/Pages/Mayy17.aspx.

Appendix A

Comparison of popular SDN controllers

Controller	**Southbound APIs**	**Northbound APIs**	**Application Domains**	**Throughput performance**	**Latency**
Beacon	OpenFlow 1.0	REST API	Research	Average	Low
FloodLight	OpenFlow 1.0, 1.3	REST API	Campus	Low	Average
Iris	OpenFlow 1.0, 1.3, OVSDB	REST API	Carrier grade	Average	Low
LibFluid	OpenFlow 1.0, 1.3	REST API	-	High	Low
Maestro	OpenFlow 1.0	REST API	Research	Low	Low
MUL	OpenFlow 1.0, 1.4, 1.3, OVSDB, OFCONFIG	REST API	Datacentre	High	Low
NOX	OpenFlow 1.0	REST API	Campus	Low	Average
ONOS	OpenFlow 1.0, 1.3	REST API	Datacentre, WAN and Transport	Low	Low
OpenDayLight	OpenFlow 1.0, 1.3, 1.4, NETCONF/ YANG, OVSDB, PCEP, BGP/LS, LISP, SNMP	REST API	Data centre	Low	Low
POX	OpenFlow 1.0	REST API	Campus	Low	High
Runos	OpenFlow 1.3	REST API	WAN, Telecom and Data centre	Low	Low
RYU	OpenFlow 1.0, 1.2, 1.3, 1.4, NETCONF, OFCONFIG	REST API	Campus	Low	High

Table A 1: Comparison of popular SDN controllers. Adapted from [41].

Appendix B

Summary of Related Work

Solutions	Objectives	Contributions	Domains Considered	Evaluation Methods	Reference Models
[52] Al-Rubaye et al., 2017	Proposed a SDN platform to support resiliency by reacting immediately whenever a failure occurs to recover smart grid networks using real-time monitoring techniques	An architecture that uses SDN to implement reactive FFR functions	Distributed Energy Resources	Experimental evaluation on a computer using RDO Openstack Mitaka on RHEL7	Hypothetical
[53] Dorsch et al., 2014	Presented a SDN based approach and platform for meeting substation communication requirements	An architecture that combines proactive FFR, load balancing and traffic prioritisation functions in a platform	Transmission and Distribution (Substation applications)	Experimental evaluation in a small LAN	Hypothetical
[54] Dorsch et al., 2016	Presented techniques for improved fault tolerance regarding link failures utilising the concepts of SDN	An architecture that combines the benefits of reactive and proactive FFR	Transmission and Distribution (Substation applications)	Experimental evaluation in a small LAN	Hypothetical
[58] Niedermeier et al., 2016	Studied the usage of NFV to construct a virtual network to transmit smart meter information in a dependable and cost-effective way	A model that motivates the benefits of NFV for smart grid applications in large-scale distributed networks	Customer	Mathematical modelling	Hypothetical
[59] Mouradian et al., 2016	Proposed an architecture for on-the-fly distributed Gateway provisioning in networks for large-scale disaster management leveraging NFV and SDN	An architecture that uses NFV to rapidly deploy functions, services and controllers as needed in the field	Not specified.	Experimental evaluation in a small LAN	Hypothetical
[60] Okay et al., 2016	Reviewed fog computing in smart grids and defined a fog computing based smart grid model	A model that explains how fog computing can be used to create data processing and storage networks in the network edge	Customer	No implementation	Hypothetical
[61] Jararweh et al., 2017	Proposed a framework to enable efficient and ubiquitous services by integrating different SDSys components with the MEC	An architecture that uses virtualisation and SDSys to provide functions and services in the network edge	Not specified	Experimental evaluation on a computer using Mininet	Hypothetical
[63] Nafi et al., 2018	Present a communication framework for NAN-based WSNs using SDN. Developed an analytical model to determine the number of switches and controllers required for the NAN	A reference topology for SDN-based NANs in smart grid. Insights into the number of controllers and switches needed for a NAN in a residential area.	Customer	Simulation model using Castalia	Suburb in Melbourne, Australia: 1000 houses with smart meters installed
[64] Sydney	Using the GENI	Demonstration of the	Customer	Experimental	Hypothetical

et al., 2014	platform, explored the modularity and flexibility provided by SDN to provide automatic fail-over mechanisms, load balancing and QoS guarantees for smart grid networks	flexibility and speedy implementation of SDN based FFR and load balancing in a real-world large -scale distributed platform		evaluation on a geographically distributed platform using GENI	
[65] Dorsch et al., 2016	Proposed a combined approach for the control of power and communication systems, exploiting the opportunities offered by SDN and MAS	An architecture that provides adaptive traffic prioritisation as a service	Transmission and Distribution (Substation applications)	Experimental evaluation on a small LAN	Hypothetical
[66] Guo et al., 2016	Proposed a novel SDN-based framework to provide fairness among smart meters, through flow aggregation and scheduling	An architecture that buffers, schedules, and aggregates data using SDN to support big-data acquisition	Customer	Experimental evaluation on a computer using Mininet	Hypothetical
[67] Meloni et al., 2016	Proposed an IoT solution based on virtualization to foster interoperability, reusability and flexibility for wide area measurement systems where virtualised PMUs are introduced	An architecture that that virtualises grid devices and creates virtual application entities that can aggregate and schedule data for transfer as a service	Transmission (Substation applications)	Experimental evaluation on a LAN connected to the internet to use a cloud-based PAS	Hypothetical
[69] Molina et al., 2015	Implemented a framework that used SDN to provide control and monitoring features with a global view of a network that interfaced with substation networks using IEC 61850 based protocols	An architecture that uses SDN applications to monitor and control other networks that use non-SDN protocols	Transmission and Distribution (Substation applications)	Experimental evaluation on a computer using Mininet	Hypothetical

Table B 1: Summary of related work reviewed in this research dissertation.

Appendix C

The Green Point Distribution Grid in the City of Cape Town

Cape Town's distribution grid spans an area covering roughly 400 km^2, supplying electricity to approximately 670000 service points. This distribution grid currently consists of about 80 main substations, 1000 primary substations, 2480 secondary substations and 6700 mini substations. It also has roughly 73000 distribution panels in the field. Figure C 1 shows how these substations are currently spread out across the city, giving a good indication of the scale of a metropolitan distribution grid. It also shows the location of the city's network control centre and the location of the Green Point suburb that was chosen as a model for the communication network design proposed in this work.

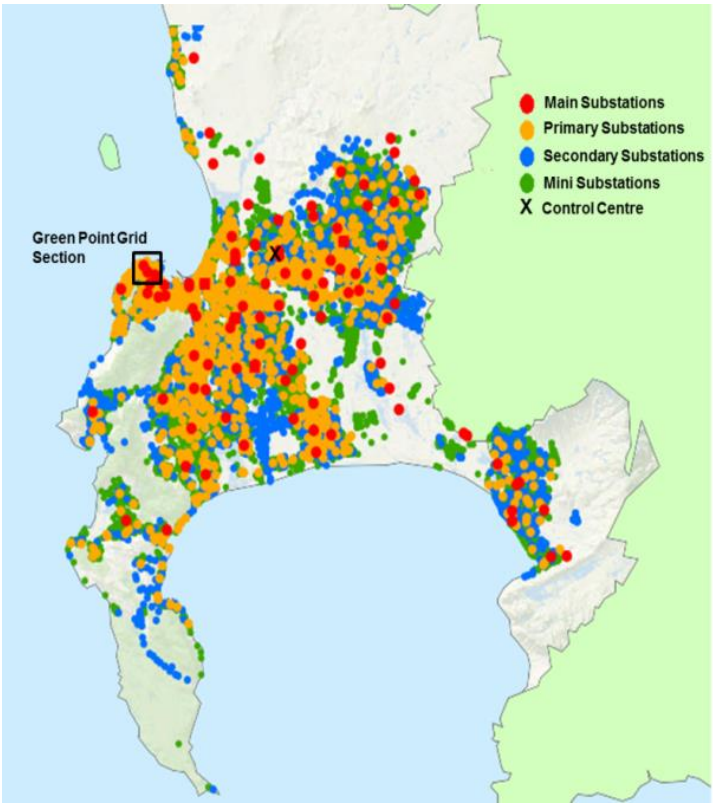


Figure C 1: Substation locations in the City of Cape Town.

Cape Town currently has good cellular network coverage provided by several service providers. 4G-LTE technology is used predominantly in these networks, although they are likely to be upgraded with 5G technology as soon as this technology becomes more commercially available. Cape Town also has good fibre network coverage, especially in densely populated residential and commercial districts of the city where various fibre providers have rolled out the needed infrastructure. The local municipality also owns its own fibre network that was developed to offer services at municipal facilities. This makes both cellular and fibre technologies viable options to consider for smart grid communication networks in the city.

A full-scale smart grid deployment in Cape Town's distribution grid could include smart devices in each of these grid components, making it a massive IIoT system. A large-scale communication network will be required to connect these devices to each other and to the servers in the distribution grid's control centre and developing a network at this scale will likely be a very complex, time consuming and expensive project. Designing smaller software-based networks for sub-sections of the grid that can be integrated at a later stage can simplify these implementations, which is why the Green Point neighbourhood was chosen for such a design.

Green Point consists of a mixture of commercial, industrial, and residential consumers with Cape Town Stadium as a major consumer on event days. The substations in this neighbourhood are supplied by two high voltage main substations that interface with the City's high voltage transmission grid. These main substations feed 13 primary substations in a radial network that in turn feed a combination of 8 secondary and 18 mini substations forming several supply rings. These substations are mainly connected via underground power cables. Each substation typically consists of a substation building that secures and protects the transformers, switchgear and other sensitive equipment used for electricity distribution and protection. The mini substations are protected by metal enclosures. A satellite image of this Green Point area, its substations, and their feeder relationships is shown in Figure C 2:

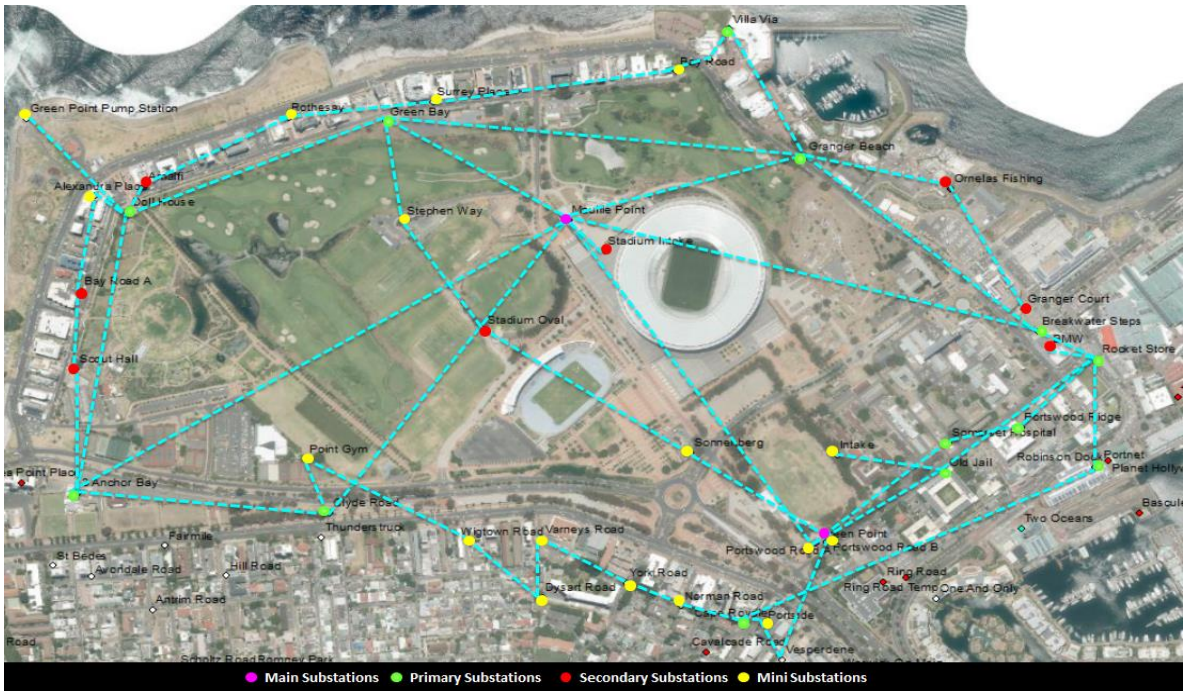


Figure C 2: Green Point Distribution Grid with substation positions and supply links.

The Cape Town distribution grid is currently monitored and controlled by a commercial distribution grid SCADA system. Due to the scale of the grid, this SCADA system was implemented to only monitor and control a limited number of substations. Therefore, many of the data collection and control activities

performed by the local utility is still performed manually. This includes the collection of asset condition data that is only collected by inspectors as part of their routine inspections.

Three of the biggest threats Cape Town currently faces in terms of its distribution grid assets are theft, vandalism and damage caused by illegal electricity connections. Over recent years the equipment used in and around substations have become targets for criminals who break into substations to steal copper-rich components or to vandalise substations. Security systems have been placed in and around some substations to trigger alarms, but these alarms currently only serve to frighten off intruders after the damage has been done to the substation structures. While this damage remains undetected and unrepaired, the equipment inside the substations is left vulnerable to the elements and to further vandalism. Exposed grid assets also become targets for electricity thieves that connect appliances directly to the grid infrastructure causing uncontrolled load increases. Over time, these load increases can overload distribution equipment and furthermore poses serious risks to public safety. A system that automates the frequent collection of asset condition data for use in an EAM system can contribute a lot to helping the local utility overcome these problems. This is one of the many potential opportunities that exist in Cape Town's distribution grid that may justify the development of a communication network for a smart distribution grid.

Appendix D

Mapping of communication reliability requirements to objectives

Communication reliability requirement	**Contributors in communication networks**	**Smart grid communication layer objective**	**Smart grid functional layer objective**
Reduced probability of communication failure (lost messages)	Link failures, node failures, source and destination unavailability, network isolation	Reduce the probability of communication failures to increase the probability of meeting communication service requirements	Ensure grid functions are executed successfully because messages reach their destinations
Reduced probability of delayed messages	Underperforming and overloaded network devices	Reduce the probability of delayed messages to increase the probability of meeting communication service requirements	Ensure grid functions are executed on time because messages reach their destinations on time
Reduced mean time to repair communication problems	Time to detect, locate, analyse, repair and test	Shorten repair times to reduce the probability of extended periods of communication failures and delays	Reduce the probability of failed and delayed grid functions caused by extended repairs

Table D 1: Mapping of communication reliability requirements to objectives.

Appendix E

Using Software-Based Networks to Improve Distribution Grid Reliability

Required Feature	Software-based Networking Capabilities Considered
Streamlined set up of networks that will enable big-data transfer from data generating devices in the field to central data repositories.	Using virtualisation, VMs that represent the physical components that usually make up distribution grid communication networks, such as network devices, storage devices and grid devices can be created as software to leverage the flexibility of software implementations over hardware. These devices can be connected to physical communication mediums or emulated networks using network emulation systems.
Configure, update or change the network during run-time without impacting the running EAM applications.	SDN offers a decoupled network control plane that can be reconfigured on a live network with limited impact on connected devices and applications. NFV also allows virtual devices to be instantiated and connected to a functioning software-based network at any time to provide additional network functions.
Streamline the evaluation of different communication network designs under fault conditions to compare their capabilities and determine their ability to support the required grid functions.	Using NFV principles, an entire virtual communication network may be developed to run on a single computer such as a PC or laptop. This will streamline the design process with the ability to setup and evaluate multiple design alternatives without the need for any hardware setup and configuration. Applications that interface with the SDN controllers and NFV MANO can provide easy-to-use, user interfaces that will allow network designers with limited programming knowledge to set up and change these virtual networks. Network emulation will provide more realistic results over simulations. The NFV MANO can be used to disable network interfaces to simulate various network failure scenarios for live network testing in a controlled environment.
Monitor and control the network remotely and implement network functions that automate network control to support data acquisition.	Application interfaces to the SDN controllers can be used to develop applications that use the monitoring and metering messages supported by SDN protocols to perform real-time monitoring, to trigger alarms and to collect information for analysis. These applications can also manually control flow table updates and automate network functions. Automated network control functions can be configured to limit their impact on running grid functions since they work directly through the SDN control plane. By using M2M principles these functions can be implemented as services, improving the network's flexibility and its ability to accommodate more heterogeneous devices and applications.

Table E 1: Arguments for using software-based networks to improve smart distribution grid reliability by enabling data acquisition functions for an EAM system.

Required Feature	Software-based Networking Capabilities Considered
Reduce the probability of communication failure (lost messages)	Application interfaces to SDN controllers enable them to implement network functions such as FFR that automate the detection and recovery from various communication failures. M2M services can also be implemented to automate failovers to back-up devices and to buffer data that cannot be successfully transmitted, to avoid data loss. NFV makes it possible to host these services and functions on VMs that can easily be provisioned in the network edge to ensure continued operations, even when an edge network is isolated from the core networks.
Reduce the probability of delayed messages	The use of NFV reduces the need for extensive network hardware developments as most VNFs can be hosted on the same NFVI. This reduces the amount of hardware in a network and therefore the risk of hardware malfunctions. The decoupled control plane offered by SDN also reduces the risk of control plane software malfunctions that can lead to communication delays. SDN controller interfaces can be used to implement network functions that reduce the risk of network device overloads and network congestion. These applications include load balancing, traffic prioritisation, traffic scheduling and flow aggregation.
Reduced mean time to repair communication problems	By automating the execution of network functions that allow the network to recover from issues that lead to communication failures and delays, the applications that interface with SDN controllers can drastically reduce the repair time for common communication problems. Applications that monitor and control communication networks will also allow network administrators to identify anomalies earlier and to correct them remotely. The NFV MANO will also enable the rapid replacement of entire VNFs and VMs that have malfunctioning elements with working copies, in cases where the malfunction cannot be easily identified.

Table E 2: Arguments for using software-based networks to improve smart distribution grid reliability by improving communication reliability.

Appendix F

List of Simulated Sensors Used in the Substation Simulation Script.

Sensor	Description	Unit
Transformer: Volts Red	Voltage measurements on the transformer's red phase	V
Transformer: Volts White	Voltage measurements on the transformer's white phase	V
Transformer: Volts Blue	Voltage measurements on the transformer's blue phase	V
Transformer: Amps Red	Electrical current measurements on the transformer's red phase	A
Transformer: Amps White	Electrical current measurements on the transformer's white phase	A
Transformer: Amps Blue	Electrical current measurements on the transformer's blue phase	A
Building: Temperature	Ambient temperature in the substation building	°C
Building: Humidity	Relative humidity in the substation building	g/m^3

Table F 1: Simulated sensor measurements with values.

Sensor	Description	Statuses
Switchgear: Breaker 1	The position of the circuit breaker in switchgear panel 1.	Open / Closed
Switchgear: Breaker 2	The position of the circuit breaker in switchgear panel 2.	Open / Closed
Transformer: Status	An indication if the transformer’s sensors are all functioning or not.	Healthy / Unhealthy
Door: Position	The position of the substation’s door.	Open / Closed
Security Alarm: Status	Indicates a triggered security alarm in a substation.	Alarm / No Alarm

Table F 2: Simulated sensor measurements with statuses.

Appendix G

End-to-end Procedure for Data Acquisition Implemented as Python Scripts.

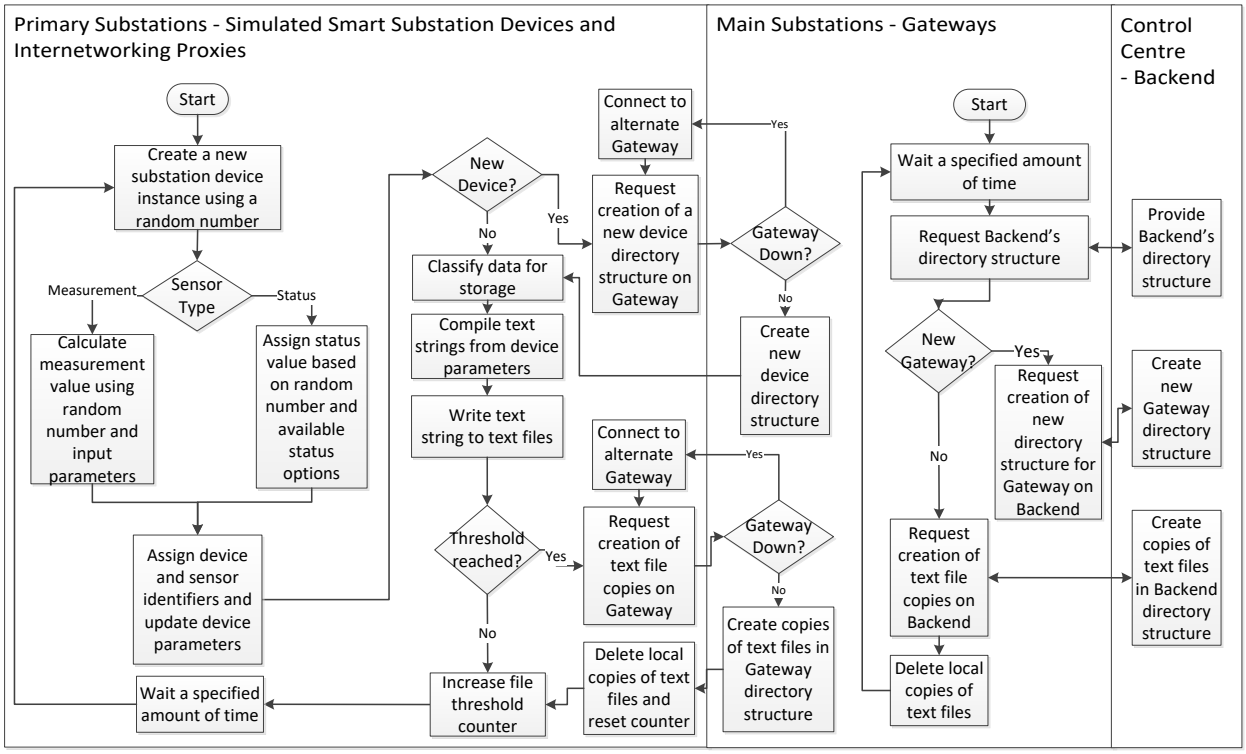


Figure G 1: End-to-end procedure for data acquisition.

Appendix H

Instructions for Accessing the Online Repository for the Implementing Testbed

The files containing the instructions and code that was used to implement the testbed used in this project have been uploaded to a GitHub online repository. This will offer researchers access to the outputs developed in this research and offer them the opportunity to collaborate on future work. To access this repository the following link can be opened in a web browser:

https://github.com/gerhard-brown/virtual-smart-distribution-grids.

Further instructions on setting up the testbed have been provided in the “Read me” file supplied in the repository.