A DEPENDABILITY ASSESSMENT FRAMEWORK FOR IOT DRIVEN APPLICATIONS

EHIZOJIE OJIE

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF A DOCTOR OF PHILOSOPHY

January 2020

AUTHORS DECLARATION

This thesis is submitted to Edge Hill University in support of my application for the degree of Doctor of Philosophy. It has been composed by me, all copyrighted material appearing in this thesis and all such use, has been clearly acknowledged. This thesis has not been submitted in any previous application for any degree.

Ehizojie Ojie

January 2020

ACKNOWLEDGEMENTS

I will like to use this medium in appreciating those, who stood by me in this challenging journey. This whole rigorous process has developed me academically to be able to withstand future challenges. This journey wouldn't have been a success, if not for the support, motivations and encouragements I received from my family and friends.

I will like to start by extending my sincere gratitude to my director of study, Professor Ella Pereira, who worked tirelessly to assist me throughout this process from the timely reading of my manuscript, to the guidance she offered me. If it was possible to choose a supervisor a thousand times, I will choose her. Her passion and maturity in handling my weakness created the success of this thesis. I will also like to specially thank my second supervisor, Dr Junyuan Wang, who was more of inspirational support to me. Her motivation created the drive for the success of this project.

I extend my sincere gratitude to my head of department, Professor Nik Bessis, thank you very much for creating a stimulating environment for learning. My profound gratitude to Dr Chitra Balakrishna, who started the struggle with me as part of my supervisory team. My special thanks to all the staffs of the department of computer science. To name a few, Dr Mark Liptrott, Dr Mark Hall, Dr Nemitari Ajienka, Dr Alex Akinbi, Dr Nonso A Nnamako and to my wonderful technician, Roger Perrin, thank you for your technical support.

To my fellow PhD colleagues which I meant at the course of my research journey, Pradeep Hewage, Sarah Mchale, Chandon Shill, Peter Ankomah, Jeffrey Ray and Oladotun Omosebi, you guys are more than a million to me, our meetings created a wonderful experience for me which I will always cherish. The office we shared and the influence you guys had on my research journey is well appreciated. Our critical discussions, exchange of ideas and Jokes will be forever remembered.

My sincere gratitude to my parents, thank you for the sacrifices you made for me to achieve this fit. You gave me hope, when all seems to be dark. Thank you. To my lovely daughter Favour Osetohamen, who came into my life at the beginning of this journey, you are a gift from God to me and finally my gratitude goes to my wife, Maria. I consider myself extremely lucky to have you lots.

ABSTRACT

One of the most recent developments in the area of computer science, communications and engineering is the Internet of Things (IoT). The novel paradigm of the IoT is gaining recognition in the numerous scenarios promoting the pervasive presence of smart things around us, through its application in various areas of the society, which include the transportation sector, healthcare sector and the agricultural sector. The most interesting part of this concept is its high impact on the enhancement of human lives.

Despite this fascinating concept has been significantly integrated into the development of the society at large, the issue of dependability in IoT application remains a major challenge to the development of this concept. The need for the IoT-based systems to be able to function according to their original specification without failures in their operation is crucial. IoT applications are mostly deployed in a constrained and critical operational environment, which involves the use of a large deployment in the components. It is important that IoT applications are dependable in the delivery of the required service, perform as they were designed to perform and survive challenging environments. Hence, a solution to address the issue of dependability in IoT application is required for the successful operation.

This research explored a systematic and comprehensive approach in creating a detailed understanding of the dependability requirements of an IoT application. An analysis of the existing approaches to the design of an IoT application was conducted. The components that make up an IoT application were identified, with variations in the number of components used in the design of the applications which leads to the classification of the small, medium and large-scale type of IoT applications.

The fault tree analysis method was used in analysing the dependability of the components and their relevance to the operation of an IoT application. Thereafter, a dependability assessment framework is created to aid in the assessment of the dependability of the components that make up an IoT application. A simulation experiment was conducted using the provision of the dependability assessment framework on OMNeT++ simulation environment. The results and findings of the evaluation on the various scales of IoT application creates an understanding of the importance of the variations of the components in the enhancement of the dependability of an IoT application.

Table of Contents

Chapter 1. Introduction 1

1.1. Overview 1

1.2. Motivation 1

1.3. Research questions 2

1.4. Research Aim and Objectives 2

1.4.1. Aim 2

1.4.2. Objectives 2

1.5. Original Contribution to Knowledge 4

1.6. Research Scope 5

1.7. Thesis Structure 5

Chapter 2. Literature Review 7

2.1. Introduction 7

2.2 Overview of IoT 7

2.3 IoT Application areas 8

2.3.2. Medical and health care 9

2.3.3. Manufacturing 10

2.3.4. Agricultural sector 10

2.4. Characteristics & Challenges of IoT Application 13

2.5. Dependability Issues in IoT Application 16

2.6. Summary 18

Chapter 3. Research Methodology 19

3.1. Introduction 19

3.2. Analysis of IoT Application 19

3.2.1. Case Analysis 20

3.3 Analysis of Dependability modelling techniques 20

3.3.1 Reliability block diagrams 21
3.3.2 Fault Trees Analysis 21
3.3.3 Markov Chain 21
3.3.4 Model Checking.....22
3.4. Fault Tree Analysis.....23
3.4.1. Standard Fault Trees 24
3.4.2. Cut Sets.....24
3.5. Simulation & Experiments.....24
3.6. Use Case Analysis 25
3.7. Summary.....25

Chapter 4. Understanding Dependability in IoT application.....26
4.1. Introduction.....26
4.2. Dependability and its related concept.....26
4.2.1. Dependability in IoT.....27
4.2.1.1. Dependability in Distributed Systems 28
4.2.1.2. Dependability in Grid Computing 29
4.2.1.3. Dependability in Cloud Computing.....30
4.2.1.4. Dependability of Wireless Sensor network 30
4.3. Survivability of Computer Systems……...31
4.4. Trustworthiness of Computer systems…….……33
4.5. Dependability Threats.....35
4.6. Dependability Attributes.....39
4.6.2. Reliability.....41
4.6.2.1. Reliable of network coverage and transmission 44
4.6.2.2. Reliability of Packet Delivery.....46
4.6.2.3. Reliability in Energy Efficiency 48
4.7. Related Attributes: Safety, Integrity, and Maintenance.....49

Chapter 5. Case Analysis of IoT Application 51

5.1. Introduction 51

5.2. IoT Selected Cases 51

5.2.1. Case 1: Target Application: An IoT based continuous glucose monitoring system 53

5.2.2. Case 2: Target Application: Mobile Health platform for diabetes management based on the internet of things 55

5.2.3. Case 3: Target Application: Wearable sensors nodes to read human physiological symptoms for elderly patients 57

5.2.4. Case 4: Target Application: IoT based approach for remote monitoring and self-management of diseases 60

5.2.5. Case 5: A Personalised Healthcare Monitoring System for Diabetic Patients ... 62

5.2.6. Case 6: Target Application: An IoT-Driven Application for Road Traffic Monitoring 65

5.2.7. Case 7: Target Application: IoT Sensors for Monitoring Potatoes Cultivation . 67

5.3. Analysis of the Findings 70

5.4. Summary 72

Chapter 6. Fault Tree Analysis of IoT Application Component 74

6.1. Introduction 74

6.2. Overview of the Fault Tree Analysis 75

6.2.1. Elements Of A Fault Tree 75

6.3 Qualitative Analysis of an IoT Application using Fault tree 77

6.3.1. Failure of a Sensor Node 79

6.3.2. Failure of the Communication Network 83

6.3.3. Failure of the Gateway device 86

6.4. Use of Fault Tree Analysis 89

6.5. Summary 90

Chapter 7. Dependability Assessment Framework for IoT Application 91

7.1. Introduction.....91

7.2. Framework Parameters 91

7.2.1 Size of the IoT Application.....93

7.3. Architecture of Dependability Assessment Framework 94

7.4 Application of the Dependability Assessment Framework 95

7.4.1. Framework Application Process.....95

7.5. Summary.....97

Chapter 8. Evaluation of Dependability Assessment Framework 98

8.1. Introduction.....98

8.2. Experimental Design.....99

8.2.1. Network Topology for Experimental Design 100

8.2.2. Simulation Environment.....101

8.3. Detailed Description of The Test Scales and Parameters. 102

8.3.1. Small Scale Application Setup 102

8.3.2 Medium-Scale Application Setup.....103

8.3.3. Large-Scale Application Setup 105

8.4. Experiment on Energy Consumption.....106

8.4.1. Results and Analysis of the Energy consumption 107

8.5. Experiment on Transmission Delay in IoT.....115

8.5.1. Results and Analysis of the Delay Metrics.....116

8.6. Experiment on Scalability of IoT application.....122

8.6.1. Results and Analysis of the Scalability Metrics 122

8.7 Analysis of the Findings 134

8.8 Use Case Evaluation 136

8.9. Summary.....145

Chapter 9. Conclusion……146

9.1. Research Achievement……..……147

9.2. Research Limitations 151

9.3. Recommendations and Future Work 152

References.....153

Appendix- A. Case Analysis.....166

Appendix - B. Overview of the Experimental Tool.....169

List of Figures

Figure 2.2: Characteristics of IoT Application 14

Figure 3.1: Adopted research methods for this research study 19

Figure 4.3: Reliability Characteristics in IoT Application 43

Figure 5.2: Continuous Glucose Monitoring using IoT 54

Figure 5.3: M-Health Platform for Diabetes Management Based on the Internet of Things 56

Figure 5.4: An architectural Framework for Reading Human Physiological Symptoms for Elderly patients 58

Figure 5.5: Remote Monitoring and Self-Management of Diseases 61

Figure 5.6: The Architecture of the Personalized Healthcare Monitoring System for Diabetic Patients 63

Figure 5.7: Architecture of An IoT-Driven Application for Road Traffic Monitoring 65

Figure 5.8: Application of Sensor Nodes in Potatoes Cultivation 67

Figure 6.1: Steps in Analysing the Dependability of the Components of an IoT Application 74

Figure 6.2: Fault tree of an IoT Application 78

Figure 6.3: Failure of a Sensor Node 79

Figure 6.4: Failure of the Energy Source 80

Figure 6.5: Failure of the Data Processing Unit 81

Figure 6.6: Failure in the Sensor Configuration Unit 82

Figure 6.9: Fault Tree Analysis of Communication process of an IoT Application 85

Figure 6.10: Failure of an IoT Gateway Device 86

Figure 6.11: Fault tree analysis of the failure in the hardware components of an IoT Gateway 87

Figure 7.1: Critical Dependability Requirement of an IoT Application 92

Figure 7.2: Variation of Components in an IoT Application 94

Figure 7.3: Dependability Assessment Framework 95

Figure 7.4: Framework Application Process 96

Figure 8.1: Processes involved in the evaluation of the dependability framework.....98
Figure 8.2: Experimental Design.....99
Figure 8.3: A representation of the small-scale configuration 102
Figure 8.4: A representation of the medium-scale configuration.....104
Figure 8.5: A representation of the large-scale configuration.....105
Figure 8.6: Experiment results of small-scale IoT application.....108
Figure 8.7: Experiment on energy consumption on medium-scale 1.....109
Figure 8.8: Experiment on energy consumption on medium-scale 2.....110
Figure 8.9: Experiment on energy consumption on large-scale 1.....111
Figure 8.10: Experiment on energy consumption on large-scale 2.....112
Figure 8.11: Experiment on delay using a small-scale scenario 116
Figure 8.13: Experiment on delay using a medium-scale scenario.....118
Figure 8.14: Experiment on delay using a large-scale scenario 119
Figure 8.15: Experiment on delay using a large-scale scenario 120
Figure 8.16: Experiment on energy consumption using Bluetooth.....123
Figure 8.17: Experiment on delay using Bluetooth.....124
Figure 8.19: Experiment on delay using ZigBee.....126
Figure 8. 20: Experiment on energy consumption using Wi-Fi 128
Figure 8.21: Experiment on delay using Wi-Fi 129
Figure 8.22: Experiment on energy consumption using MQTT 130
Figure 8.23: Experiment on delay using MQTT 131
Figure 8.24: Experiment on energy consumption using COAP.....132
Figure 8.25: Experiment on delay using COAP.....134
Figure 8.27: System architecture of Smart Healthcare for Pathology Detection 137
Figure 8.28: The system architecture of IoT-based monitoring system for heart disease patient 139
Figure 8.29: Architectural model of weather monitoring and precision farming 142
Figure 8.30: The use of IoT for Car Tracking System……...144

List of Tables

Table 2.1. IoT Application Areas and Domain 12

Table 3.1. Analysis of the Modelling Techniques 22

Table 4.1. A Summary of the Dependability in other Related Computing paradigms 31

Table 4.2. The Concept of Dependability, Survivability and Trustworthiness 35

Table 4.3. Dependability Attributes and Definitions 39

Table 5.1. IoT Selected Cases 52

Table 5.2. Components Values in Case 1 55

Table 5.3. Components Values in Case 2 57

Table 5.4. Components Values in Case 3 60

Table 5.7. Components Values in Case 5 65

Table 5.8. Components Values in Case 6 67

Table 5.9. Components Values in Case 1 69

Table 5.10. Analysis of IoT Application 70

Table 5.11. Categorisation of IoT application 72

Table 8.1. Parameters and Values for the Simulation Experiment 103

Table 8.4. Summary of the Results and Findings 114

Table 8.5. Summary of the Results and Findings on the experiment on delay 121

Table 8.6. Analysis of the Findings 135

Table 8.8. Analysis of the IoT Application of Li et al (2017) with dependability assessment framework evaluation 141

Table 8.9. Analysis of the IoT Application of Nagesh et al (2017) with dependability assessment framework evaluation 143

Table 9. Analysis of the IoT application of Thomas & Rad (2017) with dependability assessment framework evaluation 145

Table 9.1. Achievement of Research Questions 146

Table 9.3. Original Contribution to Knowledge 150

List of Publications

1. Ojie E & Pereira E, Exploring dependability issues in Internet of Application. Paper presented at 2nd International Conference on Internet of Things, Data and Cloud Computing (ICC 2017), Cambridge, United Kingdom.

2. Ojie E & Pereira E, Simulation Tools for internet of things: A review. Paper presented at the 1st International Conference on internet of Things and Machine Learning (IML 17), Liverpool, United Kingdom.

Chapter 1. Introduction

1.1. Overview

The Internet of Things (IoT) represents a vision in which the internet extends into the real world physical objects, which can be remotely controlled. This construction constitutes a network of physical objects which are embedded with software, sensors, electronic devices and connectivity that enable them to collect and exchange data through the IoT network. These objects, also known as “smart” objects can sensed the environment and can be controlled remotely across existing IoT network infrastructure to create opportunities for more direct integration which the physical world.

This relatively new paradigm is gaining recognition in various scenarios promoting the unique presence of smart things around us. These things can interact with each other and cooperate with other components to achieve a common goal. The most interesting part of the IoT vision is its positive impact on our daily lives with its continuous introduction in the several areas of the society. This concept will potentially provide new solutions to almost every aspect of the society.

The IoT is a phenomenon that occurs in such a confluence, resulting in an event between the sensor, real-time network and the data centres (Stogner, 2015). This combination of technology creates the need for a dependable application that can be used to process the substantial amount of information in a timely manner. The success of this concept will create a breakthrough in the field of technology.

1.2. Motivation

Despite the positive prospect of this construction, the issue of dependability remains a major challenge. IoT application must be dependable and provide real time information with no occurrence of failures in its operation (Macedo et al, 2014). Today's approach to the design of IoT applications does not guarantee dependability (Witrisal et al, 2019). Research on dependability in the area of IoT is still at its infant stage, considering that IoT as a concept is still developing. However, there is an absolute need for the issue of dependability to be addressed considering the current trends and advancement in IoT applications.

The need for the IoT-based systems to be able to function according to their original specification without failures in their operation is crucial for the IoT concept to be achievable. Hence, dependability of IoT system is an important factor to be considered in the area of IoT. Despite dependability, as a concept have been investigated and addressed in computer-based systems, when it comes to the paradigm of IoT, there is need for the understanding of what dependability means in the area of IoT and solution to the dependability issues in IoT applications. This research study was able to develop a dependability assessment framework for assessing the dependability of IoT applications.

1.3. Research questions

The research questions (RQ) for this study are as follows:

RQ 1. What constitutes dependability of IoT applications?

- What is the dependability in IoT?

RQ 2. How can the dependability be assessed and ensured in IoT applications?

- Do components deployed have any impact on dependability of IoT applications?

1.4. Research Aim and Objectives

1.4.1. Aim

The aim of this research study is to develop a dependability assessment framework for IoT application. The primary purpose of the framework is to assess the dependability of IoT applications through an evaluation of the impact of the components used in the design. The deployment of the framework will help in testing the effectiveness of the components in the achievement of a dependable IoT application.

In fulfilling the aim of this research study, the following objectives were explored:

1.4.2. Objectives

A. To conduct an in-depth investigation to understand the concept of dependability in relation to IoT applications.

Despite the fact, that dependability has been addressed in other areas of computer science, which include wireless sensor network and distributed systems, there is no a concise

definition of the meaning of dependability in an IoT application. In achieving the aim of this research study, an in-depth literature review was conducted to understand the concept of dependability in relation to IoT application and the key dependability requirements of an IoT were identified.

B. To critically analyse the characteristics and the functional requirement of existing IoT applications.

A critical analysis was conducted on existing IoT applications through the case study approach to assess the components that are used in the design. This process created an understanding of the structures and systematic functional constituents of an IoT application. The key characteristics of an IoT application were identified, through an analysis of various cases, which resulted in a classification of small, medium and large-scale type of IoT application.

C. To critically analyse the failures in the components of an IoT application.

The fault tree analysis method was used in analysing the components failures in an IoT application. This process created the understanding of the importance of the component and the root cause of the failures in the components that make up an IoT application. The criticality of the failures in the components to the operation of an IoT application was ascertained during this process.

D. To develop of a dependability assessment framework for IoT applications.

In achieving this objective, three layers were considered in the design of the framework. The first layer consists of the categorisation of IoT application into sizes with the values that make up the classification. The second layer of consideration in the framework consists of the components that make up an IoT application. In assessing the dependability of an IoT application there is need for a variation of the components used in the design of the application in regards to their relative performance. The third layer of the framework is based on the assessment of the application in line with the established dependability requirement.

E. To evaluate the framework for its effectiveness and viability.

The dependability assessment framework could be deployed by both the system analyst and the system developer to test for its effectiveness and viability. This process shows the practicability of the framework in assessing real-time IoT applications. In achieving this

objective two major techniques were applied, the use of simulation environment and the use case evaluation method.

1.5. Original Contribution to Knowledge

The original contribution to knowledge of this research study are summarised below:

(i) A detailed understanding of dependability in IoT, with clear measurable attributes

This research study was able to create an understanding of dependability in IoT, with the clear measurable attributes from other related areas of computer science. This include a concise understanding of the factors that needs to be addressed when considering dependability in an IoT application. The dependability of an IoT application can therefore be defined as the ability of the application to provide a service that can be justifiably trusted. The main attributes of dependability in IoT is availability and reliability, where availability is the readiness of correct service and reliability is the continuity of service.

Availability: The availability of an IoT application is the ability of the system to deliver the required service within the required time. A failure in a component of an IoT application will adversely affect the service delivery of the application. The concept of availability in IoT is directly related to reliability.

Reliability: The reliability of an IoT application is paramount to its successful operation, this can be reflected in the energy efficiency and timely packet transmission. In IoT, sensors cooperatively sense, collect and process information in the monitoring environment, there is need for real time acquisition and timely processing of information. Reliability in data transmission is a key determinant of the dependability in an IoT application. However, there is a limited understanding of dependability in the energy efficiency in IoT. This research study was able to create an understanding of processes that contribute to the reliability in energy efficiency in IoT. An increase in IoT devices will potentially produce a large amount of data that will be transmitted through the communication network. Therefore, reducing the energy demand of these devices through effective and reliable transmission network in processing the data as quickly as possible from the sensory devices will adversely improve the reliability of the application.

(ii) A framework for assessing dependability in IoT application

The main contribution of this research study is the development of a dependability assessment framework for assessing the dependability of an IoT application during the

developmental and deployment stage. The objective of this framework is to provide practical ways for system developers and analyst to assess the dependability of an IoT application.

This framework consists of processes and structures that create a logical understanding of the steps involved in the assessment of the dependability of an IoT application. The size of an application is a factor to be considered. IoT applications are complex systems with different variations in the number of components used in the design. An assessment of the application components will create an understanding of the operation. Identifying the impact of the components used in the design of an IoT application is considered as an important factor in ensuring dependability. The dependability requirements in the framework are essential characteristics, required in the effective service delivery.

1.6. Research Scope

The scope of this research study is limited to IoT applications. The dependability assessment framework is considered more suitable for the evaluation of IoT applications. However, the understanding of dependability in this thesis can be used in other emerging related areas of computer science such as Cyber-Physical Systems (CPS), Nano computing systems (NCS) and Machine to Machine communication (M2M), which consist of a similar component in their operations. This research study is focused on ensuring the dependability of IoT applications by looking into reliability and availability. The issue of security, privacy and environmental challenges in IoT are not within the scope of this thesis.

1.7. Thesis Structure

The structures of the chapters in this thesis are described below:

Chapter 2 is an overview of the concept of IoT and its application areas. The key characteristics and challenges were also explored in the sections of this chapter outlining the relevance of dependability.

Chapter 3 consist of the research design and methods that was used in the achievement of this research study.

Chapter 4 is designed to review the concept of dependability and its relation to IoT. The sections in this chapter are structured around dependability and its related concept within the IoT context.

Chapter 5 is focused on the analysis of several existing IoT applications to identify the components used in the design of the applications.

Chapter 6 is about the about the fault tree analysis of the components of an IoT application.

Chapter 7 presents an overview of the dependability assessment framework, the processes, structures and its applicability.

Chapter 8 presents an evaluation of the framework, through series of tests, with an analysis of the results and findings.

Chapter 9 shows a summary of the research achievements, recommendation and limitations of this research study.

Chapter 2. Literature Review

2.1. Introduction

This chapter discusses the overview of the concept of IoT and the related challenges. The application areas and domains of IoT are explored in detail in section (2.4). A detailed review on the dependability issues in IoT application is highlighted in section (2.5) and finally a discussion summary is presented in section (2.6).

2.2. Overview of IoT

The original concept of the IoT was coined by Kevin Ashton and is becoming more of the new technological mainstream. The IoT is a system of interrelated computing devices, that creates a network, where every-day physical objects are connected to the internet. The purpose of this concept is to enable the exchange of information from the sensor node through existing network connectivity and computing capability with a minimal level of human intervention. These devices connect to the network to provide the information they gather from the environment through a sprawling set of technologies. Thus, in effect, the IoT is a combination of a technological push for more and ever-increasing connectivity with everything and anything happening in the immediate and wider environment (Kramp et al, 2013).

Coetzee & Eksteen 2011, describes IoT as a network where objects are uniquely identified and accessible to the network, fusing the digital and physical world. These objects are interconnected with other devices to facilitate the interaction between the digital and the physical world. In the past digital world, a vast majority of internet connections are devices used directly by humans, such as computers and mobile phones. Today, every object can be connected, things can exchange information by themselves and the number of "things" connected to the internet is increasing daily. According to the prediction of Nordrum, (2016); Jiang (2015); Ahmad (2014) & Perera et al (2014); the world will have more than 50 billion devices connected to the internet by 2020, through the expansion of the IoT network.

This is a positive approach in creating a real-time, digital and virtual smart environment, enabling ‘things’ to be connected at anytime and anyplace, with anything and anyone ideally using any path and network (Tan & Wang 2010). This communication network creates opportunities and a valuable contribution to the economic development of the society (Atzori et al 2010).

The network infrastructure of an IoT application has the capabilities, standards and interoperable communication protocols where both the physical and virtual things are seamlessly integrated into the information network (Kranenburg 2007 & Xu et al 2014). The integration of RFID tags, sensors and communication technologies creates the foundation of the were physical objects and devices are associated with the internet to provide high-quality of services to end users. This connectivity will result in a wide range of new services, applications and data, leading to effective running of healthcare services, smart cities and electricity grids (Xu et al 2014).

2.3. IoT Application Areas

The novel paradigm of the IoT has been gaining recognition in the various scenarios of modern wireless communications network in promoting the pervasive presence of intelligent things around us. IoT has the potential to change the world (Aston 2010). Information has always been a key to our society. Development in technology has broadened the path with an opportunity to access the enormous amount of digital information. Virtually every aspect of our lives is becoming transformed by the invention of IoT (Ray 2016). The rapid increase in the number of connected devices to the internet and the significant advances in information and heterogeneous communication technologies have led to great emergence of the IoT. Due to these advances, IoT systems are being heavily used in real world applications as shown below.

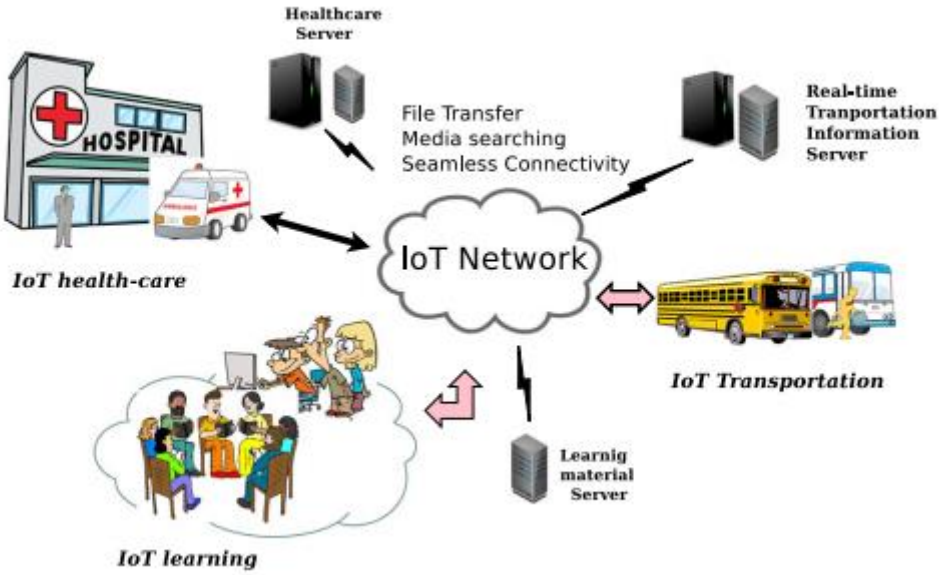


Figure 2.1: IoT and its Application Areas (Source: Gochhayat et al 2019)

Figure 2.1 represents some of the vital areas and application of IoT in the society. This include the transportation sector, health and wellness, manufacturing and agriculture. The invention of the IoT is currently transforming the way we live. The sub-sections below is a description of the application areas of this valuable concept.

2.3.1. Transportation Sector

The implementation of IoT in the transportation sector has an improvement in the satisfaction of mankind (Reddy & Mohan 2018). IoT has an effective impact in the communication, control, and information processes of the transportation sector. The concept of IoT is extended to most aspects of transportation systems. The dynamic interaction between the components of an IoT application, enables inter and intra vehicular communication, smart traffic control, smart parking, electronic, logistic and fleet management, vehicle control, safety and road assistance (Ersue et al 2014). In most of these IoT applications, the automobiles are equipped with sensors, which are connected to the internet and the control systems. IoT also plays important role in the enhancement of road safety for commuters, such as: collision detection, lane change warning, traffic signal control and intelligent traffic scheduling.

The IoT sensors provides vehicular traffic information's which is a substantial source of data for analysing the day to day running of commuters in enhancement of their comfort. The road users can utilise the vehicular traffic information to define the arrival times at their destinations and for monitoring traffic congestions (Talari et al 2017). The introduction of camera-based traffic monitoring system based on the IoT infrastructure, provides a more powerful communication with relevant and real time information. Today, traffic monitoring is conducted by sensors and GPS installed on modern vehicles which creates essential information for authorities and citizens, in the improvement of traffic and the transportation system.

2.3.2. Medical and Health Care

The IoT has created an advancement in a variety of healthcare services (Islam et al 2015). IoT-based healthcare are currently been applied to a various field in the health care sector, including care for paediatric and elderly patients, the supervision of chronic diseases, and the management of private health and fitness, among others.

In patient care, IoT enable healthcare professionals administer the necessary care to their patients through effective management of medication, patients care analysis and provision of personalised therapy. This in turn improves positive patient outcomes and their quality of life (Dimitrov 2016).

IoT network infrastructure are currently enabling remote health monitoring and in emergency notification systems. The remote health monitoring systems are used for the monitoring of patients' physiological status which requires constant close attention (Pang 2013). These monitoring systems employ sensors to collect physiological information which is analysed and stored using IoT gateways. This patient information is then sent wirelessly to caregivers for further analysis and review with a continuous automated flow of the patient health status, thus the quality of care is improved and eliminates the need for a caregiver to actively engage in data collection (Niewolny 2013; Kulkarni & Sathe 2014). The IoT is also playing a major role in the areas of monitoring of diseases, ad hoc diagnosis and providing prompt medical attention to health care patients with diabetes, cancer, coronary heart disease, stroke, chronic obstructive pulmonary disease, cognitive impairments and seizure disorders.

2.3.3. Manufacturing

The IoT enables the quick manufacturing of new products and real –time optimisation of the production process and supply, through the use of sensors and unique identifiers, devices can interact with an intelligent supporting network infrastructure production process is optimised and the entire lifecycle of the product, from production to supply is monitored with transparency about the status of the production unit, the location and disposition of lots, and the production machines. IoT also helps in the digital control and manufacturing process in automating and optimising the mechanical plant safety for effective measurements, automated controls and plant optimisation (Reddy & Mohan 2018).

2.3.4. Agricultural Sector

The advancement in IoT technologies brings a great benefit to the agricultural sector. The introduction of IoT in the agricultural sector plays a major role in modifying agricultural process. In particular, the agro-industrial area, animal farming and environmental fields monitoring, where IoT devices are applied for both diagnostics and control. Smart applications provide information on the products to both the farmers and consumers (Talavera et al, 2017).

The growing landscape of IoT can almost be applied to different sectors of the agriculture field, precision farming is a recent concept of IoT application that is gaining recognition (Krishna, et al 2017 & Sarwat, et al 2018). The IoT provides predictive data that equip farmers with the critical information on the soil, weather and environmental parameters. The driving factor behind the use of IoT in the agricultural sector is a demand for an increase in yields and food production through the optimisation of the available resources (Mekonnen et al 2018).

The implementation of IoT devices in the agricultural sector creates an understanding of the interdependency of energy, water and other required resources, through the continuous real-time monitoring, farmers can predict their yield, optimise water utilisation through smart irrigation control and precisely know when to harvest, thereby the reducing energy and labour input (Hashim et al, 2015; Kodali et al, 2014). In such optimisation of agriculture, installing wireless sensors in the field improves the effectiveness and efficiency of the farmers, enabling the evaluation of the field variables such as soil state, atmospheric conditions, and biomass of plants or animals and assessing the right control for the variables to improve their yields (Pang et al, 2015; Capello et al, 2016; Fang et al, 2014).

2.3.4. Infrastructure Management

Another key application of the IoT is the monitoring and control of the operations of structural infrastructures like bridges and railway tracks (Vermesan et al 2014). The IoT is used for monitoring of events or changes in structural conditions, that can compromise the safety thus increase the associated risk. Smart applications are used for scheduling, repair and maintenance activities, through the coordination of tasks between service providers and users of these facilities. The use of IoT devices for monitoring and operation of these infrastructures will improve the management cost of operation. These systems are designed with the combination of sensors and wireless technologies as a piece of robust networking equipment, to support the reliable communication of the service management of the infrastructure through the application-layer (Reddy & Mohan 2018).

The invention of weather systems using the components of an IoT application, provides accurate data such as the level of temperature, rainfall, solar radiation and wind speed which has a major advancement in creating accurate predictions of the environment. Besides the invention of weather systems, the water distribution systems based on IoT components are essential to the development of every city. The conventional methods of water distribution from the water source to the customer premises are not suitable and efficient, especially for

diagnosing leakages, these water distribution systems uses advance sensors for detecting any failure in the water distribution process and communicates the problem it to the maintenance team through the IoT gateway (Talari et al 2017).

The concept of IoT is improving the day to day living of mankind through its application in several valuable areas of the society as shown in the previous section, table 2.1 presents a summary of the application areas and domain of IoT.

Table 2.1. IoT Application Areas and Domain

Applications Areas	**Domain**	**References**
Medical and Healthcare	Remote monitoring Wireless body area Emergency notification systems	Islam et al (2015) Dimitrov (2016) Pang (2013) Kulkarni & Sathe (2014) Niewolny (2013).
Transportation	Smart transportation Smart Ticketing Smart traffic control Safety and road assistance	Reddy & Mohan (2018) Ersue et al (2014) Talari et al (2017).
Manufacturing	Automated controls Plant optimisation, Health and safety management,	Reddy & Mohan (2018)
Agricultural sector	Animal farming Environmental fields monitoring,	Talavera et al (2017) Krishna, et al (2017) Sarwat, et al (2018) Mekonnen et al (2018) Capello et al (2016) Fang et al (2014) Hashim et al (2015)

		Kodali et al (2014) Pang et al (2015)
Infrastructure Management	Emergency response coordination Weather systems Water Distribution Systems	Vermesan et al (2014) Reddy & Mohan (2018) Talari et al (2017)

2.4. Characteristics & Challenges of IoT Application

The physical layer of an IoT application is made up of smart objects integrated with sensors, this enable the interconnection of the physical and digital worlds allowing real-time information to be collected and processed. The sensors have the capacity to take measurements of the physical property with the unique purpose of monitoring the changes in the physical environment (Xia 2012).

These sensors require connectivity to send the data to the gateways, there is a need for a for an available amount of memory in the gateway to enabling them to receive the sensor data (Patel & Patel 2016). In some cases, a large volume of data will be produced by these sensors which requires a robust and high-performance wireless network as a transport medium to the destination gateway in a timely manner (Patel & Patel 2016). There is a demand for a wider range of IoT services and applications such as high speed transactional, connectivity context-aware applications. Multiple networks with various technologies and access protocols are needed to work with each other in the heterogeneous configuration of an IoT application (Patel & Patel 2016). Figure 2.2 below is a representation of the characteristics of an IoT application.

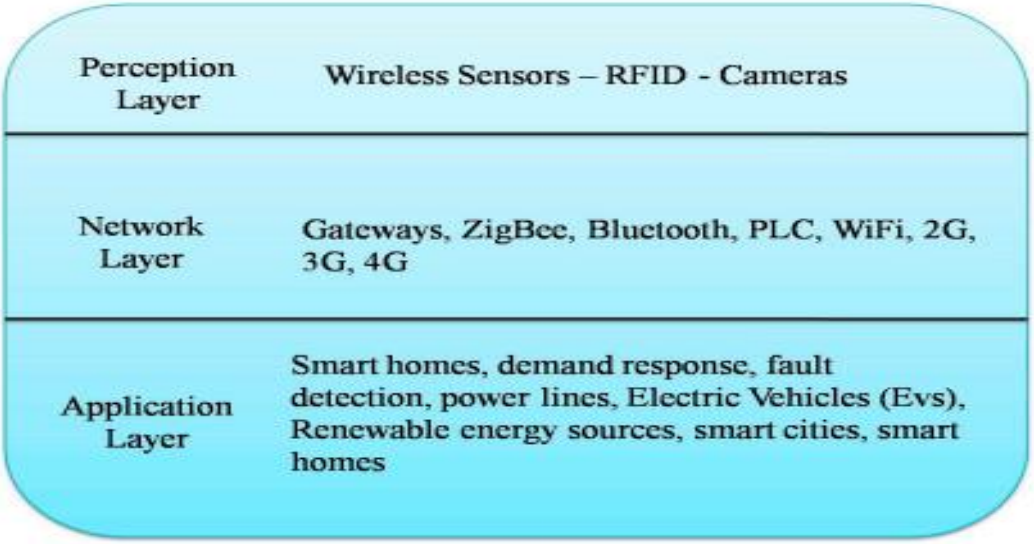


Figure 2.2: Characteristics of IoT Application (Source: Talari et al 2017)

The gateway data management in IoT application, is essential in acquiring the large amount of sensor data. Therefore, the need for a research into the consideration of an effect design of the architectures and protocols for IoT is paramount. The large amount of data streaming from the sensors has an effect of the IoT, this means that a potentially very large amount of information will be injected into the network. The information injected by sensing objects is a concern for the pervasive scenarios of the IoT (Patel & Patel 2016).

According to Kocher (2014), IoT will create a substantial advancement in several computing areas and existing technologies. Elkhodr (2013), stresses that the IoT will expand the use of identification technologies, which are already widely used in several applications, the use of wireless sensor networks, which serves as means to collect contextual data and the service-oriented architectures capabilities. The vision of the future internet will be based on the standard communication of the IoT.

The intelligent and self-configuration of the IoT nodes when interconnected will form a dynamic and global network infrastructure technologies, enabling ubiquitous and pervasive computing scenarios. The IoT is characterised by the things with limited storage and processing capacity, which has consequential issues regarding reliability and performance, except with the integration of cloud computing which has virtually unlimited capabilities in terms of storage and processing power, the true concept of IoT will be unrealistic (Chao 2011 & Zhao et al 2010).

The research study of Atzori et al (2010), indicate that IoT is characterised by a large heterogeneity in terms of devices. This presents different capabilities from the computational and communication standpoints. IoT sensors do not have the continuous power source as they are often placed in constrained environment. Long battery life is a key part to the success of IoT, the computation and information processing relies on the battery (Sachs 2018). IoT devices work as a single, limited purpose which could have customised network interfaces, operating systems, and programming models that make the most efficient use of limited computation, network, and energy resources. The management of such a high level of heterogeneity at both architectural and protocol levels are necessary and poses a great challenge (Miorandi, et al 2012 & Panda 2017).

Energy is a major technological challenge in IoT, and more research is needed to develop systems that are able to save energy during the operational environment (Shakerighadi et al 2018 & Kumar et al 2019). The IoT requires means of minimising the energy to be spent for communication and computing purposes, techniques for energy harvesting will relieve these devices from these constraints imposed by battery operations, energy is a scarce and limiting resource that needs to be handled in IoT applications. Thereby the need to devise solutions that tend to optimise energy usage in IoT is necessary (Sheng et al 2013).

According to Kranenburg & Bassi (2012), the demand in IoT devices will increase to trillions by 2025. It is unlikely that all devices will be connected in a similar topology or rather organised in an hierarchical sub domains, the number of interconnected object will need several order of magnitude, than the current computer internet network. As everyday objects will get connected to a global information infrastructure, the issues of scalability arises at the different levels of operations including: naming and addressing due to the sheer size of the resulting system, data communication and networking due to the high level of the interconnection among the large number of entities (Kranenburg & Bassi 2012 ; Stankovic 2014 & Thakare et al 2016 & Panda 2017).

The applicability of the IoT devices, is a complex mixture of various technology that provides solutions based on the integration of various heterogenous technologies. As described in section 2.3, in most cases, IoT application heavily depend on a network of connected components embedded in the physical objects, such as appliances and medical devices. The functionality and operation of these physical objects is important in communication and connectivity. The complexity of these devices and technology leads to

the challenge of creating a dependable IoT application (Vermesan et al 2014, Silva et al 2013 & Hua-Dong 2011).

As demand for more sophisticated technology in IoT continues, the need to guarantee assurance of service and reliability arises due to the importance of these technologies to human existence. It is deemed necessary that IoT systems is designed and built according to a quality standard with an aim of satisfying the users (Thomas & Rad 2017). The concept of IoT has made great progress and has been widely used in many industries as shown in section 2.3. The dependability of IoT applications in their operations are very important and there is need to increase the performance of these devices under a wide range of environmental conditions (Pereira et al 2014 & Boano et al (2016).

The IoT is a key enabling technology for applications with high societal relevance and impact. These attractive applications represent a long term and are only feasible if underlying IoT technology does not fail. Any failure in meeting the application specific requirements and in conveying information about the state of things and places in a reliable, timely and energy efficient manner may result in high cost, insufficient user satisfaction and physical damage to people of things (Boano et al 2016). Silva et al (2013), stresses that despite the positive prospects of the application, one of the most challenging problem of the IoT system, is the dependability of the applications.

According to Sefan et al (2017), it is important to evaluate the dependability of IoT applications at the early stages of planning and during design phases. The research of Sefan et al (2017), is focused on the sensitive and challenging area of IoT, which is on the measurement of the reliability of components that comprises IoT systems. Furthermore, Sefan et al (2017) proposes sensitivity analysis on the criticality of components of IoT in through an investigation into the architectures and components.

2.5. Dependability Issues in IoT Application

Silva et al (2013), stresses that despite the high degree of applicability of IoT application the IoT network still faces various challenges highlighting the dependability as one of the major ones. Stankovic (2014), pointed out that dependability has not been fully defined with in the IoT settings, that the IoT currently encounter a number of failures in the transmission of data and in the communication links (Ojie & Pereira 2017).

The research on dependability issues in IoT is still at its infant stage considering that IoT as a concept is still developing. There is need for the issue of dependability to be addressed in IoT considering the current trends and advancement in the vision and it impact on the future scenarios (Vermesan et al 2011). Avizieni et al (2014), insist that IoT applications should be able to provide dependable services that can be justifiably trusted and a system that can be relied upon under a defined functional and environmental conditions over a determined period with any or no occurrence of failures in its critical operation. Macedo et al (2014), highlighted that the dependability of IoT application can be enhanced with a strategy of an alternative spare device that can be used when there is a failure in any of the devices in the application to estimate the reliability and availability of IoT applications through a consideration of the redundancy aspects on the components in providing valuable implementation decisions at the planning and design phases.

According to Boano et al (2016), IoT applications are in several domains such as surveillance of civil infrastructure, smart grids, and smart healthcare which comprises of various complex components. Guaranteeing the dependability of these various applications is a challenge. The IoT is comprises of constrained resources, vast computing devices and operates in a harsh environmental condition (Boano et al 2016).

IoT application provides critical every-day services, that require a dependable robust system that enables user's satisfaction. Today's approach to the construction of an IoT application, do not guarantee dependability (Witrisal et al 2019). Wireless technologies suffer from physical impairments e.g. multipath propagation and interferences from competing transmissions, as well as from the effect of temperature variations and other environmental properties. This impairs the accuracy, latency, loss, and high energy consumption (Witrisal et al 2019).

According to Baunach et al (2019), a central requirement of tomorrow's IoT the dependability of devices so that operations are completed within a guaranteed response time. The key objective is to improve the hardware and software to allow dependable software execution in the IoT setting that address the inherent complexity of mixed-criticality of real-time applications. The network communication between smart items is prone to errors and likely to be corrupted by unpredictable distortions and losses (Horn et al 2019). The topology of feedback loops might change abruptly due to loss of connection. These are inherent to the IoT, which raise the need for an innovative robust method for the design of the network.

In the IoT, ‘things’ collaborate to provide a service, the type and number of devices involved in such collaboration could be large, which might comprise the dependability of the application, even if the individual devices are dependable by themselves, their composition may impair the dependability of the application. This could be as a result of energy failure of the devices or failure in the communication protocols (Bloem et al, 2019). Bloem et al, (2019) propose to build a systematic tool to ascertain whether the system design of an IoT application, can function properly in diverse environments. Secondly, to assess the composition of individual components, to ascertain if they will act correctly during the system operation. This intended technique will guarantee the dependability of IoT application under an unrealistic environmental condition (Bloem et al, 2019).

Despite the positive drive into addressing the issues of dependability in IoT applications, there are low research outcomes and solutions to this challenging concept. However, the research communities are coming up with various objectives in addressing the dependability of IoT application as reflected, in Horn et al (2019); Bloem et al (2019); Baunach et al (2019); Witrisal et al et al (2019), which are still in elementary stages, thus their ideas are presented as an insights of theirs of thoughts towards the enhancement of dependability in IoT.

2.6. Summary

In this chapter, a literature review was conducted on the overview of IoT and its application areas. This includes the key characteristics and technology that make up the composition of an IoT application. Thereafter the challenges of IoT was explored, there are still many unresolved issues, this review was able to highlight the importance of dependability in IoT applications. The research community is beginning to see dependability as a major concern to the advancement of IoT. However, research on dependability in IoT application is still at an infant stage**.**

Chapter 3. Research Methodology

3.1. Introduction

This chapter presents the justification of the adopted research methodologies used in accomplishing the aim of this research study. The sections in this chapter is organised to address the different research objectives. The first section (3.2), involves an in-depth inquiring to understand the concept of dependability in fulfilling objective (A). Section (3.3), describes the case analysis method used in addressing objective (B). Section (3.4), is a description of the fault tree analysis techniques used in analysing the components of an IoT application in accomplishing objective (C) and (D). This is followed by section (3.5), and (3.6), which outlines the techniques and methods used in the evaluation process (Objective E). Thereafter a concluding summary is presented in section (3.7), the figure below shows the research methods and techniques adopted in this research study.

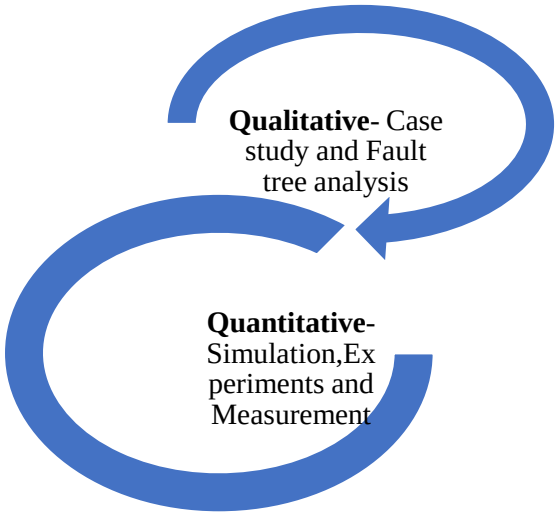


Figure 3.1: Adopted research methods for this research study

3.2. Analysis of IoT Application

In the achievement of objective (B) of this research study which involves the analysis of the characteristics and the functional requirements of an IoT application, a critical analysis was

conducted on existing applications to identify the components used in the design through case study approach, the components were examined to understand the constituents of the application.

3.2.1. Case Analysis

The case study analysis is a qualitative research method, which is often used in studying variety of systems to get the required information on the system design. Creswell (1996) & Gable (1994), identify three strengths of case study research: (1) the researcher can study information systems in a natural setting and learn about the state of the art technology; (2) the method allow the researcher to understand the nature and complexity of the process and (3) valuable insights can be gained into the systems operations.

According to Yin (2009), the case study method can be used in getting a holistic and meaning full characteristics of real-life contexts such as a system operational life cycle. Case studies are suitable for exploration and classification in the development stages of the knowledge building of a research study. The use of the case study approach in this thesis is used to create a degree of understanding of the components that make up an IoT applications. This approach was utilised through an analysis of the components in the IoT architecture used in the design of IoT applications.

3.3. Analysis of Dependability Modelling Techniques

Dependability modelling techniques can be utilised in every phase of the system or component including development, operation and maintenance (Avienesis 2004 & Ahmed et al 2016). Fault tree analysis (FTA) and Reliability block diagram models (RBD) are usually used to provide reliability and availability estimates for both early and later stages of the development. On the other hand, Markov Chain based models are mainly used in the later development phase to perform trade-off analysis among different design alternatives when the detailed specification of the design becomes available. In addition, when the system is deployed, these modelling techniques can be beneficial in order to estimate the frequency of maintenance (Bernadi et al 2012).

Some of the most widely used modelling techniques are reviewed below. It must be noted that not all techniques have been utilised in IoT context. In addition to these techniques use of simulation methods has also been identified in the literature. A detailed analysis of simulation tools in relation to their applicability to IoT approaches is provided in section x.

3.3.1. Reliability Block Diagrams

Reliability Block Diagrams (RBD) are graphical structures consisting of blocks and connector lines. The blocks usually represent the system components and the connection of these components is described by the connector lines. The system is functional if at least one path of properly functional components from input to output exists otherwise it fails. This information is then utilised by the design engineers to identify the appropriate RBD configuration in order to determine the overall reliability of the given system (Avienisi 2004, Bernadi et al 2012 & Ahmed et al 2016).

3.3.2. Fault Trees Analysis

Fault Tree Analysis (FTA) is a graphical technique for analysing the conditions and the factors causing an undesired top event, i.e., a critical event, which can cause the whole system failure upon its occurrence. These causes of system failure are represented in the form of a tree rooted by the top event. The preceding nodes of the fault tree are represented by gates, which are used to link two or more cause events causing one fault in a prescribed manner. For example, an OR FT gate can be used when one fault suffices to enforce the fault. On the other hand, the AND FT gate is used when all the cause events are essential for enforcing the fault. Besides these gates, there are some other gates, such as exclusive OR FT gate, priority FT gate and inhibit FT gate, which can be used to model the occurrence of faults due to the corresponding cause events (Avienisi 2004,Bernadi et al 2012 & Ahmed et al 2016). Once the fault tree model is constructed, both qualitative and quantitative analysis can be carried out. A qualitative analysis in this context allows the identification of all combinations of basic failure events, known as cut sets, which can cause the top event to occur.

3.3.3. Markov Chain

A Markov chain is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event which are used to point the transition from one state to another. The initial state and the probability represent the starting state and the transition probability from state to state respectively. The process starts from an initial state and transitions from the current state to the next state occur on the basis of transition probabilities, which only depend upon the current state based on the Markov or the memoryless property. Markov chains are usually classified into two categories: Discrete Time Markov Chains (DTMC) and Continuous Time Markov Chains

(CTMC). Markovian models are frequently utilized for reliability analysis in scenarios where failure or repair events can occur at any point in time. Markov modeling has also been utilised for analysing the dynamic behaviour of the other reliability models (Avienisi 2004,Bernadi et al 2012 & Ahmed et al 2016).

3.3.4. Model Checking

Model Checking allows to describe the behaviour of a given system in the form of a state machine and verify its temporal properties in a rigorous manner. Probabilistic model checking extends traditional model checking principles for the analysis of Markov Chain and allows the verification of probabilistic properties. Some notable probabilistic model checking include PRISM and ETMCC (Rodrigues et al 2012). Probabilistic model checking techniques have been considerably adopted to verify the reliability and availability properties of many systems. The PRISM model checker has been utilised for the reliability and safety analysis of applications by augmenting it with a simulation tool (Rodrigues et al 2012, Ahmed et al 2016, Avienisi 2004 & Bernadi et al 2012). An analysis of the dependability modelling techniques is presented in table 3 below.

Table 3.1 Analysis of the Modelling Techniques

Techniques	Access	Academic	Dependability Attributes		Remarks	
			Reliability	Availability	Qualitative	Quantitative
RBD	Open	Yes	✓	X	Not Available	Success and Failure Probability
FTA	Open	Yes	✓	✓	Root Cause Analysis	Success and Failure Probability
MCP	Open	Yes	✓	X	Not Available	Probability of Failure
MC	Open	Yes	✓	X	Not Available	System Probability

The critical areas of consideration in this analysis was the accessibility of the tool, the academic acceptance and usage in existing literature, the potential of addressing the dependability attributes and relevance to both qualitative and quantitative research.

The result of the analysis in table 3.1, shows that fault tree analysis technique is a tool that can be used in the modelling of dependability of IoT application. FTA is an open access tool that has been widely used by academics to analysing the dependability of IoT application (Kabir, 2016 & Papadopoulos, 2012; Papadopoulos et al 2011). The analysis reveals that Markov chain is a mathematical process that is used quantitively in assessing the probability of failure of a system while Reliability block diagram (RBD) is a diagrammatic method for showing how component reliability contributes to the success or failure of a system. This an open access tool specifically designed for addressing reliability in computer system through a quantitative approach unlike FTA that can be used for addressing reliability and availability both qualitatively and quantitatively. A detailed review of the components and applicability of fault tree is shown in session 3.4 below.

3.4. Fault Tree Analysis

The Fault Tree Analysis (FTA) is another well-established and well understood technique widely used to determine system dependability (Kabir, 2016 & Papadopoulos, 2012). In fault trees, the logical connections between faults and their causes are represented graphically. FTA is deductive in nature, meaning that the analysis starts with a top event (a system failure) and works backwards from the top to determine the root cause (Papadopoulos et al 2011; Aizpurua & Muxika, 2013). The results of the analysis show how different component failures or certain environmental conditions can combine together to cause the system failure. The qualitative analysis is performed by reducing fault trees to minimal cut sets (MCSs), which are a disjoint sum, consisting of the smallest combinations of basic events that are necessary and sufficient to cause the top event (Kabir 2018, Gudemann & Ortmeier, 2010).

According to Avienisi et al (2004) & Ahmed et al (2006), the fault tree is a tool that can be used for the evaluation of the risk of computer systems. Fault Tree is a graphical technique for analysing the conditions and the factors causing an undesired top event, i.e., a critical event, which can cause the whole system failure upon its occurrence (Bernadi et al 2012). The cause of system failure is represented in the form of a tree rooted by the top event. The preceding nodes of the fault tree are represented by gates, which are used to link two or more events causing one fault in a prescribed manner. For example, an OR FT gate can be used when one fault suffices to enforce the fault. On the other hand, the AND FT gate is used when all the cause events are essential for enforcing the fault. Beside these gates, there are some other gates, such as exclusive OR FT gate, priority FT gate and inhibit FT gate, which

can be used to model the occurrence of faults due to the corresponding cause events (Avienisi et al 2004, Bernadi et al 2012 & Ahmed et al 2016).

3.4.1. Standard Fault Trees

Standard fault trees (SFT) are usually performed at the qualitative level where each MCS contain a single event or multiple events combined by logic gates. The order of a minimal cut set defines the number of basic events that contribute to that minimal cut set. A 1st order MCS consists of a single basic event, i.e., a single failure event alone can cause the system failure. Therefore, this single component becomes a candidate for upgrade or to replicate. On the other hand, a 4th order MCS contains four basic events. The lower the order of a MCS the higher the importance of that MCS (Kabir, 2016). There are many algorithms available to perform the qualitative analysis of fault trees. A comprehensive list of these algorithms is available in Ruijters & Stoelinga (2015).

3.4.2. Cut Sets

Cut sets are a top-down approach and it is one of the primary SFT algorithms. Many other algorithms are developed based on this algorithm (Fussel & Vesely, 1972). This algorithm starts its operation with the top event of the fault tree and recursively explores the cut sets by expanding the intermediate events into their contributing basic events. This process continues until all the intermediate events are expanded and no more basic events are left in the fault tree.

3.5. Simulation & Experiments

Simulation is the quantitative approach that is utilised in this research study for the measurement of the effect of the dependability assessment framework, to ensure it fulfils the purpose of design (Maxion 2009 & Ary et al 2010). This method enables a critical evaluation and is primarily an investigatory approach of developing knowledge through measurements and observations (Creswell 1996). Simulation is a method that has been widely used in the computer science research and other related studies (Angelo et al, 2016). This enables the possibility to investigate systems by setting up experiments and for modelling of the system behaviour, over a given time through the creation of experimental environments of a real-life scenario that can be used in practice (Ayash 2006; Gupta et al 2016; Kellner et al, 2010).

To understand and evaluate the functionality of a system component, it is important to perform experiment or series of tests. The use of the simulation method in this research study is for the evaluation of the dependability assessment framework. This technique was used in the evaluation of the components of an IoT application. The simulation tool used in this research study is OMNeT++ simulation environment. OMNeT++ is a C++-based discrete event simulator for modelling communication networks (Varga & Hornig 2008). OMNeT++ model consists of modules that communicate packets between the source node to the destination gateway. The active modules are termed simple modules; they are written in C++, using the simulation class library (Varga & Hornig 2008). The simple modules are further grouped into compound modules and so forth; the number of hierarchy levels is not limited the generated packets sent through the wireless network connections that span between the modules in the network setup.

3.6. Use Case Analysis

The use case analysis technique is used in achieving the effectiveness of the dependability assessment framework, in identifying the dependability requirements and in defining the processes in an existing IoT application, to ensure it will ultimately fulfil purpose of deployment. This process begins with identifying a typical IoT applications that can be used for modelling. In this research study, various existing IoT application in different domains are used as a use case examples, to reflect the true characteristics of a real-life system. The overall goal of this step is to provide enough details to understand the effectiveness of the dependability assessment framework.

3.7. Summary

The methods and techniques used in actualising the aim of this research study is discussed in detail in this chapter. In understanding the concept of dependability in IoT application, a literature review was conducted to get the perceptions of other scholars. A case analysis method is a viable approach in analysing existing IoT applications, in understanding the nature of the system design. The fault tree analysis approach is a logical method for analysing the root cause and consequences of failures in an IoT application. The simulation experimental method and use case analysis, are feasible approach for the evaluation of the dependability assessment framework.

Chapter 4. Understanding Dependability in IoT application

4.1. Introduction

The focus of this chapter is to create an understanding of dependability in IoT application and its importance to IoT. The terminology of dependability in the area of computer science is used non-uniformly by many authors. Section (4.2), is structured around dependability and its related concept. An in-depth understanding of the similarities of the concept of survivability and trustworthiness to dependability was reviewed. This followed by section (4.3), with a discussion on the dependability attributes and their relationship to IoT application. The relevant dependability attributes in IoT application was ascertained in section (4.4), 4.5 the dependability threats in IoT, session 4.6 and 4.7 is centred around the dependability attributes in IoT. Finally, this chapter concludes in section (4.8), with a discussion on summary of the findings.

4.2. Dependability and its related concept

In early 80s, dependability in computer systems emerged, with a consistent set of concepts and terminologies (Avizienis et al, 2004). A systematic exposition of the concepts of dependability in computer systems can be view differently depending on the application (Laprie, 1990). Therefore, there are numerous view points and definitions of dependability in computer science, but they are all complementary with similar attributes (CCITT 1984, IEC 1992, ISO 1990, Avizienis et al 2004 & Laprie 1995). In this respect, the ISO/CCITT definition is consistent with the definition given in (Hosford 1960), for dependability: “the probability that a system will operate when needed”. In the area of IoT, dependability has no concise definition, as the issue of dependability in IoT is an emerging concept, hence the definition in this research study is adopted from Laprie (1995) on computer system, which is consistent with Hosford (1960).

The concept of dependability is to ensure that computer systems are designed and deployed in an effective manner to be able to function appropriately within their specified life span with the lowest minimal failure or fault rate (Laprie 1995). An IoT application as described

above, depends on connectivity and communication between embedded objects. The essence of dependability is to ensure reliable service.

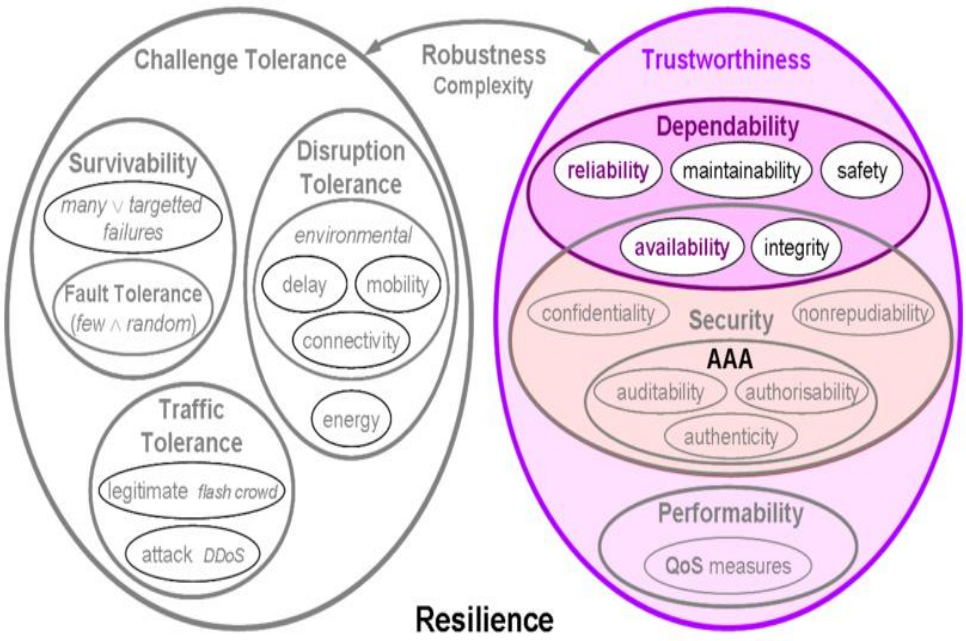


Figure 4.1: Dependability and related concept (Source: Laprie 2008)

As shown in the above figure (4.1), dependability is a property of a system that can be relied upon, with the delivery of services that can be justifiably trusted. The main attributes of dependability is clearly centred around the notion of availability and reliability which clearly state the continuous operation of the system to produce services that can be justifiably trusted (Avizienis et al 2004 & Laprie 1996). These attributes will be explained further in details in the later sections of this chapter. There are other similar and related concepts to dependability such as survivability and trustworthiness which needs to be reviewed to get the true understanding of the relationship between these concepts and dependability.

4.2.1. Dependability in IoT

The complexity of the design and deployment of IoT application brings the challenge of dependability in IoT (Boano et al 2016). At the same time, although sever attempts have been made to define dependability in IoT application, there is no concise definition of dependability in the area of IoT (Woo et al 2018 & Kharchenko et al. 2019).

Dependability in IoT according to ISO/IEC 60300 refers to reliability of the system. This definition is centred around the end to-end quality assurance that guarantees that every component that constitute the system works efficiently and effectively within the period it is developed to function (Thomas & Rad 2017). According to Stefan et al (2017), the

dependability of an IoT application can be viewed as the successful operation of the sensor nodes and the communication links in delivery of the required service. Silva et al (2013), defines dependability in IoT as a system that can be relied upon under defined functional and environmental conditions over a determined period.

In the context of this thesis, dependability in IoT is defined as the ability of a system to deliver a service that can be trusted with no critical failures in its operation. "This implies the ability of a system to deliver a service that can be justifiably trusted. This is the reliability and availability of a system operation" (Avizienis et al 2004 & Woo et al 2018). In other words, it is the extent to which a system can be relied upon, to perform exclusively and correctly under a defined operational and environmental condition within a specified period of time. This notion is centred around the fundamental concepts and initial definition of dependability of computer systems by Avizienis and Laprie, which is now been integrated into other areas of computer science with various definitions (Kharchenko et al. 2019; Power & Kotonya 2019).

The following sections below present the definition of dependability in other computing paradigms, followed by figure 4.1 summarising the key definitions and dependability attributes.

4.2.1.1. Dependability in Distributed Systems

Dependability in distributed systems is defined as the trustworthiness of hardware and software systems, so that reliance can be placed on the service they provide (Slater 1998 & Michel 1997). The main dependability attributes commonly known and accepted in distributed systems are availability, reliability, safety, and security.

Reliability is the probability of a system functioning correctly over a given period of time, and the most difficult part of any distributed system is to coordinate the computations so that a correct result is found.

Availability is the probability of a system functioning correctly at any given time, and distributed systems typically have the greatest advantage over non-distributed systems in this area. The greater redundancy and reconfigurability of distributed systems allow for very high availability to be designed into distributed systems.

Security, the ability of a system to protect the data and identities of its users, is the aim of most multi-user distributed systems. Often there is information which must remain private,

or system capabilities which should only be used by certain users and ensuring security in the system is a difficult task. It is an emergent property, dependent upon all the components and interactions of the system, and typically comes at the cost of ease of use or performance.

Safety, the ability of a system to avoid damages to life, property, or the environment, is typically not an issue in distributed systems. Distributed systems are been design safety-critical, the issues pertaining to safety in distributed systems are usually not applied.

4.2.1.2. Dependability in Grid Computing

Grid computing is an extension of distributed computing environment, where geographically distributed resources are shared, selected, and aggregated based on the availability, performance, and capability. Grid computing systems are distinct from current distributed systems as its focus is on sharing of resources (Nazir et al. 2012). The purpose of grid computing is to eliminate the resource island and to make computing services ubiquitous. With grid technologies, it is possible to construct large-scale applications over the grid environments (Guimaraes et al 2013; Haider & Nazir 2016 & Wang et al 2018). The computing resources are highly heterogeneous, the processes and communications have a significant impact on its dependability. (Nazir et al. 2012; Moon & Youn 2015).

Dependability in Grid computing is defined as the probability of successful running of a given task (Haider & Nazir 2016). This can be referred to as the successful execution of a task without failure in its operation. Failure probability in grid computing environments is potentially high due to its heterogeneous nature as compared to other conventional parallel computing environments (Haider & Nazir 2016 & Wang et al 2018). Therefore, it is critical to derive measures to ensure dependability in Grid computing. In a large-scale system many nodes performing tasks for applications are related to computation, I/O, network and communication, which pose an increase in the probability of failures (Jafarlou 2012 & Egwutuoha 2014). Faults are unavoidable in a complex distributed environment like grid that is scalable and heterogeneous and the diagnoses of faults in such an environment is a challenging task (Gu et al 2013).

The design goals of a dependable grid computing system include availability, reliability, continuity of service, quality of service, flexibility, and adaptability (Haider & Nazir 2016). However, there are many challenges to construct dependable grid services, this include the failure of a power leading to power loss of one part of the distributed system; physical damage to the grid computing component as a result of natural events or human acts; and

the failure of network system or the application software which could lead to the loss of the service. Due to the diverse failures and error conditions in the grid environments, the dependability in the development, deployment and execution of an applications over the grid technology remains a challenge (Balaji et al 2012).

4.2.1.3. Dependability in Cloud Computing

With the continuous growth of application requirements and a significant advancement in the research of cloud computing systems, a large number of cloud computing systems nowadays are based on different structures and virtualization technologies which are still being developed. However, dependability of cloud computing system is always a critical issue for all the cloud service providers, brokers, carriers and consumers around the world (Pan & Hu 2014, Mesbahi et al 2018).

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to the configuration of computing resources which include, networks, services, storages, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction. There are a various of types of failure modes that may affect the dependability of a cloud service, including hardware failures, software failures, management system failures, human operational faults and environmental failures (Joshi et al 2009, Quan 2014 & Abohamama 2017).

Dependability in cloud computing is normally defined as "the ability of the system to perform its required service" (Mesbahi et al 2018). Dependability is a term in cloud computing to describe the time-dependent characteristics associated with the performance of a system, it includes characteristics such as availability, reliability, and security under given conditions of use and maintenance support requirements. Where availability is the ability to be in a state to perform as required. A system's availability varies across actors with different desired levels of service. While reliability is ability to perform as required, without failure, for a given time interval, under given conditions. The system will provide actor desired levels of service with respect to a system's operational profile over a given period of time (Mesbahi et al 2018, Pan & Hu 2014).

4.2.1.4. Dependability of Wireless Sensor network

The dependability of wireless sensor network (WSN), is a property that integrates the attributes needed for the application to be justifiably trusted (Elghazel et al 2015). It is usually defined as a characteristic that enables a WSN application to deliver the required

service within the stipulated time without failure in the operation of the system and the ability of the system to avoid failures that are more frequent or more severe and outage durations that are longer than is acceptable to the users (Bruneo et al 2010 & Wang 2017).

According to Elghazel et al (2015), developing a dependable WSN starts with defining the dependability requirements of users. In order to satisfy these needs, it is crucial to understand the causes of network failures from delivering a correct service.

The attributes of dependability in WSN can vary in numbers and degree of importance considering the nature of the application and the intended service (Elghazel et al 2015). The network, thus, is made dependable by adjusting the balance of the techniques to be employed according to the user's needs. However, in the classical definition of Avizienis et al (2004), a network is considered as highly available if its downtime is very limited. While reliability in WSN, is the ability of the system to deliver the correct service. The main purpose of the design of a WSN, is the correct delivery of the data packets from the sensor node to the end user. Thus, the reliability of WSN is centred around the effective transmission of data and can be classified into different levels which include packet reliability, event reliability, Hop-by-Hop reliability and End-to-End reliability.

Table 4.1. A Summary of the Dependability in other Related Computing paradigms

Computing paradigms	**Distributed Systems**	**Grid Computing**	**Cloud Computing**	**Wireless Sensor Network (WSN)**
Dependability Definition	Dependability in distributed systems is defined as the trustworthiness of hardware and software systems, so that reliance can be placed on the	Dependability in Grid computing is defined as the probability of successful running of the given task. This can be referred to as the successful execution of a	Dependability in cloud computing is normally defined as "the ability of a system to perform as and when required. Dependability is a term in cloud	Dependability in WSN is a characteristic that enables a WSN application to deliver the require service of the application with the stipulated time

	service they provide	task without failure in its operation.	computing to describe the time-dependent characteristics associated with the performance of a system	without failure in the operation of the system.
Dependability Attributes	-Availability -Reliability -Safety -Security.	-Availability -Reliability -Continuity quality of service (QoS) -Flexibility -Adaptability.	-Availability -Reliability -Security	-Availability -Reliability -Safety -Integrity -Maintainability

4.3. Survivability of Computer Systems

Survivability as defined by Fisher et al (1997), is the ability of a system to provide essential services in the presence of attacks and recover full services in a timely manner in the presence of threats such as attacks or large-scale natural disasters. The ability of an entity to continue to meet its functional requirements during events, such as cyber-attacks, physical attacks, natural disasters, and traffic overloads. Survivability is a subset of resilience. For a given application, survivability can be quantified by specifying the range of conditions over which the entity will survive, the minimum acceptable level or post-disturbance functionality with the maximum acceptable downtime.

The term downtime, in survivability is used to refer to periods when a system is unavailable. Downtime or outage duration refers to a period of time that a system fails to provide or perform its primary function. The system unavailability are related concepts. The unavailability is the proportion of a time-span that a system is unavailable or offline. This is usually a result of the system failing to function because of an unplanned event, or because of routine maintenance. This term is commonly applied to networks and servers.

The common reasons for unplanned outages are system failures or communications failures. The concept of survivability is important to build survivable systems, to state accurately what we mean by a system being survivable, we cannot determine whether we have made a system that is survivable (Ellison et al. 1999).

The definition and analysis of survivability requirements is a critical step in achieving system survivability (Linger 1998). The figure below depicts an iterative model for defining these requirements. Survivability must address not only requirements for system functionality but also requirements for system usage, development, operation, and evolution. Thus, five types of requirements definitions are relevant to survivable systems. These requirements are discussed in detail in the Fisher (1997).

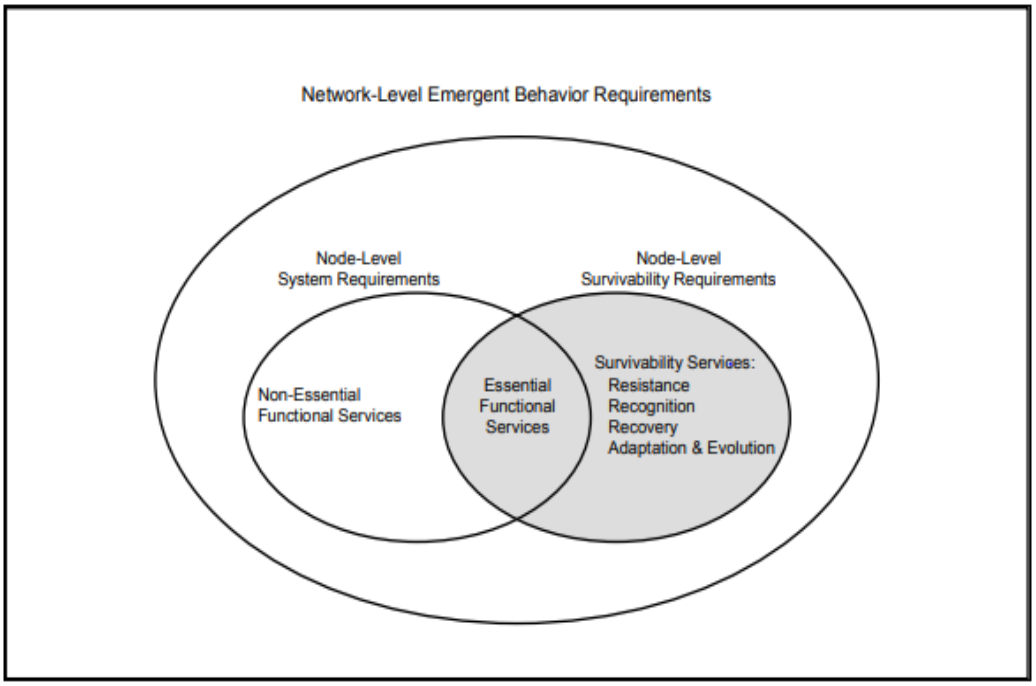


Figure 4.2: Survivability Requirements with System Requirements (Source: Fisher 1997)

As shows in (Fig 4.2), survivability has its requirements on the node and network performance. These requirements include the resistance to, recognition of and recovery from intrusions, compromises, adaptation and evolution to diminish the effectiveness of future intrusion attempts. The system requirements are organised into essential services and non-essential services. These essential services must be maintained as the severity and duration of intrusion to the system. Thus, definitions of the requirements for essential services must be augmented with appropriate the survivability requirements (Fisher 1997).

4.4. Trustworthiness of Computer Systems

According to Avizienis et al (2004), trustworthiness in computer system is the assurance that a system will perform as expected despite environmental disruptions, human and operator error, hostile attacks, and implementation errors. Trustworthy systems are designed to produce expected results that will not be subject to subversion (Schneider 1998). Trustworthiness of software systems are determined, by the following characteristics: correctness, safety, quality of service (performance, reliability, availability), security, privacy, performance, and certification. A full description of the attributes of a trustworthy system is available in the research study of Becker et al (2006). The figure below represents the attributes contributing to trustworthiness, on the baseplate of component-based software engineering (Hasselbring 2002).

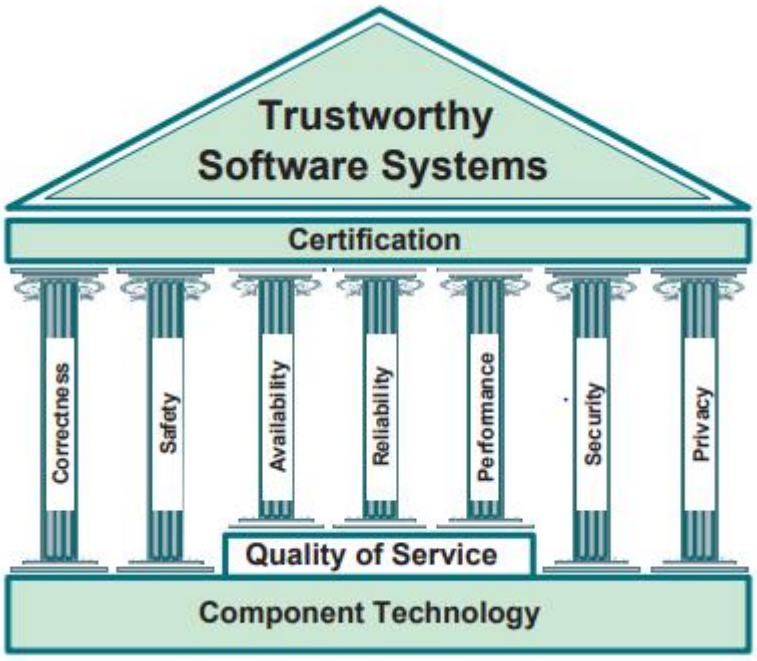


Figure 4.3 The Five Pillars of a Trustworthy Software System (Source: Becker et al 2006)

The research study of Becker et al (2006), categorically states the attributes of a trustworthy system is about the correctness of service with the absence of faults in the system where safety is the absence of catastrophic consequences on the environment. Becker et al (2006), further classified quality of service in trustworthy systems in regard to the same attributes of dependability in Avizienis et al (2004) as availability, reliability and performance. The full description of trustworthiness in computer system, is available in the research study of Beker et al (2006), which clearly shows the relationship with the dependability attributes despite

they have different goals and achievements in computer systems. Table 4.2 below shows an analysis of the similarities of these concepts.

Table 4.2. The Concept of Dependability, Survivability and Trustworthiness

Concept	**Dependability**	**Survivability**	**Trustworthiness**
Goal	The ability of a system to deliver a service that can be justifiably trusted.	The ability of a system to provide essential services in the presence of attacks and recover full services in a timely manner in the presence of threats such as attacks or large-scale natural disasters.	The assurance that a system will perform as expected despite environmental disruptions, hostile attacks, security and privacy and implementation errors.
Attributes	Reliability Availability Related Attributes: Safety Integrity Maintainability	Resistance Recognition Recovery Adaptation	Correctness Safety Availability Reliability Performance Security
References	Avizienis et al (2004) Hosford (1960) Laprie (1995) Prasad et al (1995)	Avizienis et al (2004) Ellison et al (1999) Fisher (1997) Linger (1998)	Avizienis et al (2004) Becker et al (2006) Hasselbring (2002). Schneider (1998)

The above table shows the similarities in these concepts and their relationship to dependability. Prasad et al (1995), indicate that the original attributes that make up dependability in computer systems is availability and reliability as that was the initial attributes that defines dependability traced back to the definition given by the international organization for telephony CCITT (1984), at a time when availability was the main concern to telephone operating companies. The ISO definition is clearly centred upon availability. The other attributes such as safety, maintainability and integrity are related concepts to survivability and trustworthiness.

4.5. Dependability Threats

The threats to the dependability of a system are faults, errors and failures. Threats are things that can affect a system and cause a drop in its dependability. A system can be classed to be a failure either because it does not comply with the specification, or because the specification did not adequately describe its function. An error is that part of the system state that may

cause a subsequent failure: a failure occurs when an error reaches the service interface and alters the service (Avizienis et al 2004).

Failure occurs when the given system fails more frequently or more severely than the acceptable level to the user(s). By propagation, several errors can be generated before a failure occurs in the system. A failure occurs when an error is propagated to the service interface and causes the service delivered by the system to deviate from the correct service the failure of a component causes a permanent or transient fault in the system (Laprie 1996 & Avizienis et al 2004). However, some failures in the system can lead to the entire failure of the system, therefore the different levels of criticality of failures are explored in the subsequent sections below, which shows a detailed understanding of the types of failures.

A fault is active when it produces an error, otherwise it is dormant (Laprie 1996 & Avizienis et al 2004). Faults can be classified according to the behaviour of the failed components, failures in a system can be classed on different levels which are distinguished as crash, omission, timing, response and byzantine.

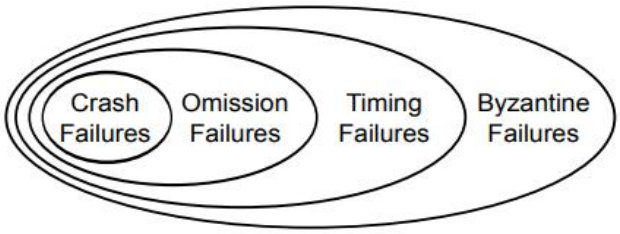


Figure 4.4. Failure Class Hierarchy (Source : Cristian et al 1995)

The failures classes in the above figure, form a hierarchy that creates a design of a fault tolerant system to avoid service failures in the presence of faults. Fault tolerance is the property or mechanism that enables a system to continue to operate properly in the event of the failure of some of its components, unfortunately it is not possible to avoid failures. Instead the aim is to build systems to minimise the impact and criticality of failures when they do occur (Cristain 1995, Zhuo et al 2015 & Lin et al 2019). The subsequent sections explore the classes of failures.

4.5.1. Crash Failures

The most restricted class of failure is the crash failures which are those failures that occur when a component or system is halt (Rohr 2015). Crash failure is considered as the cause of a component to halt or lose its internal state. This is a subclass of timing failures which consist of cases in which the component answers a request too late or rather too early. Another categorisation of crash failure is presented by Avizienis et al (2004) which distinguishes failures based on whether the content or timing behaviour of a system and whether output deviates from the expected behaviour. Availability is commonly defined as the average (overtime) probability that a system or a capability of a system is currently functional within a specified timeframe (Musa et al 2004 & Rohr 2015).

Different types of crash failures can be distinguished according to how much of the internal state and whether the component restarts. This include amnesia-crash, partial amnesia crash which causes a component to lose some of its internal state, a pause-crash halts a component for a certain period of time without loss of the internal state, while the halt-crash stops a component from permanently working.

4.5.2. Omission Failure

When a process or channel fails to do something that it is expected to it is termed an omission failure. A failure is considered as an omission failure, if it causes a component to omit response to an input. A set of crash failures is contained in the set of omission failures as each crashed component cannot respond to the inputs and a component may omit responses although not crashed. Omission failure can be in the process or communication (Rohr 2015).

Process omission failures occurs when a system fails or crash with no further progress on its services. A crash is clean, if the process either functions correctly or has been halted. A crash is termed a fail-stop, if other processes can detect with certainty that the process has crashed. While the communication omission failures occur in the sending process (send-omission failures), the receiving process or in the channel (channel omission failures).

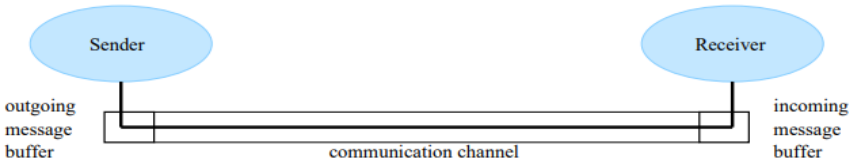


Figure 4.5. Communication Omission Failure (Source: Franke 2006)

A system model consists of a finite set of processes that communicate and synchronise by sending and receiving messages through a communication channel. The underlying communication channel needs to be failure free to ensure there is no alteration, loss or duplication of messages. As shown in the figure above, omission failure be as a faulty process that occur as a result of when a faulty node crash and omits sending the messages that it was required to send. Communication omission failures occur either due a crash of the system or communication channel failures which result to the loss of messages in the sending process (the outgoing message buffer) to the receiver (Hadzilacos 1985, Flocchini & Gasieniee 2006)

4.5.3. Timing Failure

This is a service failure that illustrates the broad range of possible deviations from the intended system functionality. Services which involves actions within a specified time frame may suffer from timing failures if the actions are not executed. The early delivery of information shows the justification of a correct services. The arrival time or duration of information is considered as the timing of service delivery. An occurrence of a timing failure in the service delivery, could be as a result of the component of the system in delivery the correct response either too early or late with the specified time interval. Late timing are often referred to as performance failures (Cristian 1995, Avizenis et al 2004 & Rohr 2015).

These are often reflected in the area of performance, responsiveness and throughput (Muntz 2000). These failures can be classed as halt or erratic. Halt failures occurs when the service is halted with the or when the external state becomes constant, i.e. the system activity, is no longer perceptible to the user or cases of where the failure is silent with no service delivered at the service level. Erratic failures are temporary service disruptions that usually occur at an unpredictable time. A further important criterion is the severity of consequence of failure for the system's environment ranging from minor failures up to catastrophic failures (Rohr 2015).

4.5.4. Byzantine failure

A byzantine failure is the loss of service due to a byzantine fault in the system that require consensus. Byzantine failures are considered as the most general and most difficult class of failures among the failure modes (Lamport 2016). The term arbitrary or byzantine failure is used to refer to the type of failure in which an error may occur in a communication channel,

which leads to an arbitrary behaviour of the system in producing duplication or unreliable data. Byzantine failures in a system is hard to detect and can have a profound impact on a system. A standard communication protocols needs to have a mechanism to overcome arbitrary failures in a channel of communication (Lamport 2016 & Rohr 2015).

Byzantine failure in network system could result to where faulty nodes can behave arbitrarily and a strong effect to coordinate other faulty nodes to compromise the replicated service (Aublin et al 2013 & Lamport 2016). A failure is considered a byzantine if it causes arbitrary behaviour of a component and a loss of service in the system level requirement. This is classed as a general class of failure that can impose restrictions on a failing component (Clement et al 2009 & Driscoll et al 2003).

4.6. Dependability Attributes

A system may be seen as not dependable when it does not adequately describe the dependability attributes (Avizienis et al, 2004 & Laprie, 2004). Based on preliminary findings, for an IoT application to be classed as dependable it must have the following attributes: reliability and availability (Ojie & Pereira 2017). The related concept of safety, maintainability and integrity can be introduced, if they are important to the application. Dependability is defined by the following dependability attributes.

Table 4.3. Dependability attributes and Definitions

Dependability attributes	**Definitions**
Availability	Readiness for correct service
Reliability	Continuity of correct service
Safety	Absence of catastrophic consequences on the users and the environment
Integrity	Absence of improper system alteration
Maintenance	Ability for a process to undergo modifications and repairs

The above dependability attributes as shown in table 4.2, are discussed below in the context of IoT application.

4.6.1. Availability

As stated, in the introductory part of this chapter, this research study is mainly focus on achieving the two primary attributes of dependability in IoT applications. In the classical definition, a system is highly available if the fraction of its downtime, is low (Avizienis et al 2004 & Laprie 2004). In WSNs and in the IoT context, availability is the readiness of correct delivery of service. This include a long sensing duration for the sensors in the application, the strength of the communication protocols in send the sense data to the destination gateway (Taherkordi et al 2006). Taherkordi et al (2006) stresses that availability in IoT is the amount of time taken by the system in the delivery of correct services. Availability is directly related to the concept of reliability, in regard to reliance on the system performance (Fairbairn 2014).

Roman et al (2012), stresses the importance of resilience in the IoT architecture to assure a certain level of availability in providing the tailored specific needs in terms of performance (Roman et al 2012). The availability of a system is the probability that the system is functioning properly at a given time. Availability can also be calculated over an interval, where it denotes the fraction of the system is operation. Traditionally, availability is assumed as a characteristic of the network structure. In other words, a network is said to be available as long as the source nodes can provide relevant information to the destination gateway. Generally, a failure in a sensor node or communication link may interrupt or compromise data transmission (Avizienis et al 2004 & Costa et al 2014).

When addressing availability in IoT, one of the major concerns is on the component's failures. In IoT, a failure is a condition, where a sensor node is not operating as expected, which may be reflected in the way sensors produce and relay data packets (Costa et al 2014). Sensor node failures can be classified into two distinct groups: hardware failures and coverage failures. A hardware failure manifests when sensors run out of energy, when sensors are damaged, when they are disconnected or when a faulty condition arises due to problems in the manufacturing process (Roman et al 2014; Sauter 2014 & Costa et al 2012). Thus, a sensor with a hardware failure is assumed as a faulty node in the application. On the other hand, coverage failures may diminish the monitoring quality of the applications, with less quality from the monitored environment (Silva et al 2012 & Costa et al 2014).

A node failure may inactivate a node for relaying functions or from sensing functions and compromise the quality of retrieving data. In Niyato et al (2007) & McGarry & Knight (2011), node failure is classified as hard or soft. A hard failure is the result of significant problems in some modules e.g. the communication and energy. While a soft failure does not inactivate a sensor node in the application, but the transmitted or sensed information may not be correct or precise (Sauter, 2014 & Costa et al 2014). Availability in IoT is strongly is strongly related to communication issues. The level of availability indicates how well a deployed network is retrieving and processing data, in regard to the monitoring requirements of the considered application (Costa et al, 2016). The failure of a wireless network will directly impact packet transmission (Silva et al, 2012 & Costa et al 2016).

However, some failures in an IoT application may result from the deployment mechanism and application requirement. Understanding the causes of failures in the components of an IoT application is crucial when addressing availability in IoT (Aviziensis et al 2004 & Costa et al 2014). The ability to assess the impact of the components used in design of an IoT application will create the readiness for proper services to be produce by the system which inturn will create a degree of reliability. Fault tree analysis method can be used in the evaluation of the components of an IoT application in ensuring reliability and continuity of the services provided by the application (Chen et al 2017 & Silva et al 2013).

4.6.2. Reliability

The reliability of computer systems has been a long-lasting challenge with extensive studies, the research study of Aboeifotoh & Colbourn (1989) focused on the reliability of wired networks with unreliable links under the assumption that the nodes were perfect. The concept of reliability has been defined in different contexts as discovered during the literature review of Mahmood et al (2015) & Katiyar et al (2011). According to Laprie (1995) reliability, is the ability of a system to operate continuously without interruption. This clearly state the continuous operation of the system to produce services that can be justifiably trusted.

Avizienis et al (2004), defined reliability as the probability that the system functions properly and continuously for a specific period of time. A system is perfectly reliable, if it continues to provide the needed services. This can be attributed to the unlikely event that the constituent components are themselves perfectly reliable and the system's suffers from no error in the constituent component (Avizienis et al 2004).

According to kempf et al (2011), despite the continuous improvements in the technological component, the issue of ensuring reliability in IoT systems and its related services remains a major challenge. IoT has a tendency of widespread exploitation. Kempf et al (2011) stated that reliability requirements in the system could be satisfied in an IoT architecture, but could be constrained to limited processing capabilities, scarce energy resources and unreliable communication channels. Mostly in harsh environments, the network and radio signal of IoT is often affected by interferences which may result in significant loss of packet (kempf et al 2011).

Gubbi et al (2013), stated that reliable and timely delivery of sensor data plays a crucial role to the success of IoT application. The success of IoT depends on the delivery of high-priority events to the sinks, without any loss on the path from the original sources to the destination (Maalel, 2014). The IoT network, requires reliable and robust data transport system to function properly in spite of noisy, faulty and non-deterministic underlying physical world realities (Stankovic 2008 & Maalel 2014). A high level of reliability is required for real-world applications. According to Damaso et al (2014), the most straightforward strategy for achieving system reliability is the assessment of the independent components and its respective structural function through system modelling such as the use of fault tree and simulations methods (Yunus et al 2011 & Damaso et al 2014).

Furthermore, Angelopoulos (2016), stated that communication protocols also have an important role to play in ensuring reliability in IoT due to the of constrained of large packet and processing power. The reliability of the transmission of data is an important consideration during the design of an IoT application. The evaluation of the energy consumption of the sensor node in an IoT application is another important step in ensuring the reliability (Angelopoulos 2016 & Damaso et al 2017). The reliability of an IoT application is normally in the operations of the components, if one of them fails, the whole system fails may fail which could result in the service disruption (Anzanello 2008; Kempf et al 2011 & Song et al 2016).

The heterogeneous nature of IoT components demands strong testing capabilities to ensure service performance meets the user requirements (Esquiagola et al 2017). In a practical telecommunication or computer network, each component of the network is subject to failure. There have been a few approaches in research studies for addressing networks with unreliable links and nodes to in evaluating network reliability. In order to ensure that the IoT system retains its usefulness over a long period of time, it is imperative to test the

functionality of the components as the architecture of IoT, utilises different components for various tasks (Kempf et al 2011). According to Silva et al (2013), fault tree analysis (FTA) and reliability block diagram models (RBD) are usually used to provide reliability and availability estimates for both early and later stages of the IoT and WSN network components. This ensure the system models are more refined and have more detailed specifications (Ahmed et al 2016 & Silva et al 2013).

According to Angelo (2016), experiments and simulation is often used to know the status of a component in real-time operation whereby the component's reliability can be defined. The node reliability in IoT, is directly affected by the battery, which is often the power source of sensor node, which implies that a low battery level, means low reliability. Over time, the battery is consumed and can reach a level that is unable to meet the energy requirement of the node, which might result to a high probability of failure in the application (Damaso et al, 2014).

The notion of reliability is to ensure that computer system functions properly for a given length of time (Avizienis et al 2004 & Laprie 1995). In view of these above perceptions, definitions and various connotation of reliability, reliability can be represented as a system that constitutes the following characteristics shown in the figure (4.3). These characteristics when coined together will provide concise understanding of the definition of reliability in IoT application (Yunus et al 2011 & Gupta et al 2011).

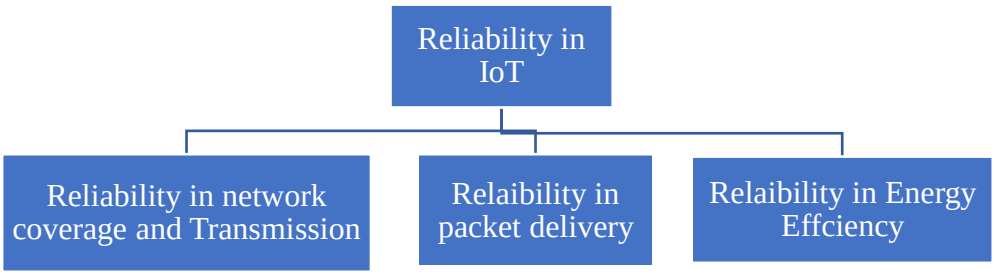


Figure 4.6. Reliability characteristics in IoT Application

For IoT application to be reliable, the system must possess the above characteristics. The subsections below contain the description of the identified characteristics of reliability in IoT application. These reliability characteristics of IoT are the major area of focus in this research thesis.

4.6.2.1. Reliable of network coverage and transmission

The reliability of the network coverage and initialisation of the sensor nodes has an adverse effect on the reliability of the application. Sensor networks under deployment perform various sensing and actuating tasks. A fundamental aspect of the successful operation of an IoT application is to have the ability to sense and communicate events to the gateway or the sink node. The sink node provides command and control functionalities to the entire network (Gupta et al 2017& Saifullah et al 2017). However, sensor nodes are subject to probabilistic events of random failure. Some of the factors contributing to the occurrence of such events include the use of low-cost sensing and communication modules, operation in harsh environments, and reliance on limited energy sources (Saifullah et al 2017).

In IoT, the sensors cooperatively sense, collect and process specific information in the monitoring area, laying the basis for real-time acquisition, processing and transmission of information. It is of great significance to study the reliability of data transmission, a key determinant of the results of monitoring events. The reliability of data transmission is an integral part of network reliability (Mahmood et al 2012 & Gupta et al 2015).

According to Gupta et al (2015), when the rate of failure of the nodes is constant, then the ability of the network to perform the assigned task of collecting information or detecting events will decline exponentially with time. This is because as the nodes in a network die, the ability of the network to acquire information about the environment, in which it is deployed, drops significantly since the number of sensing points reduce and the probability of missing out on the detection of events increases. Thus, the reliability of the sensing coverage will reduce exponentially as more nodes keep on failing on a regular basis (Yunus et al 2011 & Gupta et al 2015).

Consequently, to assure reliable and timely event detection in IoT, reliable event transport to the sink node within a certain delay bound must be effectively handled by an efficient transport protocol mechanism. Several transport protocols have been developed for sensor networks in recent years (Saifullah et al 2017). These protocols are mainly designed for congestion control and reliable data delivery from the sink to the sensor nodes and from the sensor nodes to the sink. However, none of these protocols address the application-specific real-time delay bounds of the reliable event transport in IoT. Clearly, there is an urgent need for a new real-time and reliable data transport solution with efficient congestion detection and control mechanisms for IoT (Gupta et al, 2015).

In IoT, reliability could be affected due to sensors-to-sink transport and the correlation among the sensor readings (Gupta et al, 2015). Hence, conventional end-to-end reliability definitions and solutions would only lead to over-utilisation of scarce sensor resources. On the other hand, the absence of reliable transport mechanism altogether can seriously impair event detection. In the area of IoT, a reliable protocol is a protocol where data is delivered to the intended recipients successfully. Reliability is a synonym for assurance and reliance. Thus, the sensors-to-sink transport paradigm requires a collective event to-sink reliability notion rather than the traditional end-to-end reliability notion. Appropriate action needs to be taken to assure the desired reliability level in the event-to-sink communication. To assure reliable and timely event detection, it is imperative that the event features are reliably transported to the sink node within a certain delay bound. This can be called event-to-sink delay bound, which is specific to application requirements and must be met so that the application-specific objectives of operation are achieved. Reliable event detection at the sink/gateway is based on collective information provided by source node (Vuran et al 2004).

Another important parameter in the concept of reliability is in terms of estimation of the event at sink based on the data received from the deployed sensors as enumerated by Vuran et al (2004). According to Gupta et al (2015), network latency and coherence are important parameters in case of IoT. Network latency refers to the time taken by the packet to reach sink or gateway, while coherence refers to the delivery of the packets at the sink in sequence that they were generated at the nodes. Decision making in IoT may get adversely affected if information is not received in time-bound and coherent manner. In case the constraints are not met there is a likelihood that the re-construction of the sensed event may not be correct thus leading to incorrect decision making. The event-to-sink delay bound has three main components as outlined below:

(i) Event transport delay: It is mainly defined as the time between when the event occurs and when it is reliably transported to the sink node. Therefore, it involves the following delay components:

a) Buffering delay: It is the time spent by a data packet in the routing queue of an intermediate forwarding sensor node. It depends on the current network load and transmission rate of each sensor node.

b) Channel access delay: It is the time spent by the sensor node to capture the channel for transmission of the data packet generated by the detection of the event. It depends on the channel access scheme in use, node density and the current network load.

c) Transmission delay: It is the time spent by the sensor node to transmit the data packet over the wireless channel. It can be calculated using transmission rate and the length of the data packet.

d) Propagation delay: It is the propagation latency of the data packet to reach the next hop over the wireless channel. It mainly depends on the distance and channel conditions between the sender and receiver.

(ii) Event processing delay: This is the processing delay experienced at the sink, when the desired features of event are estimated using the data packets received from the sensor field. This may include a certain decision interval during which the sink waits to receive adequate samples from the sensors.

(iii) End to end delay: The average time it takes a data packet to reach the destination. This includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue. This metric is calculated by subtracting time at which first packet was transmitted by source from time at which first data packet arrived to the destination gateway (Rohal et al 2013).

4.6.2.2. Reliability of Packet Delivery

Packet reliability refers to the process of ensuring the delivery of every data packet that contains the event information observed by the relevant sensor nodes to the sink (Mahmood et al 2012). The essence of packet reliability is ensuring the delivery of every data packet that contains the event information observed by the relevant sensor nodes get to the sink reliable (Yanus et al 2011 & Mahmood et al 2012).

The effective performance of wireless communication in large deployment of sensor is important to reliability. The primary aspect of wireless communication performance is the delivery of packet at the required time. More precisely, the performance of the packet received which include large fraction of packets that were transmitted within a time window, and the reception rate in line with the communication medium. There is very little literature that has extensively evaluated packet delivery performance on a high number of sensors within the IoT context (Karthikeyan et al 2018; Zhao & Govindan 2003).

IoT applications, are smart in nature and normally communicate data wirelessly over long distances than the traditional computer system. There are high risks in the reliability of wireless communication on large network. The loss rate on wireless links is much higher than that of wired links, and this effect accumulates quickly as the number of hops increases.

When a link between two nodes fails, the messages sent through that link will fail (Froncesa & Culler 2004; Koutsiamanis et al 2018).

According to the research study of Mahmood et al (2012) & Gupta et al (2015), the un-reliability of the medium of communication as well as the fact that is shared has an obvious impact on the ability of the nodes to communicate. As more and more nodes fail to communicate with each other, the ability of the network to pass on the acquired information to the sink reduces. This lack of ability to pass on the information collected by the nodes to the sink is reflected in a parameter called packet delivery ratio. Thus, as the packet delivery ratio of the sensors in an IoT or WSN application drops constantly then the reliability of the delivered sensed information degrades exponentially (Prasanna & Arasu 2014).

There is need for a high ratio in terms of the number of packets sent by the source node and the number of packets received at the destination gateway (Dong 2012; Kaur & Saxena 2018). Wireless communications are inherently susceptible to interception and interference. The quality of the communication link from the sensors has an impact in the performance on the packets received at the destination node. However, the radio communication range of a sink node is still limited due to standardization, regulatory constraints and physical limitations of radio wave propagation which in turn has a negative impact on the reliability of the sent packets in an IoT application (Suriyakrishnaan & Sridharan 2018). Therefore, a research is needed through simulation and experiments in finding the best communication protocols for IoT application. Simulation environment provides a flexible environment that enable the evaluation of the system-level impact of design choices (Goins et al, 2016).

4.6.2.3. Reliability in Energy Efficiency

There are real-world challenges in designing and deploying wireless sensors in practice, including wireless-link-quality dynamics and interference on communication range and reliability. However, there is very limited research on the reliability and energy efficiency in IoT applications (Ali et al 2017; Al- Kadhim & Al-Raweshidy 2019). An increase in IoT devices and larger networks to accommodate them will naturally produce a large amount of data that will need to be transmitted through the communication network. The potential for IoT connectivity to improve efficiency of energy in this intense process is far reaching. Reducing the power demands in IoT devices involves getting the data as quickly as possible from the sensor nodes, will adversely improve the reliability of the application.

For the successful implementation of IoT application, the issue of reliability in the energy consumption during the operation of IoT application is paramount. Most industrial monitoring applications follow a common operational pattern: data is acquired by some sensor of the system, processed in a controller unit and some information is then sent through a wireless channel. This process repeats over time, and this process creates a fundamental to the energy consumption of the application. Based on this assumption, the energy required to operate a wireless sensor device can be broken down into three main blocks: for data sensing or acquisition, for data handling or processing and data communication and networking (Martinez et al 2015 & Yosuf et al 2019).

According to Martinez et al (2015), the energy consumed per packet is a parameter, that depends mainly on the specific radio technology. Two main factors contribute to this, the radio power and transmission time. Radio power tends to be maximized to increase its range, although it is legally limited in each radio band such as Industrial, Scientific and Medical radio band (ISM) or other existing standards of operation. Transmission time is a parameter determined mainly by the modulation, depending on how a message is spread over time (Sendra et al 2011 & Martinez et al 2015).

Data gathering in IoT is one of the fundamental goal, which requires the sensor nodes to monitor the sensing field for as long as possible. Sensor nodes have limited energy resources and are powered by small batteries, energy efficiency is a critical issue in the design of IoT application (Jerew & Bassam 2019). Efficient routing of packets is a major concern in the IoT, the right choice of an effective routing protocol, can help to enhance the overall reliability and energy efficiency of an IoT application (Sendra et al 2011 & Martinez et al 2015).

4.7. Related Attributes: Safety, Integrity, and Maintenance

There is a lot of existing research addressing integrity and security in IoT application (Alli et al 2016; Khalaf & Mohammed 2018; Baker 2019) Security as a concept is a huge research area, in regard to the relationship to dependability these are two separate concepts. Dependability is about putting trust on the service of a system, 'reliance that the system can produce the require services within a specified time', while integrity in the context of computer systems, 'refers to methods of ensuring that data is real, accurate and safeguarded from unauthorised user'. From the logical point of view integrity is about protecting of the service, while dependability is the effective production of the service. Dependability comes first, then security can follow. However, it's an important factor to be considered, but is not classed as an attribute of dependability IoT, therefore is not part of the scope of this thesis. This same applies to safety and maintainability.

To the best of the knowledge of the author of this thesis, safety and maintainability has little literature addressing these attributes in IoT applications, as these factors are not critical, overwhelming and is not a current challenging demand to the IoT. As indicated in table 2, of this thesis, safety focuses on the catastrophic cause or harm of IoT devices to mankind, safety of humans, specifically how the IoT may directly harm humans. The core IoT devices are safe, due to their low power. In some circumstances the data provided by the IoT may be used in safety related scenarios, however in these cases, the sensors itself cannot harm humans, instead it is reliance on the IoT data that is the concern (Avizienis et al 2004 & Laprie 2004; Fairbairn 2014). Within this thesis, it is assumed that the IoT is only used to gather information and therefore does not have any physical threats to cause harm to humans, therefore safety is not a concern for this work.

The other attribute that is commonly omitted is 'maintainability' to undergo repairs, with the common view that once the IoT components are been deployed, the network is fixed for the specified period of time of operation. The component could be replaced and more nodes can be deployed inexpensively and in some circumstances the replacement of batteries within devices, as long as it is not too frequent, is acceptable and does not pose any research challenges, rather the reliability of the devices will ensure effective operation (Fairbairn 2014).

4.8. Summary

Dependability can be defined in IoT as the ability of a system to deliver a service that can be justifiably trusted. The main attributes of dependability in IoT are reliability and availability. Reliability is paramount to the success of an IoT application, it has been identified that there is still a lack of research, in the reliability of IoT, compared to the other related attributes. There is a substantial amount of research needed into reliable energy efficiency and reduction in transmission energy in IoT. There have only been minimal research on real time communication protocols for packets transfer in IoT, in regard to the timely transmission and network delay, mostly in large scale IoT scenarios. The related attributes of safety and maintainability have minimal levels to the dependability of an IoT application.

Chapter 5. Case Analysis of IoT Application

5.1. Introduction

This chapter explores in detail the key technologies, components and processes that make up an IoT application. Section 5.2 shows the IoT cases selected for this research study, a total of seven cases, were selected based on their relevance to the aim of this research study, these cases were critically analysed. This was done to achieve a better understanding of IoT applications, the constituents, and the architectures and components that have been used for development of IoT application. Followed by a critical anlysis of the IoT aapplications in section 5.3 where the main components and categorisation of the applications was presented. A discussion of the findings was presented in section 5.4.

5.2. IoT Selected Cases

In considering the analysis phase in this thesis, the cases were selected to identify the complexity of the components that make up an IoT application. In selecting the cases in this research study, appropriateness and adequacy in the cases was considered (Yin, 1994; Ahmed et al 2016 & Gustafsson 2017). The appropriate IoT cases that demonstrates the true characteristics of an IoT application were selected through a multiple case design approach and establishing an adequacy in the number of cases that creates both an evidence and alternative explanations of the IoT use cases that will satisfy the inquiry, expectation and objective of this research study (Patton 1990; Kuzel 1999 & Gustafsson 2017).

IoT is still at its exploratory stage, it takes multiple cases to understand the architectural framework. A detailed understanding into the typical cases of IoT application was conducted to illustrate the processes involved in the development of IoT application based on the application areas explored in chapter 2 of this research study. Table 5.1 below contains the cases selected for this analysis.

Table 5.1. IoT Selected cases

Case Number	Purpose of Design	Authors
Case – 1	Glucose monitoring system	Gia et al (2017)
Case – 2	Diabetes management	Al-Taee et al (2015)
Case- 3	Remote monitoring and self-management of diseases	Pradeebha et al (2018)
Case- 4	Detection of physiological symptoms for elderly patients	Ullah et al (2017)
Case-5	A personalized healthcare monitoring system for diabetic patients	Alfian et al (2018)
Case-6	An IoT driven application for road traffic monitoring	Masek et al (2016)
Case-7	IoT Sensors for Monitoring Potatoes Cultivation	Sherine et al (2016)

The cases in table 5.1, are selected due to the information that can be derived from them. Strategies for the selecting of cases could be either random selection or information-oriented selection (Ahmed et al 2016). In random selection, cases are randomly selected from a large sample mainly for establishing credibility or avoiding subjective bias. In information-oriented selection, cases are selected to demonstrate a characteristic or attribute, a full description of the information-oriented selection approach is available in Ahmed et al (2016).

In this analysis, the information-oriented selection approach was used in the selection of the cases. With the information-oriented selection approach, the cases are selected for their significance, as they reveal certain findings and can be exemplars or typical cases from which generalisations can be drawn through the logical deduction (Widdowson 2011). This cases represents the successful use of an IoT application in real life scenario. Therefore they can been seen as viable cases that can be used in the analysis of an IoT application. In analysing these cases the following step were adopted as shown in figure 5.1 below.

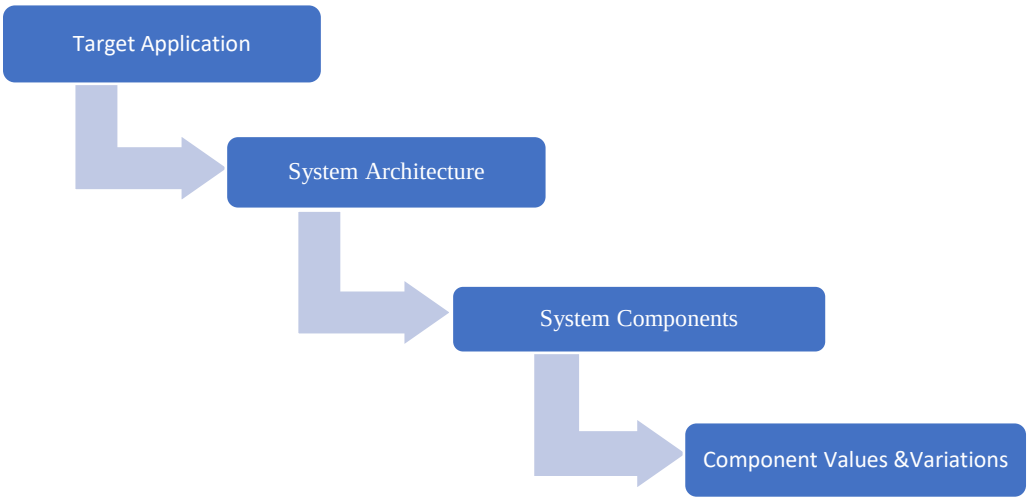


Figure 5.1: Steps in Analysing the IoT Cases

As shown in figure 5.1 above, the following steps was used in the analysis of the IoT cases involves four processes, the target application, the system architecture, the system components and the components values used in the design of the system. A brief summary of these concepts are as follows: The target application states the specific relevance of the system to the area of study, the system architecture provides a brief description of the main building blocks composing the system considering both hardware and software elements, the system components the main components that make up the system and lastly the component values are the number of devices used in the development of the application

5.2.1. Case 1: Target Application: An IoT based continuous glucose monitoring system

Gia et al 2017, implemented IoT-based system architecture which connects a sensor node to a back-end server. Through the system, doctors and caregivers can easily monitor their patient anytime, anywhere through a browser or a smart-phone application. The sensor nodes of the system are able to obtain several parameters (i.e. glucose and body temperature) and transmit the data wirelessly to the gateway (Gia et al 2017). In addition, the sensor node is integrated with the power management unit and the energy harvesting unit for extending the operating duration of the sensor device. With the assistance of the customized nRF receiver, a patient's smart-phone becomes a gateway for receiving data from sensor nodes as shown in figure 5.2 below.

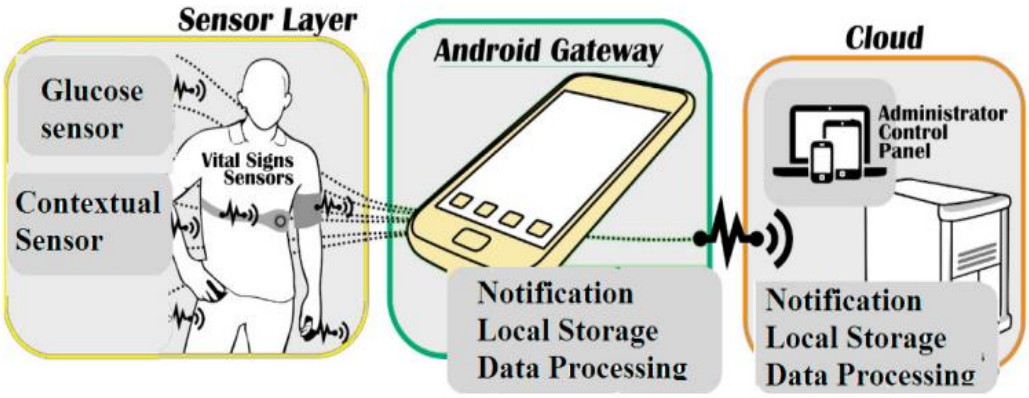


Figure 5.2: Continuous Glucose Monitoring using IoT (Source: Gia et al 2017)

A. System Architecture:

The system architecture of Gia et al (2017), shows the design of a system for the continuous glucose monitoring (CGM) system utilising the IoT based approach using the data from a sensor device to a back-end system for presenting real-time glucose, body temperature as a graphical contextual data in human-readable forms to end-users such as patients and doctors. The CGMS architecture shown in figure 5.2 is based on an IoT architecture.

B. System Components

The system includes three main components such as a portable sensor device, a gateway and a back-end system.

B1. Sensor device structure

As shown in figure 5.2, the sensor device structure consists of primary component blocks such as sensors, a microcontroller, a wireless communication block, energy harvesting and management components. The micro-controller performs primary tasks of the device such as data acquisition and transmission and receives the glucose data from an implantable glucose sensor through a wireless inductive link receiver while it collects environmental and body temperature through the data link (Gai et al 2017). The nRF wireless communication block is responsible for transmitting data from the micro-controller to the gateway equipped with an nRF transceiver. The block includes a RF transceiver and an embedded antenna. The energy harvesting unit and the power management unit in the sensor node were included in the sensor node components of the sensor node (Gai et al 2017).

B2. Gateway and back-end structure

As indicated in the research study of Gai et al (2017), the mobile gateway was used to collect data from wireless sensor devices and transmits the data to cloud servers. The gateway performs its tasks by using an nRF transceiver and a wireless IP-based transceiver (Wifi, GPRS or 3G). The nRF transceiver, which is a plug-able component, is compatible with all types of smart devices such as Android, Iphone and smart tablet. The gateway consist of data processing unit, local database, and a user interface. The backend system comprises of cloud and a user accessible terminal which the caregiver can assess the real-time data in cloud remotely through a web browser or a mobile application.

C. Components values and variations

The total number of components contained in the IoT application of Gia et al (2017), are seven sensor nodes which strategically placed in the body of the patients and one mobile gateway. The communication protocols used in this research are Wi-Fi and Bluetooth. The component values are represented in the table below:

Table 5.2. Components values in Case 1

Components	**Values**
No of Sensors	7
No of Gateways	1
Communications protocols	Wi-Fi; Bluetooth

5.2.2. Case 2: Target Application: Mobile Health platform for diabetes management based on the internet of things

Al-Taee et al (2015), presents an IoT-based platform to support self-management of diabetes through the use of mobile healthcare approach that allows for multiple care dimensions of diabetes by means of remote collection and monitoring of patient data and provision of personalized and customized feedback on a smart phone platform as figure 5.3 below.

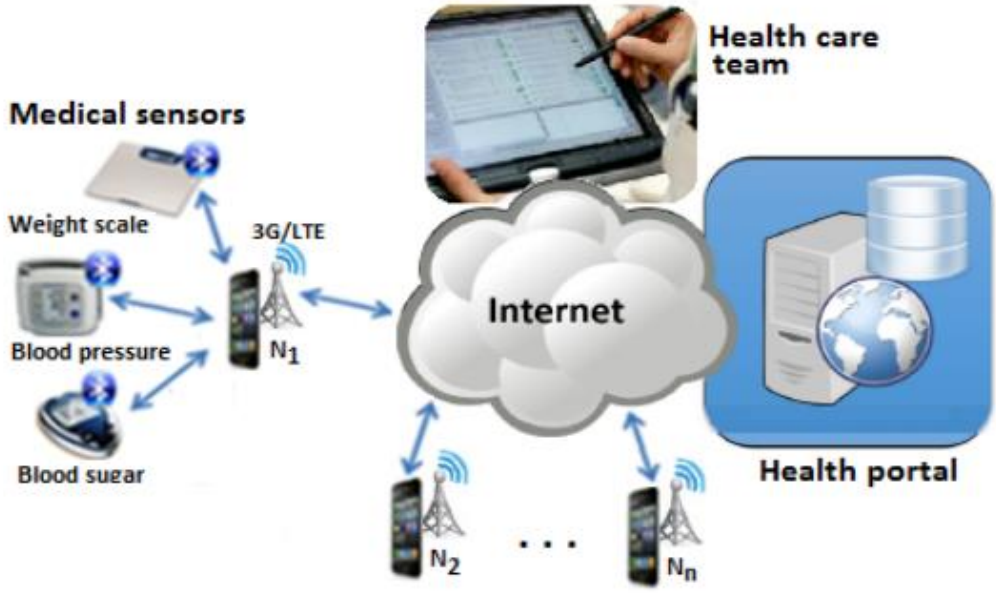


Figure 5.3: M-Health Platform for Diabetes Management Based on the Internet of Things (Source Al-Taee et al 2015)

A. System Architecture

As shown in figure 5.3 above, the system architecture comprises of three layers; physical objects, network, and a remote web-based layer, called health portal. The physical layer of the platform incorporates wireless nodes; each of which encompasses a set of medical sensors linked wirelessly to a mobile device. The sections below shows the description of the various components of the architectural platform of the research study of Al-Taee et al (2015).

B. System Components

B1. Sensory Layer

The sensors are placed in the physical layers of the system. Several medical sensors (blood glucose monitor, blood pressure, pulse rate monitor, and weight scale). All these devices communicate through Bluetooth connectivity. The network layer is represented by the long-range connectivity between the physical layer and the destination gateway which is based on existing GSM network (3G/LTE) (Al-Taee et al 2015).

B2. Gateway and backend structure

The smartphone gateway is connected to each of the medical sensors, plays a key interface role between this layer and the health portal applications. It also acts as an access terminal for patient's interactivity with the platform (Al-Taee et al 2015). The health portal layer represents the application layer of the platform that is built on the internet. It interfaces the various objects of the physical layer to other objects. It is also responsible for remote data collection and storage, data processing and monitoring, and making decisions based on constraints specified by individual patient treatment plans. In addition, it handles all user requests and generates appropriate responses (Al-Taee et al 2015).

C. Components values and variation

The analysis of the IoT case of Al-Taee et al 2015, show that the application contains are three sensor devices and a smart gateway. The communication protocols that was used in this case are Bluetooth and 3G LTE. The component values are represented in the table below:

Table 5.3. Components values in Case 2

Components	**Values**
No of Sensors	3
No of Gateways	1
Communications protocols	Bluetooth; 3G LTE

5.2.3. Case 3 Target Application: Wearable sensors nodes to read human physiological symptoms for elderly patients

In traditional health care system, patients might need to stay in hospital, but WBAN unburdens the patients to continue with their normal daily life routine outside the hospital environment. Through the use of WBAN technology, diagnosis of diseases can be made remotely at very early stages. The use of the health monitoring systems of Ullah et al (2017), will enable healthcare practitioners to administer medical treatment to elderly patients uninterruptedly, which will ultimately enhance the quality of life and health of aged patients who might find it difficult to come to the hospital environment (Ullah et al 2017).

A. System architecture

The system architecture of Ullah et al (2017), is made up of wearable wireless sensors nodes into the human body. These sensors work independently, sensing various human physiological data and is communicated wirelessly to outside world through the external server for medical analysis. The body physical parameters are being monitored using these sensors these collected body parameters; either as low-level post processed or raw samples are wirelessly transmitted to sink for further analysis and processing. The body conditions are constantly monitored by these sensor nodes and sensed data is checked for optimum level. If any parameter(s) are out of the normal (threshold) range, these sensors have the capability to send an alert signal as shown in figure 5.4 below.

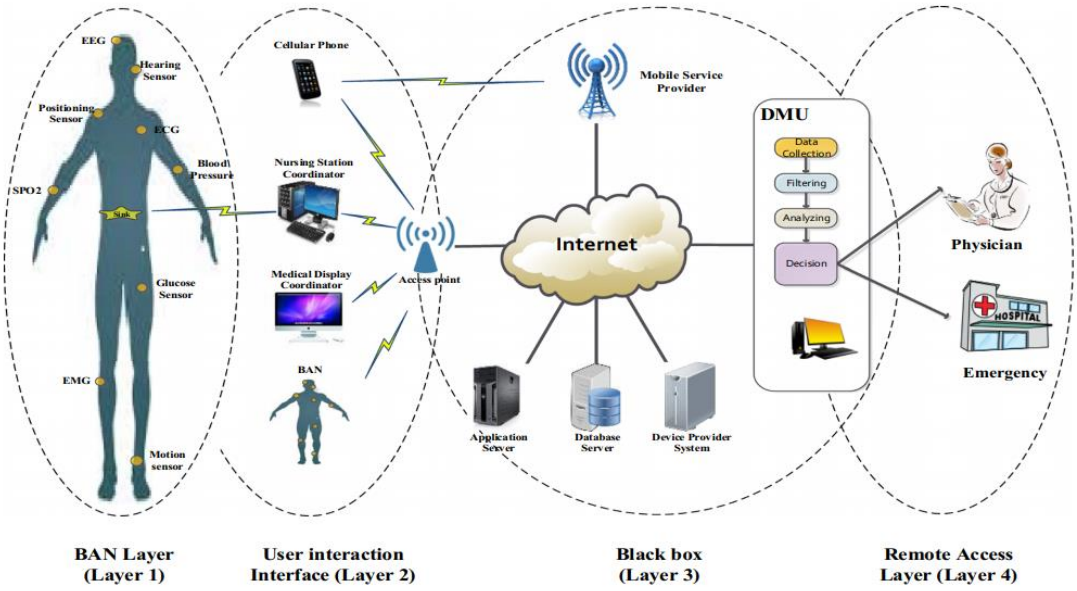


Figure 5.4: An architectural framework for reading human physiological symptoms for elderly patients (Source Ullah et al 2017)

As shown in figure 5.4, the state of the art techniques and WBANs standard technology is precisely defined in the architecture of (Ullah et al 2017) with its main components which are shown in the various layers of the application.

B. System components

B1. Sensors

The first layer (Layer 1) named BAN layer integrates multiple wireless sensor nodes operating in a limited geographic area, thus forming a Wireless Personal Area Network (WPAN). Based on its design, sensor nodes are positioned on/in the human body in the form of wearable sensors sewed in fabrics, small spots (on-body sensor), or implanted in the human body (in body sensor). These sensors continuously sense human body for the desired parameters and forward it to an external server for further analysis through the use of Bluetooth communication protocols. The sensor nodes have the capability of local processing before transmission, for processing the data collected by sensor nodes and relays it to central coordinator called sink. The data sensed by the sink nodes are then transmitted to the gateway devices, as shown in the above architecture in figure 5.4 through the use of the wireless communication protocol.

B2. Gateway and Backend Structure

The case of Ullah et al (2017) uses different gateway devices, such as Bluetooth-based smartphones, digital monitors and display units to communicate the data from the sensor nodes. These devices receive and forward data to the external monitoring unit through Wi-Fi. The Decision Measuring Unit (DMU) is connected to the back end medical server placed at the hospital through the internet. It automatically performs all major computing functions. The main function of DMU is to collect data, filter and analyse it for decision making. The processed data by the DMU is transmitted to the remote medical server. This server is placed at the hospital, where physician concerned make appropriate decisions on the information received.

C. Components values and variations

The results of the analysis of case of Ullah et al (2017) indicates the values of the components contained in the application. The total number of sensor nodes placed in the human body through the WBAN connection was 12 sensor nodes and 3 gateways connected through Bluetooth for the short range connectivity and Wi-Fi for the long range connectivity. These values are represented in the table 5.4 below.

Table 5.4. Components values in Case 3

Components	Values
No of Sensors	12
No of Gateways	4
Communications protocols	Bluetooth; Zigbee; Wi-Fi

5.2.4. Case 4: Target Application: IoT based approach for remote monitoring and self-management of diseases

Pradeebha et al (2018), presents an IoT-based platform for monitoring the patient's healthcare remotely. For patients diagnosed with certain ailments, there is a need to continuously monitor their health conditions. The patient's physiological signals are acquired by the sensor devices to check temperature, pulse rate and blood pressure by attaching it to the patient's body.

A. System architecture

The architectural platform of (Pradeebha et al., 2018), consist of a set of medical sensors linked wirelessly through Bluetooth to an android application, which transfers the data to a web-centric Disease Management Hub (DMH). As shown in figure 5.5 below, the sensed values are then transmitted to a PC for storing, analysing, and monitoring the patient health status in real time and notify relevant doctor if the patient is at risk and to access the data at any time. In this system an android application is used as an interactive device that fetches these parameters from the sensor devices and displays it on the smart phone application and transferring the collected data through android app to the database maintained at the hospital site (Pradeebha et al., 2018).

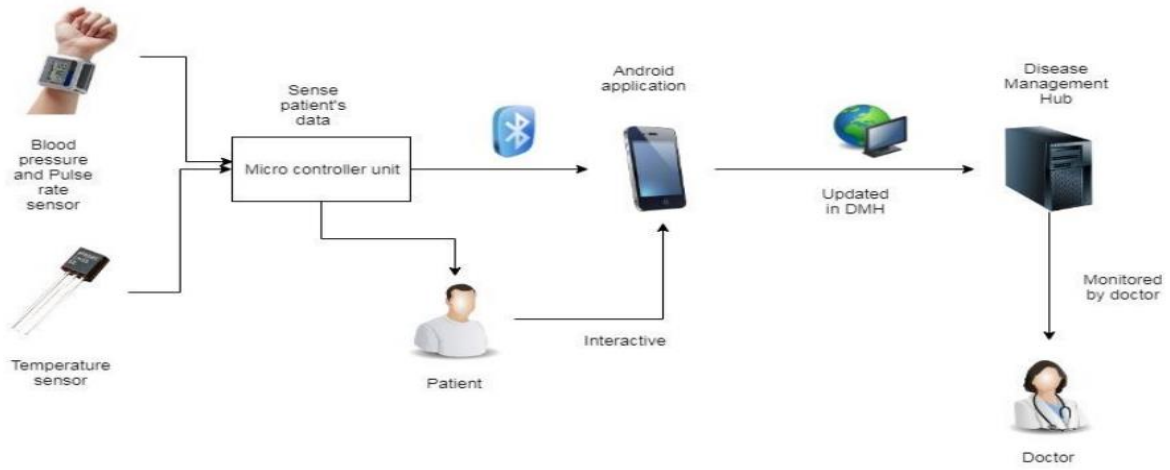


Figure 5.5: Remote monitoring and self-management of diseases (Sources: Pradeebha et al., 2018)

The medical server keeps electronic medical records of registered users and provides various services to the users, medical personnel, and informal caregivers. The DMH authenticates users, accepts patient's health monitoring session uploads, formats and inserts this session data into corresponding medical records, analyse the data patterns, recognizes serious health anomalies in order to contact emergency care givers, and forwards new instructions to the users, such as physician prescribed exercises. The doctor can access the data from his/her office via the Internet and examine it to ensure whether the patient is within expected health metrics (pulse rate, blood pressure, temperature) and ensure that the patient is responding to a given treatment. As shown in (Fig 5.5), the system consists of the following components:

B. System Components

B1. Sensor

Sensor Kit: The major component of the system is the sensor kit. The equipment mainly consists of different sensors, each for measuring body temperature, pulse rate and blood pressure connected to a Microcontroller unit. The microcontroller unit along with sensors can be connected to the patient's smartphone through Bluetooth.

B2. Gateway and Backend Structure

Android Application: Patients can login into the application using the login id and password used while registering with the hospital. Once registered the patients can use the Android app for transmitting their health status to the Disease Management Hub (DMH). The measured sensor values are displayed to the patient and then forwarded to the DMH.

DMH is maintained by the hospital. It consists of the database which includes the data, such as list of doctors available, a list of patients, patient details etc. Every patient is assigned a particular doctor. Once the doctor login into the DMH, then he or she can view a graphical report of the patient's vital signs such as temperature, blood pressure and pulse rate. This helps in analysing a patient's health condition even more efficiently by looking at the pattern of changes over a period of time. The doctor sets particular thresholds for each of the parameters. If a particular value crosses the threshold, then alert message is sent to the patient and the caregiver (Pradeebha et al., 2018).

C. Components values and variations

The total number of components used in the case of Pradeebha et al (2018), shows that IoT comes in various sizes and variation but achieve the same purpose and goal of acquiring the physical data and transmitting it to the digital world. The application of Pradeebha et al (2018) contains three sensor nodes and one smart gateway device for communicating the received data the end user as represented in the table below.

Table 5.6. Components values in Case 4

Components	**Values**
No of Sensors	3
No of Gateways	1
Communications protocols	Bluetooth; Wi-Fi

5.2.5. Case 5: A Personalised Healthcare Monitoring System for Diabetic Patients

In the study of Afian et al (2018), a personalised healthcare monitoring system for diabetic patients was developed to manage their health condition. The proposed system records various health parameters of the patients and sends the information across to for further analysis to avoid critical health conditions. The main idea behind the system is to collect patients' vital signs using sensors and then transfer the data over a wireless network to a remote service platform with the use of machine-learning methods, the patient (user) can review their ongoing health patterns and predict future changes in their health status.

A. System Architecture

The system architecture of Alfan et al (2018) as shown in figure 5.6, consists of the integration of BLE-based sensors, smartphone, real-time data processing and machine learning-based methods to predict diabetes and BG levels. The proposed model is expected

to help the patients (user) monitor their vital data from Bluetooth Low Energy (BLE) -based sensor using their smartphone. Additionally, the proposed model helps patients to discover the risk of diabetes at an early stage as well as help patients to obtain future predictions of their BG levels (Alfan et al 2018).

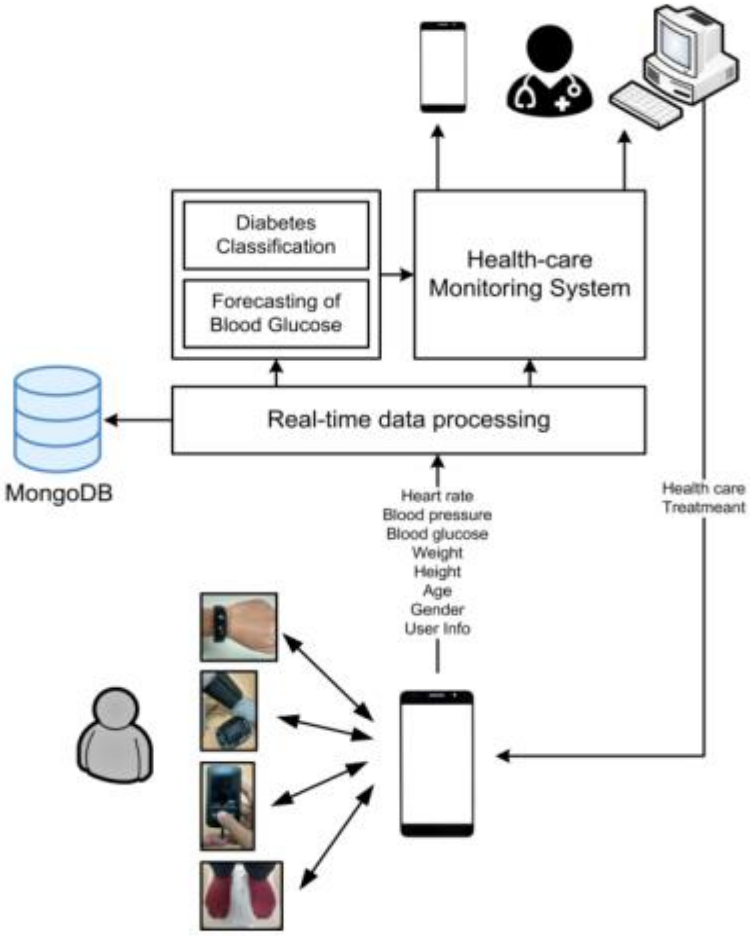


Figure 5.6: The architecture of the personalized healthcare monitoring system for diabetic patients (Source: Alfian et al 2018)

As shown in figure 5.6 above, shows the architecture of the personalized healthcare monitoring system for diabetic patients. As indicated above, the system is made up various components to ensure functionality of the system. Below are the description of system components that make up the architecture of Alfan et al (2018).

B. System components

B1. Sensors

As shown in Figure 5.6 above, the BLE-based sensors collect the patient's physiological data from the patient and then transfer the data through Bluetooth to the smartphone. The BLE-based sensor device used in the research study of Afian et al (2018) are a smart band, a blood pressure monitor, weight scales, and a glucometer sensors. These sensors are used to collect the patients' heart rate, blood pressure, weight, and BG level. Bluetooth communication protocols was used as the transfer medium between the sensor node and the smartphone (Alfian et al 2018).

B2. Gateway and Backend Structure

As shown in the architecture of Afian et al (2018), a prototype android application was developed as a central device to receive the patient's health parameter from the sensors as well their personal data which include the patient's gender, height, age, and other personal information. The smart gateway is an integral part of the architecture and it main purpose is to establish and maintain communication of data between sensors and remote server. The sensor data are transmitted through a wireless communication protocol (Wifi) to a remote server for real-time data processing. The real-time data processor receives the sensor data from the smart device and stores it to the database (NoSQL MongoDB), the sensor data is then analysed based on machine-learning algorithms to predict future changes in health status given the current data of the patients. Further description of this process is available in the research studies of Afian et al (2018).

C. Components values and variations

As shown in the case of Afian et al (2018), the total number of sensory devices reading various parameters from the human body is 4 in values which transmits the reading through Bluetooth communication protocols to the smart mobile gateway application running on android iOS as shown in table 5.7 below.

Table 5.7. Components values in Case 5

Components	Values
No of Sensors	4
No of Gateways	1
Communications protocols	Bluetooth

5.2.6. Case 6: Target Application: An IoT-Driven Application for Road Traffic Monitoring

Masek et al (2016), proposes an IoT application with embeded sensor devices targeted to manage real time road traffic conditions with complex road intersection.There has been tremendous growth in the number of vehicles using existing road network infrastructure in urban areas which comes with a consequence of related management problems, which range from traffic congestion control to driving safely and the environmental impact. The critical consequences of road congestion is related to delaying the emergency services (i.e, police, fire, and rescue operations, or medical services), strongly depend on the efficiency and travel time of the emergency vehicles (Masek et al 2016).

Therefore, new implementations and mechanisms are being proposed by the research community to improve the traffic management systems using IoT application components. Masek et al, (2016) designed a system using sensor nodes for data sensing and gathering which communicates and measures traffic parameters (e.g, traffic volume, vehicle speed, road segment occupancy, etc.) to a traffic management entity through the deployed wireless communications networks as shown in (Fig 5.7) below.

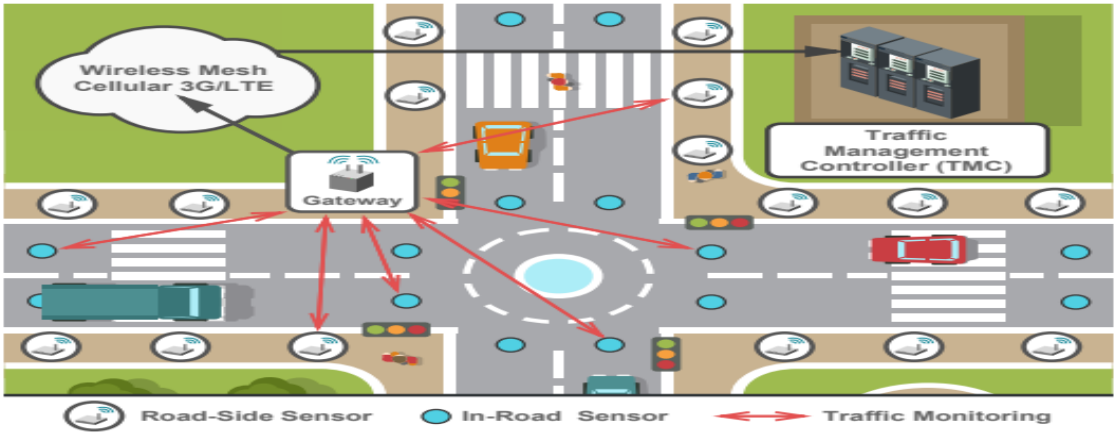


Figure 5.7: Architecture of An IoT-Driven Application for Road Traffic Monitoring (Source: Masek et al 2016)

A. System Architecture:

The system architecture of Masek et al (2016), shows the design of the traffic management system. The system logic comprises four complementary phases, as depicted in figure 5.7. The key building block of the represented is data sensing and gathering functionality, in which heterogeneous road monitoring sensor nodes measures the important traffic-related parameters (e.g., traffic volume; speed; and occupancy of the road segments) over certain time intervals. Further, the measured data is forwarded through the wireless communication (3G and Zigbee) to the gateway then to the traffic management controller (TMC) for processing.

B. System Components

Traffic Data Sensing and Gathering

The sensor nodes are placed on the side of the road and inside the road to enable effective traffic management. The main wireless technology utilised for data sensing and gathering on the road networks, embedded devices. The sensors are being deployed ubiquitously they are mounted on the vehicles, the roadside units, under the pavement to sense and report the unexpected events. In case of embedded in-vehicle sensors, the parameters related to the car operations are monitored and measured. In case of embedded devices, the sensors are used for measuring the speed of passing vehicles, the traffic volumes, or other parameters of the environment (Masek et al, 2016). This enables the collection of data from a specific region of interest, under particular time constraints while minimising the cost and spectrum usage as well as maximising the system utilisation (Masek et al, 2016).

C. Data transfer and Gateway processing

The data is transferred through the communication protocols from the sensor's nodes to the gateway. The IoT gateway device bridges the communication gap between IoT devices, sensors, network and the management system. By systematically connecting the field, the gateway devices offer local processing and storage solutions, as well as the ability to autonomously control field devices based on data input by sensors. The gateway device enables effective communication of the data from the sensor nodes. After, the processing and optimisation the data is sent through the cellular technology (3G/LTE), to a remote traffic management controller (TMC).

D. Components values and variations

The analysis of the application of the Masek et al, (2016), reveals the variations in the number of components contained in the application. The number of sensor nodes in the application is thirty strategically placed around the sensing location with one gateway device. The communication protocols used in the design of this application is ZigBee and 3G LTE. The component values are represented in the table below.

Table 5.8. Components values in Case 6

Components	**Values**
No of Sensors	30
No of Gateways	1
Communications protocols	ZigBee & 3G/ LTE

5.2.7. Case 7: Target Application: IoT Sensors for Monitoring Potatoes Cultivation

Sherine et al 2013, proposes a system to be used for the monitoring of potato crop cultivation. Sensor nodes are distributed into the tubs carat area of the farm area to monitor the soil condition and the effective management of the growth of the crop. The field is divided into tubs each of one carat area. Each carat contains sensor nodes distributed on it, with approximate separation of six meter and a node put on every its edges shared with another carat as shown in figure 5.8 below.

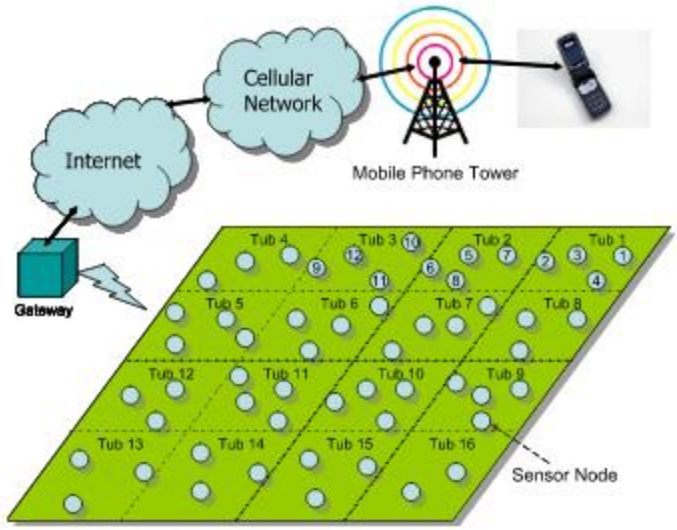


Figure 5.8: Application of sensor nodes in potatoes cultivation (Source: Sherine et al 2013)

A shown in above in figure 5.8, this application is use as a decision tool for farmers to monitor the irrigation, and other plating practices scheduling. This helps to improve potato crop and save of resources such as irrigation water and fertilizers. This modelling is efficiently through the deployment of the sensor nodes in the crop field to sense the required parameters and send it to the user on real time, where it is analysed to get the complete accurate picture of the field characteristics to take suitable decision in the improvement of the crop fields (Sherine et al 2013).

A. Sensor Node Structure

The system architecture of the proposed solution of Sherine et al (2013) starts from describing the sensor node architecture. The sensors nodes uses sensing modalities in the variations in the nodes hardware. The sensor board are attached to the data acquisition boards for reading the soil moisture. These sensors and data acquisition boards are compatible with the processor and the radio platforms that is been deployed in the farm field. The network monitors the crop during the two stages of the field before and after the emergence of plants.

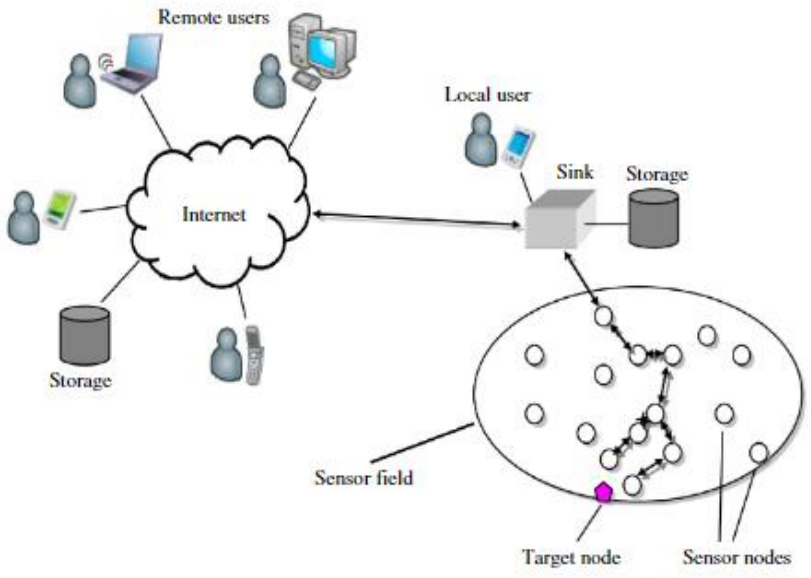


Figure 5.9. Application layer of the system (Source: Sherine et al 2013)

As shown in figure 5.9 above, the application layer involves the process of reading the sensed values with a specific rate and request from the lower layers to send it also with a specific rate or on special events. The overall architecture of the proposed solution is formed by having a number of scattered sensor nodes within the architecture which communicate wirelessly to the sink gateway and to a base station connected to the local or remote user application, which receives the network data and appropriately process it (Sherine et al,

2013). Each, cluster has member nodes and one head node responsible for receiving, aggregating, and transmitting the data of its cluster members.

B. Sink Gateway

The data is received in the sink gateway, stored and processed the information about the cultivation field to the user. As shown in figure 4.10 above the sink has a storage unit for saving data the memory for the gateway processing unit. The gateway sink can is also used to manage the agricultural field. The data is stored and analysed in order to provide the farmers and users an overall perspective on the area they monitor and support them with a number of actions. The gateway device enables effective communication of the data from the sensor nodes. After, the processing and optimisation the data is sent through the routing protocol to the base station (Sherine et al, 2013).

C. Components values and variations

In summary, the analysis of the application of Sherine et al, (2013), a high number of sensor nodes was used in the monitoring of the cultivation field. According to the prediction of Perera et al (2015), around 50 to 100 billion wireless devices will be connected to the internet by 2020. This raises a high concern, into the research of the dependability of the operation of these network devices. Although, this application reflects more of a wireless sensor network specific application, the system specification and operation is a replica of a true characteristics of an IoT application, as similar components, structures and processes are involved as shown in the above sections of this thesis. The values identified in this application are shown in the table below.

Table 5.9. Components values in Case 1

Components	**Values**
No of Sensors	56
No of Gateways	1
Communications protocols	ZigBee

5.3. Analysis of the Findings

In the process of analysing the cases in the previous section it was clearly observed that IoT applications comes in different variation and sizes. Though with similar components, but it varies in the sizes of the application. In case 1 the sensor device structure consist of the primary components blocks such as microcontroller which performs the task of data acquisition and transmission, while in case 3 the sensors relay the data directly to the central coordination which be referred to as the sink and the data are sent to the gateway. Case 2 and 4 has similar sensor operation structure. Among the cases studied, some differences were identified in the architecture. In some cases, the components could communicate directly while in other were into segments in to sensor network and application layer. On the one hand, some cases adopt a cluster-based communication scheme as its routing protocol between the sensors and the gateway. Similar to conventional gateways of the other IoT systems the gateways as described in case 1 and 3, the gateway consists of the data processing unit, the local database and the user interface as compared to the gateway of case 5. Table 5.10 below is a summary of the identified components in the assessed cases and number of values.

Table 5.10. Analysis of IoT Application

Type Device Measurement	**No of Devices**	**Communications Protocols**	**Gateway Types**	**Num**
Case 1: Continuous Glucose Monitoring using IoT (Gia et al 2017)				
Glucose	7	Wi-Fi	Android gateway(Smart phone)	1
Temperature		Bluetooth		
Case 2: Diabetes Management base on IoT (Al-Taee et al 2015)				
Glucose	3	Bluetooth	Smartphone	1
Pressure		LTE (3G)		
Wieght				
Case 3: Wearable sensors for elderly patients (Ullah et al 2017)				
EKG	12	Bluetooth	Smart Phone	4
EMG		Zigbee	Computers	

SpO2		Wi-Fi	Wireless network Hub	
BP			Smart TV	
Case 4: IoT Based approach for remote monitoring and self-management of diseases(Pradeebha et al 2018)				
Blood pressure	3	Bluetooth	Smart phone (Android)	1
Pulse rate		Wi-Fi		
Temperature				
Case 5: A Personalised Healthcare monitoring System for Diabetic Patients (Afian et al 2018)				
SpO2 /Heart rate	4	Bluetooth	Smart Phone (Android)	1
Smart Band				
Blood pressure				
Glucometer				
Case 6: An IoT-Driven Application For Road Traffic Monitoring (Source: Masek et al, 2016)				
Traffic Volume	30	3G/LTE	Mobile Gateway	1
Speed		ZigBee		
Case 7: IoT Sensors for Monitoring Potatoes Cultivation (Source: Sherine et al, 2016)				
Humidity	56	ZigBee	Mobile Gateway	1
Temperature				
Irrigation				
Soil texture				

As shown in table 5.10 above, irrespective of the application type they all had similar components and sequence of operation. However, from the analysis of the selected cases, a variation in sizes and components used in the application was identified to be different values and number indicating that IoT application comes in different sizes and scales which could be small, medium and large. In the case 3 a large amount of IoT components were identified in the architectural design as compared to the case 1.

After a detailed comparison of the selected cases, it was confirmed that IoT comes in various sizes, comprising of various components that make up the application. Therefore, IoT application can be classified as an heterogeneous network that consist of various devices connected on the same network (Qui, 2018). Complexity of components could possibly lead to failure if not properly assessed for its functionality. The dependability of the operation of these components in providing a justifiable service to the user is paramount. Base on the theoretical analysis of the above applications as shown in table 4.8, an assumption can be made on the categorisation of IoT applications, as small, medium and large scale application as shown in table 5.11.

Table 5.11. Categorisation of IoT application

Scale	**Components**		**Total Values**
	Sensor	**Gateway**	
Small	1-10	1-5	15
Medium	10-25	1-5	30
Large	25 and above	1-5	30 and above

As shown in table 5.11, an assumption can be made in the categorisation of an IoT application as a small-scale applications are categorised as 1 to 10 sensors with a variation ranging from 1 to 5 gateway not more than 15 in the total components. A medium scale application can be categorised to comprise of 10 to 25 sensors with not more than 5 gateways and a large-scale application can be classed an application that consists of values that are more than 25 in the total components.

5.4. Summary

The analysis conducted in this section shows that IoT applications consist of various component. This component includes the sensory devices, gateway, and communications protocols which represents the three main building blocks of an IoT application. Thereafter this analysis leads to a further categorisation of IoT application as it was highlighted above, that there is variation in the number of devices in the application which create a high level of complexity. This analysis creates a level of certainty that regardless of the environment in which the system is intended to be deployed or the application scenarios the main components below represents an IoT application. From the analysis conducted, the sensors

are placed in the physical layers to sense and read the environmental parameter of the application. The communications protocol, create the link between the communications in the sensors node and transmits the data to the gateway. The gateway accumulates data and facilitates the connections to the digital world.

Chapter 6. Fault Tree Analysis of IoT Application Component

6.1. Introduction

This chapter is focused on the fault tree analysis of the components of an IoT application. As stated in the previous chapter, the components that make up an IoT application includes the sensors, communication protocols and the gateway. Section 6.2 contains the overview of the fault tree elements that was used in the construction of the fault tree in this research study. This was followed by section 6.3, where a qualitative analysis was conducted through the standard and component fault tree in analysing and tracing the root cause of the failure in an IoT application, through a systematic top-down approach. Thereafter, the critical faults of the components of an IoT application are presented in section 6.4, with a discussion of the deductive analysis and consequences of the identified faults in the system.

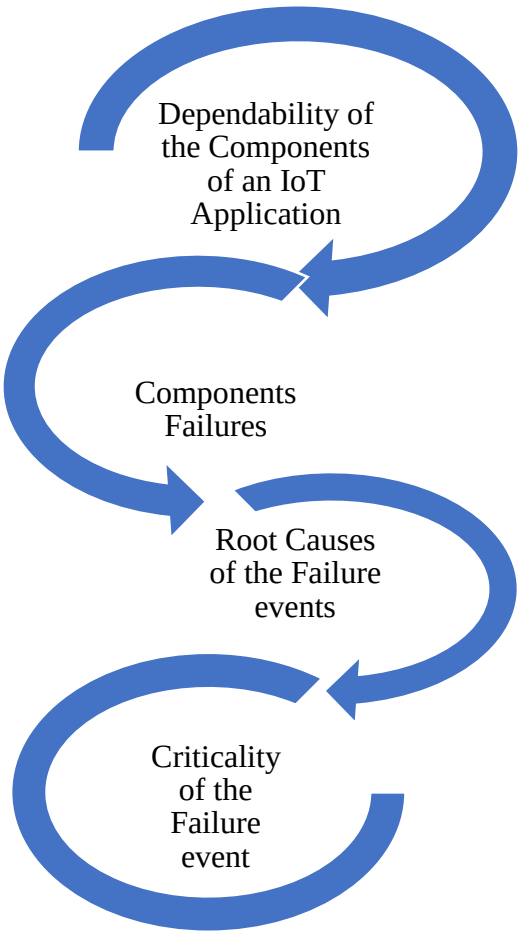


Figure 6.1: Steps in analysing the Dependability of the Components of an IoT Application

6.2. Overview of the Fault Tree Analysis

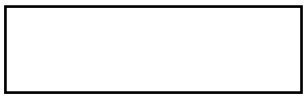
Fault tree analysis (FTA), is a deductive technique commonly used to evaluate the system's dependability. This technique describes the root causes that lead to system failure. The system is then analysed in the context of its operation to find credible ways in which the undesired failure in the event can occur. FTA is represented in a graphical model that shows the combination of events that leads to failure in the application (Silva et al 2013).

The qualitative analysis allows the identification of all the combinations of the basic events, known as cut sets, which can cause the top event to occur (Ahmed et al 2016). In analysing the components of an IoT application, the qualitative analysis shows the combinations of the failures that must occur together to cause a top-level failure. The qualitative results include: the minimal cut sets of the fault tree, qualitative component importance, and minimal cut sets potentially susceptible to common cause (common mode) failures.

6.2.1. Elements Of A Fault Tree

A fault tree uses a tree-like structure, which is composed of events and logic gates. The failure event, are represented either as normal or faulty condition, in components of the application. Logic gates are used to represent the cause-effect relationships among the events. The inputs events of the gate, are either single events or combinations of events. Which is as a result of the output of the other gates attached to the main logic gate of the top event. The elements, of the fault tree that were used in the analysis of the components of an IoT application in this research study, are represented below.

Top / Intermittent Event:



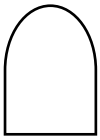
The top or intermittent event in a fault tree is a system failure or a unit failure which occur as a result of the consequences of the failures in the component these are represented as minimal cut sets. The minimal cut sets, in the fault tree, indicates the failure that can lead to a system failure. The process of building a fault tree is performed deductively, by defining

the 'TOP' event, which represents the system failure condition and by proceeding back to the possible root cause of the failure event.

OR Gate:

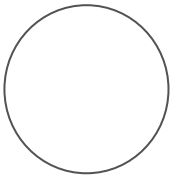
The OR-gate represents a union of the inputs attached to the Top event. Anyone or more of the input must occur to cause the event above the gate. The IoT is a critical system which comprises of three main components (the Sensor node, the Communication network and the Gateway), which can be represented in boolean as, A (failure in the sensor node), B (failure in the communication network) and C (failure of the Gateway). If one of these components fails in a typical three-input OR- gate, then the Top event (Q), fails automatically.

AND Gate:



The AND-gate represents, the intersection of the events attached to the Top event in an IoT application. The AND-gate shows that the Top event will only occur if all the input events occur. In a typical IoT application, with component A, B, and C. The output event Q occurs, with the combination of the failure of component A, B and C, in contrast to the OR-gate, where, either one of the component failures will lead to failure of the entire application. The AND-gate, specify the relationship between the failure events in components of the application which collectively represent the cause of the event.

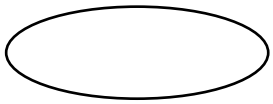
Basic Event:



A basic event in a fault tree is an initiating fault that does not require any further development or expansion and is graphically represented by a circle. Basic events are represented as leaf nodes in the fault tree and they combine to cause intermittent events. In IoT applications,

there are basic events that occur in the application that contributes to the failure of the application.

Conditioning or Priority Event:



The ellipse is used to record any critical failure conditions that apply to the logic gate. These conditions are event that occurs in the operation of the system that adversely lead to failure in the application or restrict the successful operation of the logic gates. There are three main components in an IoT application (three tier system). A critical failure in any of the component of an IoT application will affect the performance. The conditioning event in a system component are critical to the operation.

6.3. Qualitative Analysis of an IoT Application using Fault tree

The emphasis in conducting a qualitative analysis using the fault tree approach is to create an understanding of the causes of a failure in an IoT application. In the analysis of the various IoT applications conducted in chapter five of this thesis. The main components that constitute an IoT application were identified. This showed a high indication that an IoT application is a three-way system.

A failure in any of these components could adversely affect the operation of the application which impacts on the dependability in the service delivery. In analysing the dependability of the components, the standard fault tree approach was used to construct a graphical model that describes the relevant failures that can occur in the system leading to the top event, using the top-down approach. Where the failure in the IoT application, which is referred to as the top event, is as a result of a failure in the component of the application, as shown in the figure below.

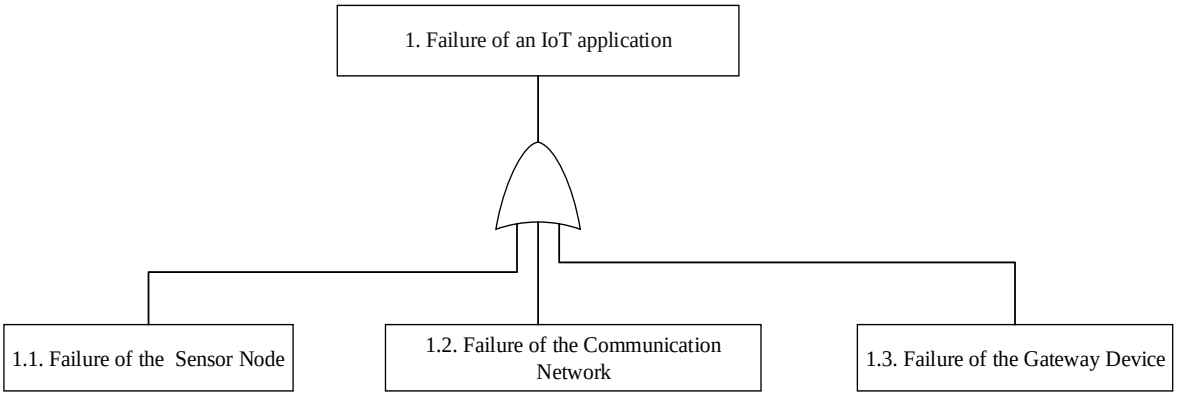


Figure 6.2: Fault tree of an IoT Application

Figure 6.1 above, the failure in an IoT application is a resultant of the failure of one of the components of the application. From the analysis of the fault tree using the OR gate, a failure of a component could automatically cause the failure of the application. This indicates a high level of functional dependency of the components. An IoT application is a dependent system, although the components are independent. In the construction of an IoT application, the three components need to be functional in delivery of its services (Bauer et al 2013 & Xing et al 2017).

Assessing and evaluating the reliability and availability of an IoT system is critical since it guarantees the success rate of IoT service delivery (Xing et al 2017). It is important to analyse the reliability associated with the components in a communication system the sensors, the communication protocols (transmission links) and the connecting gateways, since the unreliability of an underlying component will adversely undermine the function, which in turns lead to a failure in service delivery of the entire application (Xing et al 2017 & Domb 2019).

The complex interactions of the components complicate the reliability of an IoT application (Xing et al 2017). Particularly, functional dependence in the IoT system, where the failure of one function in a component triggers the failure of the component and then causes a failure in the other components (dependents components) and application service delivery (Xing et al 2014). According to Xing et al (2017), in a relay-assisted communication network of smart home systems, some sensors are functionally dependent on the relay node (transmitting data gathered from related sensors to the sink node to realise long-distance transmissions). Therefore, when the relay fails, its dependent sensors become isolated (Wang et al 2015 & Xing et al 2010).

The functional dependence of a component of an IoT application affects the systems reliability due to the fact its failure can cause a severe failure (Xing et al 2017). In analysing the sensor nodes and its constituents. Specifically, failures of the trigger (relay node) and the propagated failures of the corresponding dependent function. If a sensor node, experiences a propagation failure before the relay fails, this propagated failure can spread throughout the system and cause a crash in the entire system. On the other hand, if the relay failure occurs first, the failures of the dependent sensors can be isolated, and the rest of the system may continue to function. This indicates that, there priority events or conditional events that could result to failure in a particular component of an IoT application, that not all events in the component can cause a failure. The subsequent sections below shows the fault tree analysis of the independent components that make up an IoT application.

6.3.1. Failure of a Sensor Node

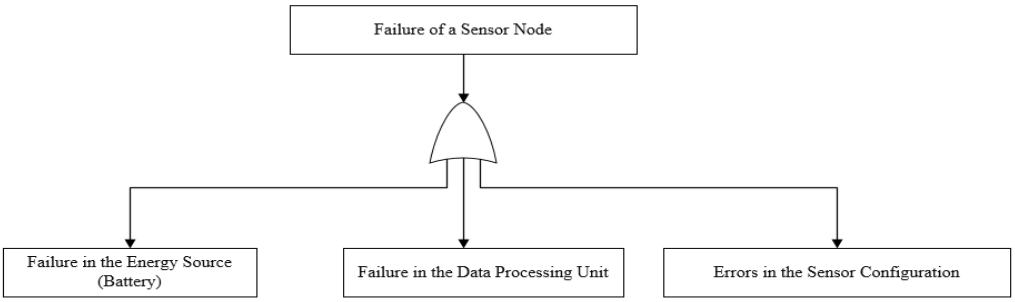


Figure 6.3: Failure of a sensor node

The three main units of a sensor node (Fig 6.3): are the energy source which is usually the battery, the data processing unit and the configuration unit (Gupta 2013; Sethi P & Sarangi 2017; & Bouguera et al 2018). The combination of these unit creates effectiveness in the operation of the sensor nodes. However, the failure of the in any of this unit will affect the operation of the sensor node (Luan 2017). The analysis of these units is shown in the segments below.

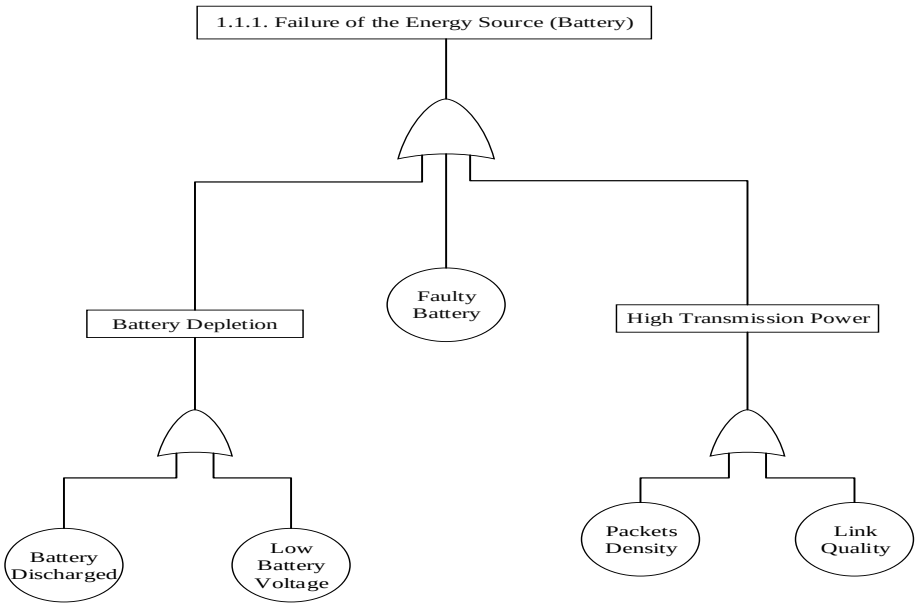


Figure 6.4: Failure of the Energy Source

The failure of the energy source is critical, when considering using wireless sensor nodes in the design of an IoT application. Sensors depend on the battery as their main source of power (Shelke et al 2013 & Kim et al 2019). As indicated in the fault tree (Fig 6.4), the failure of a sensor node's battery could be as a result of the depletion of the battery, a fault in the battery or the high usage during transmission and operation. Battery depletion are usually caused by a discharge in the battery or low battery voltage. According to Hayashi et al (2017), device failure from unexpected battery depletion is uncommon, but can be life-threatening. This is usually as a result of low voltage. High transmission usage during the operation of a sensor has an adverse effect on the battery life of a sensor node which could critically lead to failure of the application (Wu et al 2013).

According to Bouguera et al (2018)**, e**nergy efficiency is the key requirement in maximising the lifetime of a wireless sensor node. For the effectiveness of the operation of an IoT application, the sensor nodes needs to operate reliably for an extended period of time. Wireless sensor nodes are typically powered by a battery source that has finite lifetime, which limits its capability (Dutta et al 2012). Wireless Sensor nodes are mostly used in the design of IoT applications, as IoT applications are usually used in collecting physical parameters about a given phenomenon. These sensor nodes are generally deployed to operate over long time periods without human intervention during system operation (Bouguera et al 2018).

Timely transmission of data, from the sensor nodes to the gateway is essential in minimising the low level of energy contained in the battery (Khriji et al 2018; Shang & Farooq 2018).

Energy is consumed during the data transmission process, when there is delay in data transmission process, this could in turn lead high energy consumption (Dutta et al 2012 & Khriji et al 2018). Therefore, the high efficiency and reliability of sensor nodes depend on the communications protocol to a large extent to meet the constraints on service quality under limited energy conditions. Therefore, it is imperative to analyse the factors that contribute to failures in the of data processing unit of a sensor node

An important aspect in the deployment of a wireless sensor node is ensuring that there is always adequate energy available to power the system. The sensor node require energy for sensing, communicating and data processing. More energy is required for data communication and processing. The data processing unit of a sensor node is composed of the sensing unit, the transceiver which is used for communication between the transmitter and the receiver as shown below in figure 6.5.

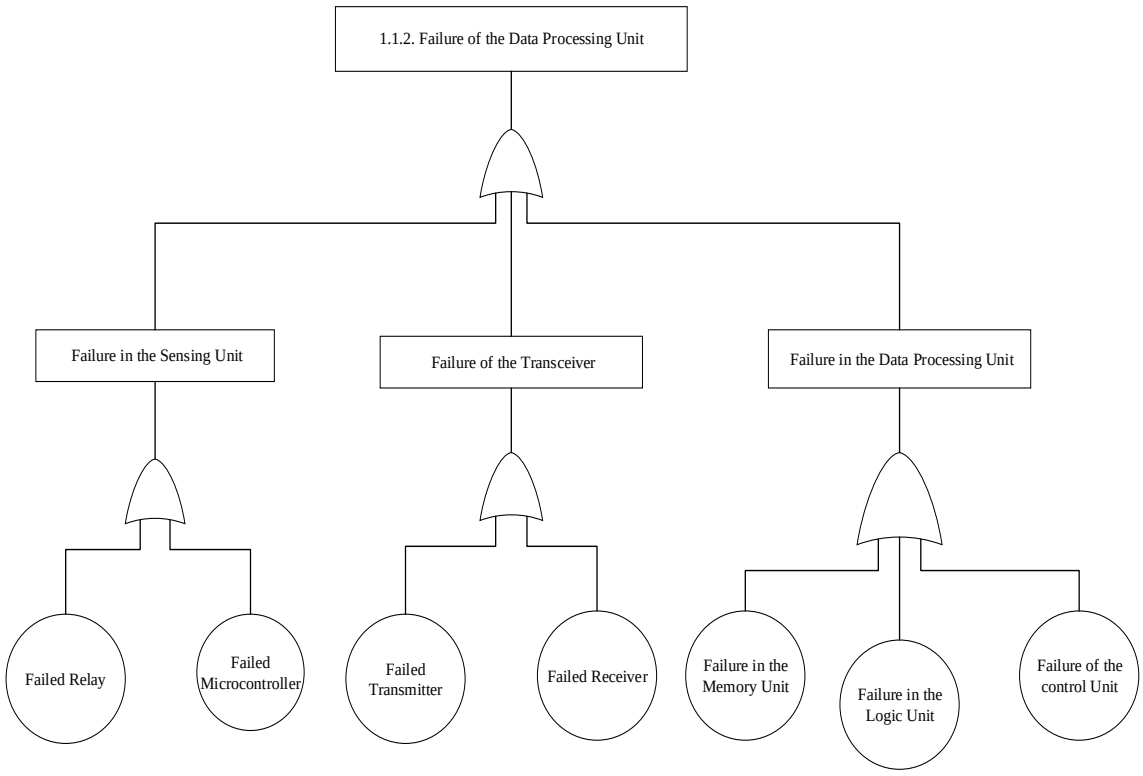


Figure 6.5: Failure of the Data Processing Unit

The data processing unit are the programmable electronic component that processes the streams of data. The data is transmitted to and from these components as multiplexed packets of information (Vieira et al 2006 & Elkhier et al 2013). The processing unit is responsible for performing tasks, processing data, and controlling the functionality of other components of the sensor node. The micro-controller performs tasks, processes data and

controls the functionality of other components in the sensor node. A wireless sensor connects with other nodes, through a transceiver, which functionality mainly depends on the transmitter and receiver. The failures of the each of these electronic components has an effect of the operation of the sensor node.

A sensor device runs on a programmed configuration, which is normally in a form of a piece of code used in extracting the data from the monitored physical environment. The configuration of the sensors devices are in classes and methods in the device instruments. The failure in the sensor configuration often tends to be as a result of the device accuracy or the programming unit error as represented in figure 6.6.

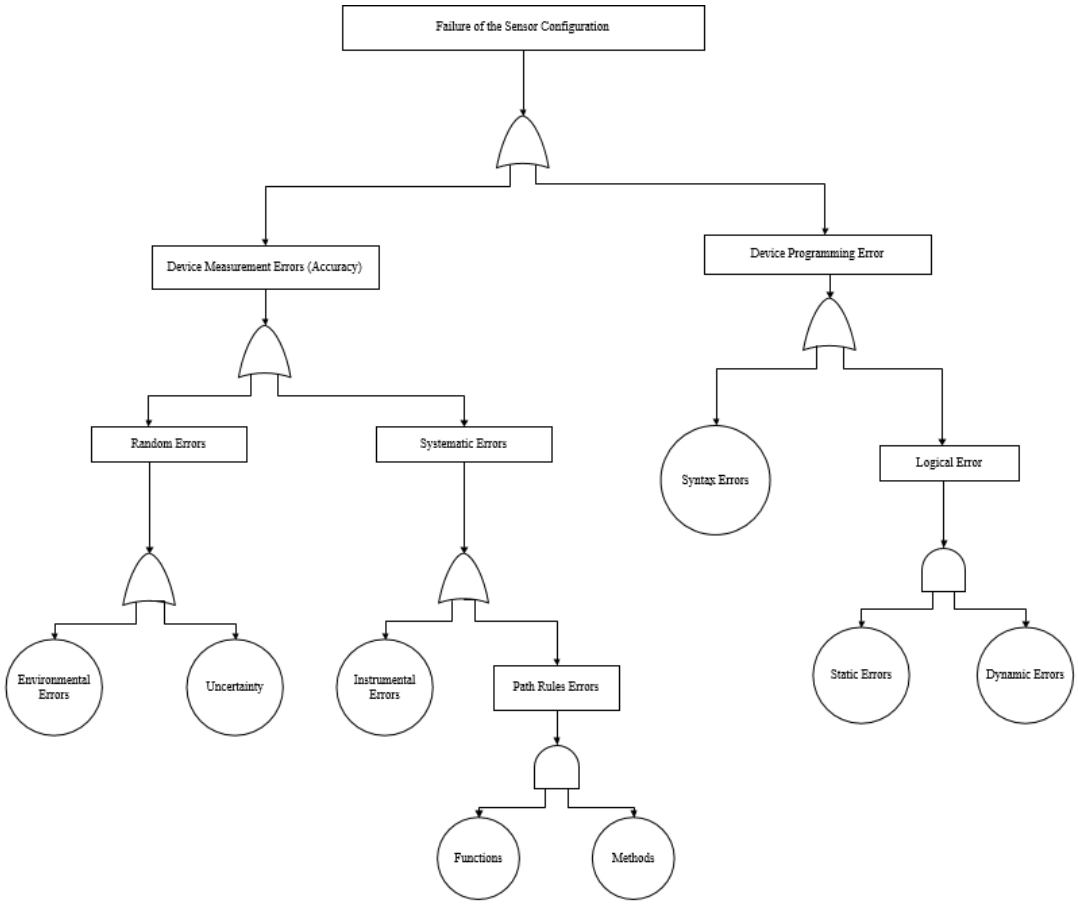


Figure 6.6: Failure in the Sensor Configuration Unit

The device measurement errors, the fault tree, is as a result of the error in the sensor configuration which is either random or systematic. Random errors can be fluctuations in the measurement of the physical parameter due to the precision limitations of environment or uncertainty of errors in the measurement device. According to Prabhakar & Cheng (2009), data readings collected from sensors are often imprecise. The uncertainty in the data can

arise from the measurement errors due to the sensing instrument. Systematic error is the inconsistence and repeatable failures often as a result of the faulty device. This could happen as result of the rules of the programme sensor node which affect the readings of the wrong parameters in the physical environment. The purpose of a sensor node is to read the true physical parameters.

As shown in the fault tree, a failure in the configuration unit of the senor node will lead to fault in the sensor device, which could adversely result to the failure of a critical IoT systems (Saarnisaari & Braysy 2006; Jesus et al 2017). A failure in the device programming unit of a sensor node configuration, could adversely lead to the errors in the readings of the sensor node. This could be as a result of the programme logic error in program's source code that results in an incorrect or unexpected reading of the sensor node during operational runtime. This type of runtime error may simply produce the wrong output or may cause a program to crash while running (Mottolla and Picco 2011).

6.3.2. Failure of the Communication Network

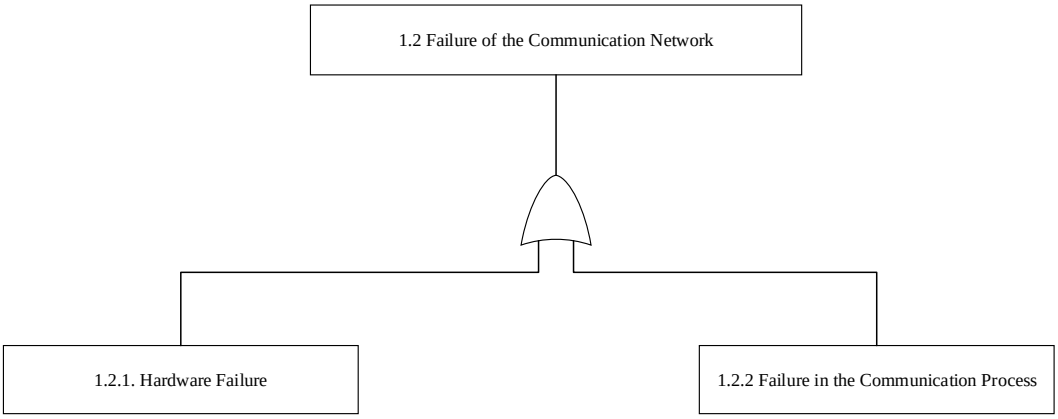


Figure 6.7: Failure of the Communication Network

The fault tree analysis in figure 6.7 above, consist of two main units: the hardware unit (networking device) and the communication process (data transmission process). The networking devices are the component hardware that enables effective communication links between the sensor nodes and the gateway device (Li et al 2011 & Bourgeois 2014). The failures in the network elements has an impact on network reliability (Gill et al 2011). The reliability in network devices is critical, mainly due to a failure in network devices could have an adverse impact on the effective communication of the data (Fan et al 2017 & Gill et al 2011). The failure in the elements of the communication links could adversely lead to

network delay in routing the packet to the destination gateway (Mukherjee & Biswas 2018). The figure 6.7 below, shows that the failures in the networking devices that could result to the failure of the communication links.

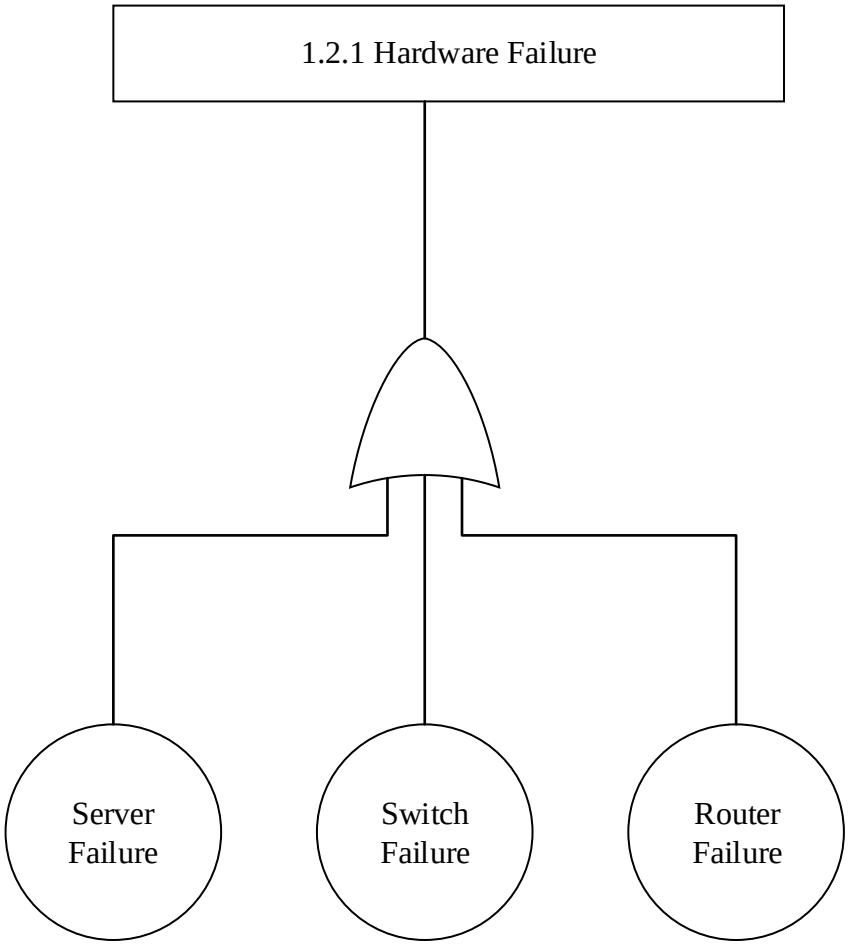


Figure 6.8: Failure hardware device (Networking devices)

A failure or fault in the network device or communication link could adversely affect the data transmission rate (Fig 6.8), but not necessary lead to complete failure in the data transmission, as, networks have alternative paths during data transmission (Marina and Das 2006; Mekki et al 2019; Salman & Jain 2019). The failure of the network devices, has an impact in the data transmission, which is classed as a basic event in the fault tree. The failure in the communication process is relevant in analysing failure of the communication network of an IoT application. The failures in the communication process of an IoT application is shown in figure 6.9 below.

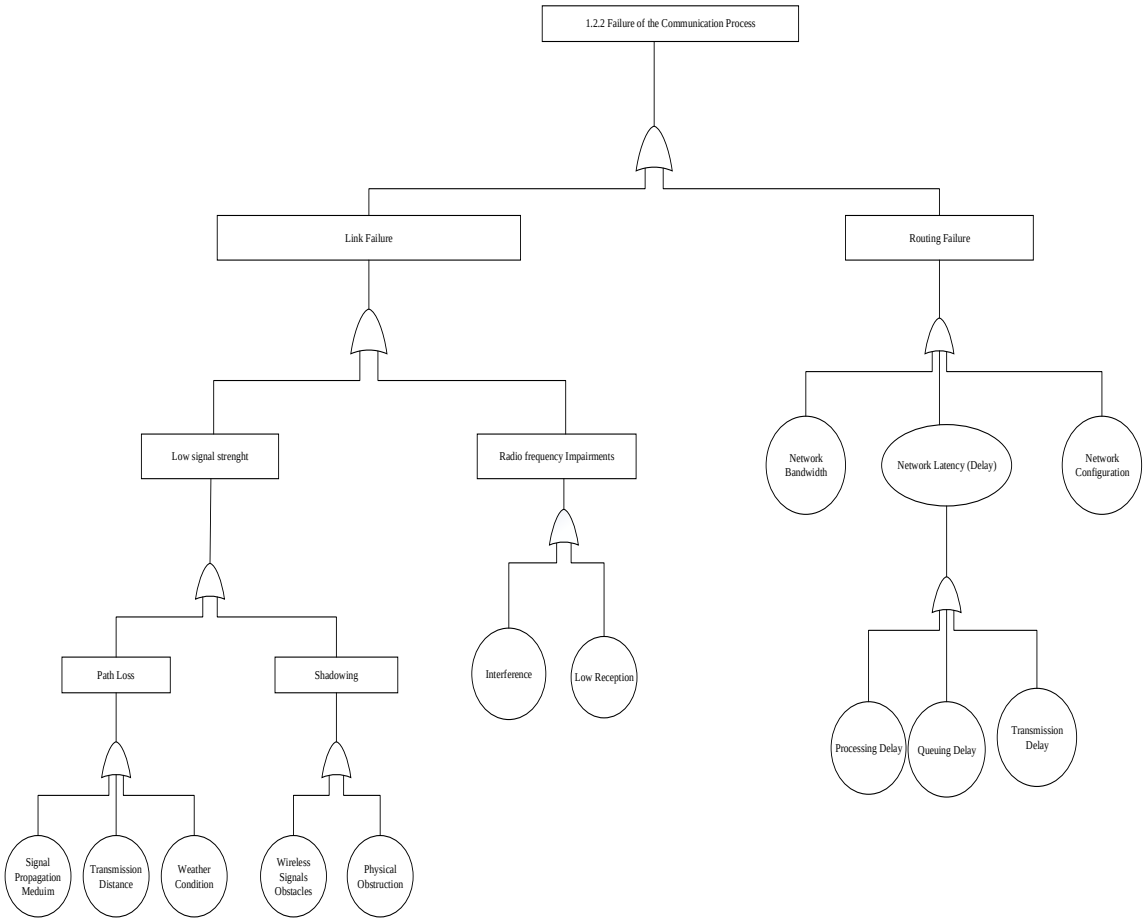


Figure 6.9: Fault Tree Analysis of Communication process of an IoT Application

The fault tree (Fig 6.9) above, indicates that failure in the routing of packets is indicated as a major conditioning event that could adversely affect the reliability of an IoT application in regards effective service delivery. Network delay is one of the most critical concerns in achieving reliability in data transmission in an IoT application (Mukherjee & Biswas 2018). The delay of a network is the amount of time it takes for the packets to travel across the network from one communication endpoint to another (Srinidhi et al 2019). The packets must reach the gateway within a given time. The delay of the packets to the destination gateway directly affect the network performance and thus increase the transmission energy as more time is spent in the routing of packets (Shang & Farooq 2018). An IoT application depends on an effective communications protocols for the transmission of data packets.

However, in cases where large amount of sensor nodes are dispensing packets, there might be variations in transmitting the data packets. Hence, a reliable communication protocol with high throughput and bandwidth is required to transmit the packet in a reliable and timely manner (Marina and Das 2006; Malathi & Jayashri 2018). The more the number of sensors contained in the application the more packets that are been transmitted. The failure in the

transmission range and channel of the protocol, has an adverse effect on the timely transmission of the packets (Malathi & Jayashri 2018). In achieving time reliable communication of the packets, it is necessary to weight the impact of the various communication protocols through simulation and experiments (Noda et al 2011; Malathi & Jayashri 2018; Salman & Jain 2019). An effective communication protocol will in turn lead to low processing time of the data packets to the gateway (Marina and Das 2006). The failure in the transmission range, could have an impact in the data transmission, this is classed as the basic events in the fault tree, as signal failures could be temporary failures due to weather condition, blockages, obstacles or minor fault in the network.

6.3.3. Failure of the Gateway device

The gateway device is an important component of an IoT application, as this has an impact in the successful routing of the data packets, mostly in applications that has a large amount of sensor nodes (Haikun et al 2018). An IoT gateway, serves as network medium in communicating with the sensor nodes, mostly in local and short distance communication (Chuan et al, 2014). The gateway device is integrated into the IoT network, for reliable transmission and intelligent processing of the acquired sensed data (He et al, 2012). The sensed data is collected in the IoT gateway for further processing and creates a connection with the wider network for remote monitoring of the sensor's environment.

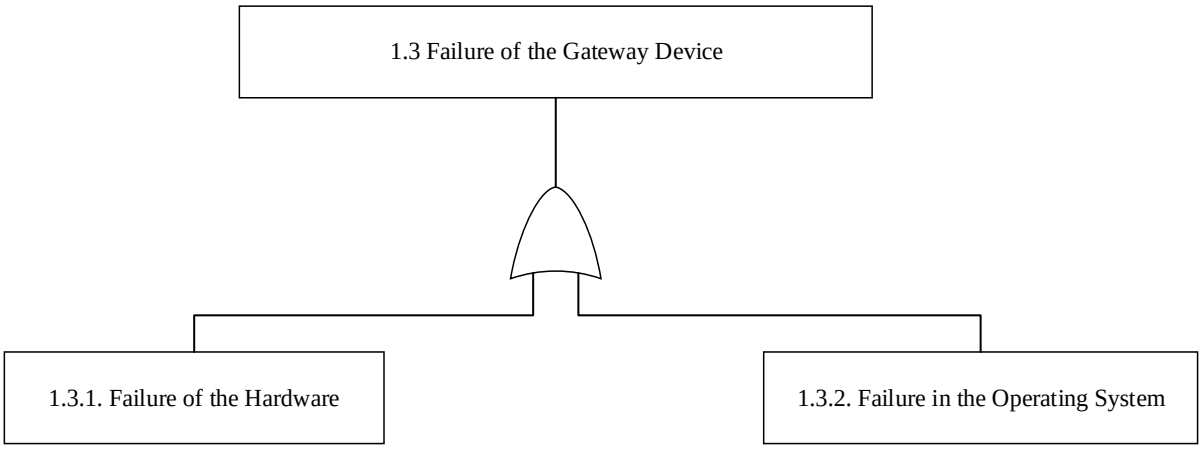


Figure 6.10: Failure of an IoT Gateway device

The fault tree in (Fig 6.10) of an IoT gateway comprises of two main units, the electronic processing unit and the operating system. The operating system of an IoT gateway is a configuration, running on the device hardware which act as the control units for the effective functionalities (Chen et al 2011 & Haikun et al 2018).

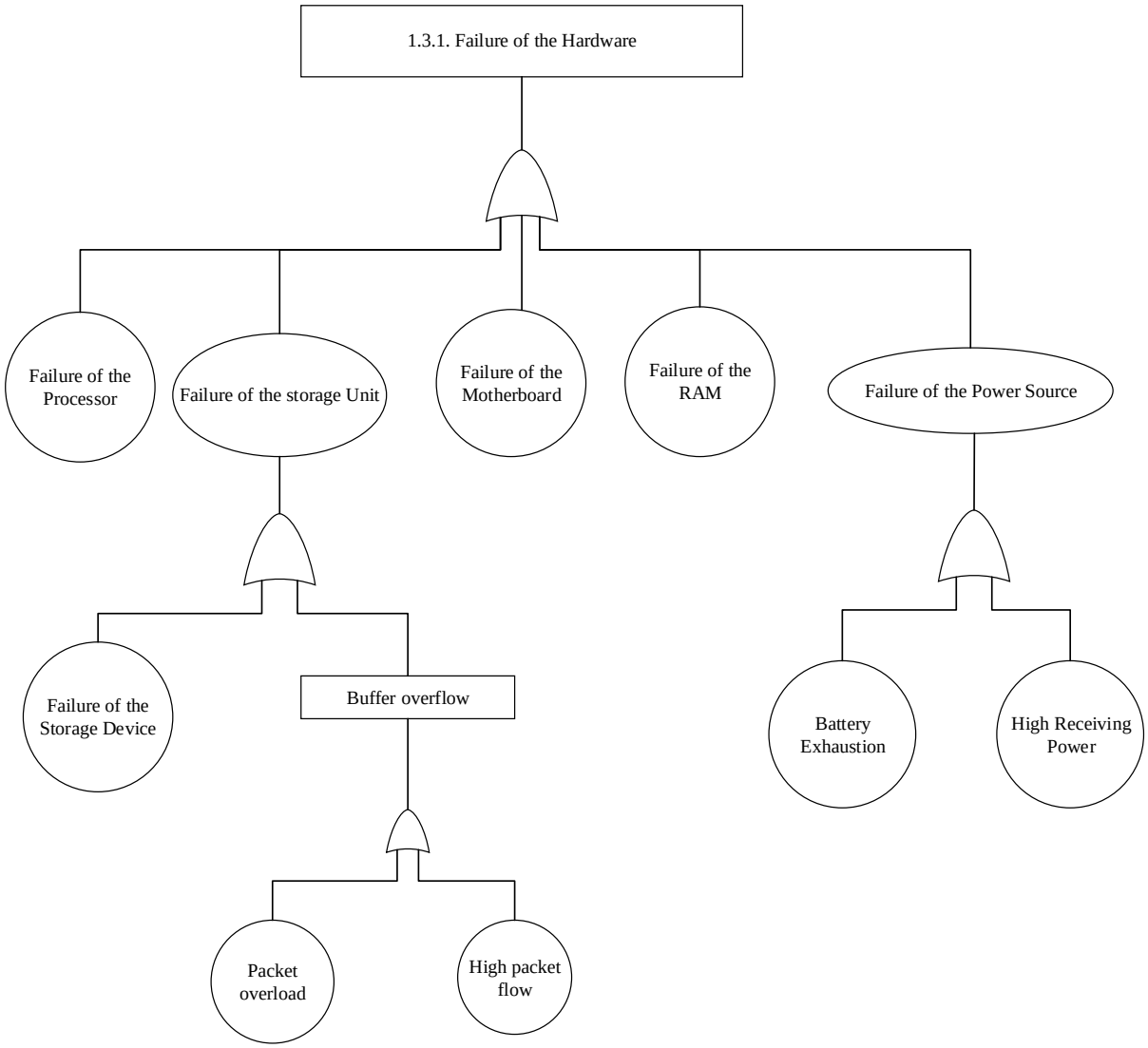


Figure 6.11: Fault tree analysis of the failure in the hardware components of an IoT Gateway

The failure in the storage unit is a conditioning event to the successful operation of a gateway device. The storage capacity of the IoT gateway has an impact on the reliability of an IoT application in regard to effective service delivery (Fig 6.11). IoT gateways needs to be able to store and process huge amounts of data from extensive sensor networks and deliver advanced edge analytics. The heterogeneity of possible scenarios and massive deployment of the enormous number of sensors in the IoT environment, a scalable gateway to accumulate the data is important (Guoqiang et al 2013 & Pezez et al 2018). The expansion of the IoT

interconnected smart devices will trigger frequent packet congestion and will influence failure in the IoT gateway (Chen et al 2019). The storage capacity in the IoT gateway need to be scalable to accommodate the vast amount of data (Aazam et al 2014 & Chang 2018).

The IoT gateway is often used for effective routing of the packets to the remote world through the use of the communications links (Karthikeya et al, 2016). Failures in the communication link with the remote world, does not lead to the failure of the gateway device (Chuan et al, 2014 & Haikun et al, 2018). The most critical part of an IoT gateway, is the data processing and storage as indicated above in the fault tree. The ability of the device to accumulate the number packet been sent from the sensor node. IoT sensor generate data constantly, and often requires the gateway to receive and process the vast amount of data (Elkheir et al, 2013; Chang 2018; Domb 2019). The failure of the data processing and storage unit has an adverse impact on the gateway device, as this failure, can lead to the entire failure of the application.

The electronic processing units comprises of the processors, the motherboard and the random acess memory. A failure of any of these device drivers could adversely affect the function of the operating system, as operating systems requires drivers to function effectively. However, not all the failures that occur in device drivers are critical to the operation of an IoT gateway device (Haikun et al, 2018). Errors in the operating system of an IoT device could be as a result of the following determinants.

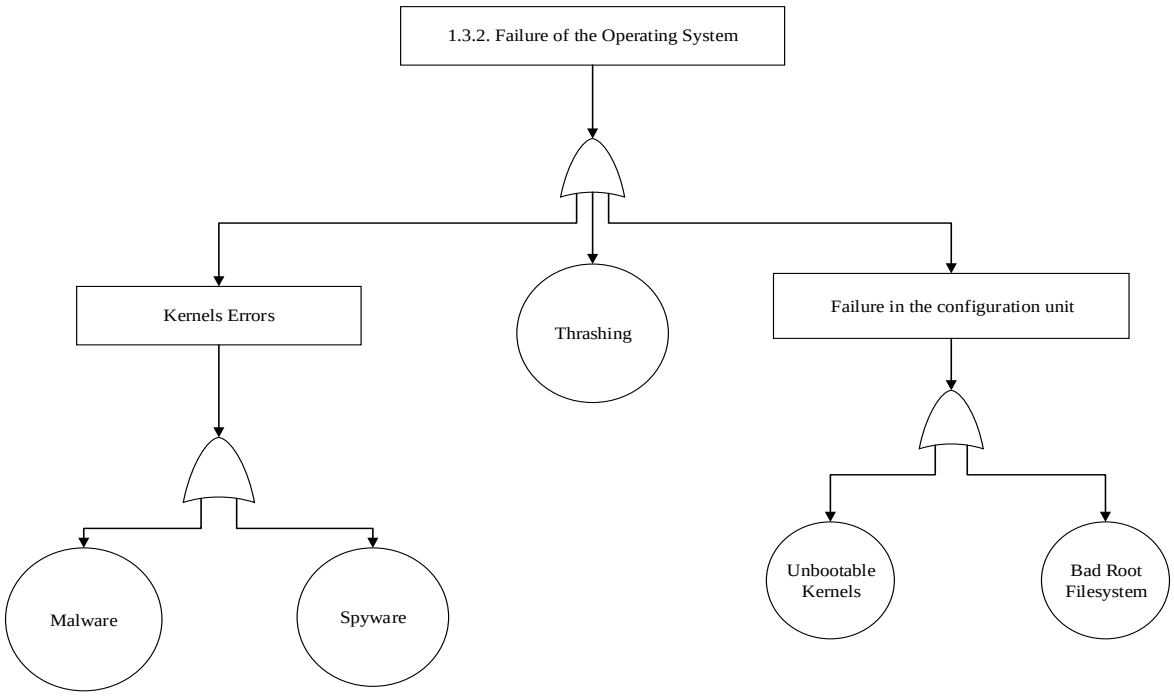


Figure 6.12: Failure in the operation system of an IoT Gateway

As shown in (Fig 6.12) the failure of the operating system of an IoT gateway could be as a result of the errors in the kernel, thrashing and errors in the configuration. Operating systems are not standalone solutions, they operate on device hardware, through the use of one or more device drivers as shown in the fault tree. A kernel error is a failure in that is critical to the operating system. The kernel is an important part of the operating system that handles the management of the memory and the device drivers. Software applications are prone to errors like bugs or exceptions, these errors, are seen as a basic that necessarily do not result to the failure of the gateway device. Thrashing, occurs when the memory resources are overused, leading to a constant state of paging and page faults, inhibiting most application-level processing. This causes the performance of the gateway device to degrade. A failure in the configuration unit of an IoT gateway could be as result of the unbootable kernel or the bad root file system.

6.4. Use of Fault Tree Analysis

The use of fault tree analysis method led to identifying the critical parameters in assessing the dependability of an IoT application. A fault tree analysis was conducted on the main components of an IoT applications which include the sensor, communication protocol and the gateway device. The three key parameters identified through the fault tree analysis are energy consumption, delay and scalability. It has been observed that for the sensor node, efficient energy consumption is critical as battery is the main source of power for a wireless sensor node. Energy is required for all the processes involved in a sensor node, from sensing to transmission. A failure of the battery will entirely lead to the failure of the sensor node. Hence a reliable transmission of data with low energy consumption is required for the successful operation of an IoT application. The failure of the timely routing of packets can adversely affect the dependability of an IoT application (Chen 2017; Bouguera et al 2018 & Yosuf et al 2019).

Network delay has been identified as another critical parameter when it comes to addressing the dependability of an IoT application. The data packets must reach the gateway within a given time. The delay of the packets to the destination gateway directly affect the network performance and thus increase the transmission energy as more time is spent in the transmitting the packets. The storage capacity of the IoT gateway has an impact on the reliability of an IoT application in regard to effective service delivery. IoT gateways needs

to be able to store and process huge amounts of data from extensive sensor network (Koutsiamanis et al 2018; Kaur & Saxena 2018).

6.5. Summary

The results of the fault tree analysis conducted on the dependability of an IoT application. It can be certain that the three components of an IoT application are relevant to successful operation of an IoT application. IoT application is a three way system. Failures of one of the components of the application will lead to the entire failure of the system irrespective of the purpose of design. However, from the analysis it was ascertained that there are criticality in the level of failure of a component of an IoT this leads to the entire failure of the system as not all failure in the components is critical. The essential dependability requirement in an IoT application was identified through this analysis. This include the energy consumption of the sensors, the delay in the communication protocols and the scalability of the gateway.

Chapter 7. Dependability Assessment Framework for IoT Application

7.1. Introduction

The objective of this chapter is to present and discuss the concept of dependability assessment framework. The intended purpose for the design of the structures contained in the dependability assessment framework, is to enable IoT system developer and analyst to assess the dependability of an IoT application, during the design phase and before the deployment of an existing IoT application. The first section in this chapter (section 7.2) is about the parameters used in the design of this framework, followed by section (7.3) which shows the general architecture and overview of the framework. Thereafter the application of the dependability assessment framework was described in section (7.4) and finally a discussion of the entire processes and usage was presented in section (7.5).

7.2. Framework Parameters

The objective of this framework is to provide practical ways to assess the dependability of an IoT components. The dependability parameters can be described as characteristics, requirement or measurable factors that can help in defining the dependability in an IoT application. An evaluation of the components of an IoT application creates a basics for the selection of the right components that will produce the expected result in order to avoid critical failures in the operation of the application. Hence it is important that the components used in the design of the IoT application are tested for their validity using the provision of the dependability assessment framework.

The components of an IoT application was critically examined to identify the important dependability elements in assessing the dependability of an IoT application through the fault tree. A typical IoT application will consist of sensors, communication protocols and a gateway device. The critical dependability requirements of an IoT application are represented in the figure 7.1 below.

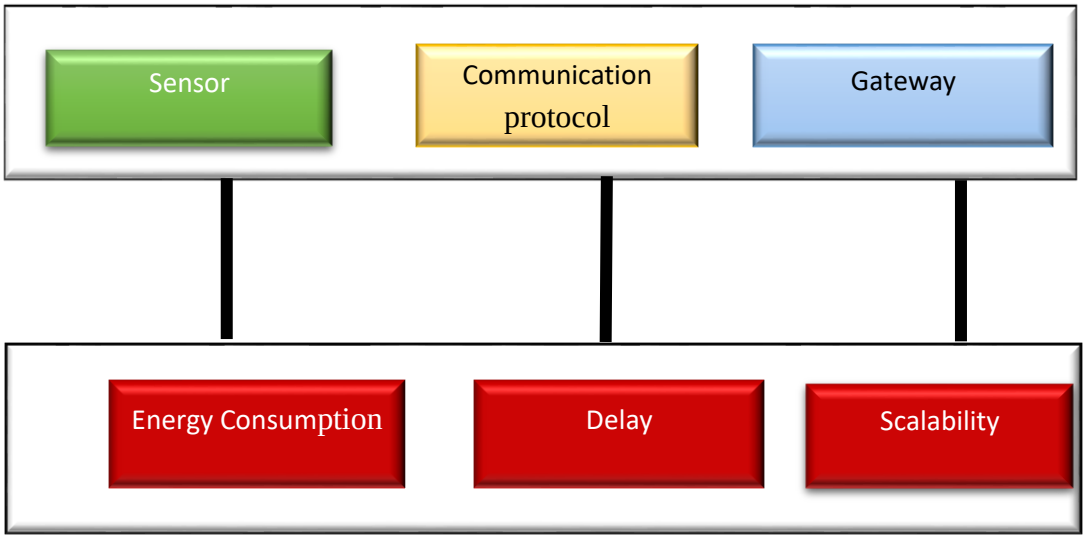


Figure 7.1: Critical Dependability Requirement of an IoT Application

The critical parameter are the logical deductions from the fault tree analysis. This include, the impact of the energy consumption on the sensor node, for communication protocol the network delay in the transmitting the packets transmission and for gateway the scalability of the gateway. The following parameters are described below:

- **Energy Consumption**

From the derivation of fault tree, the failure of the energy source is critical, when considering using wireless sensor nodes in the design of an IoT application. The critical constraint on sensor nodes, is that sensors depend on the battery as their main source of power. Sensors are deployed unattended wireless and are mostly in large numbers, so it will be difficult to change or recharge batteries in the sensor node during the system operation. Sensor consumes energy in sensing the physical environment and in acquiring data. It is essential that the processes of communication in an IoT network minimise energy during its operation to avoid critical failures in the operation (Ali et al. 2017). Effective timely transmission of sensor data from the sensor nodes is essential in minimising the low level of energy in the sensor nodes. The higher the delay in communication process, the increase in the energy consumption.

- **Network Delay**

As indicated in the fault tree analysis, network delay is one of the most critical concerns in achieving reliability in data transmission in an IoT application. This is the amount of time

taken for a packet to be transmitted from the sensor node to the destination gateway. The communication protocols used in the transmission of the packets from the sensor node to the gateway has a huge impact on the delay in the system (Kim et al. 2017). Communication protocols must perform well as the network grows larger or as the workload increases to provide real-time communication in the during the operation of an IoT application (Buratti et al. 2009 & Kim et al. 2017).

- **Scalability**

This is an essential parameter to the successful performance of any IoT network which involves large number of sensor nodes is the scalability of the gateway device. The IoT gateway is a network medium that ensures effective communication of data to the digital world through its communication with the sensor nodes both in short and long-distance communication (Benyamina et al. 2009 & Volger et al. 2016). However, as the number of sensor nodes increases in the application, then the need for a scalable gateway becomes a priority in an IoT network to be able to receive the large amount of packets from the sensor nodes (Volger et al. 2016).

7.2.1. Size of the IoT Application

In assessing the dependability of an IoT application, the size of the components in the application is an important step to ensure the successful operation. IoT applications are complex systems made up of a combination of components. Assessing these components will create an understanding of the nature of the network devices in the operation of the system. An identification of the particular components used in the design of the application is paramount to the dependability of the application. Components has variations, identifying the communication channel of these variations, will thus create an effective service delivery of the application.

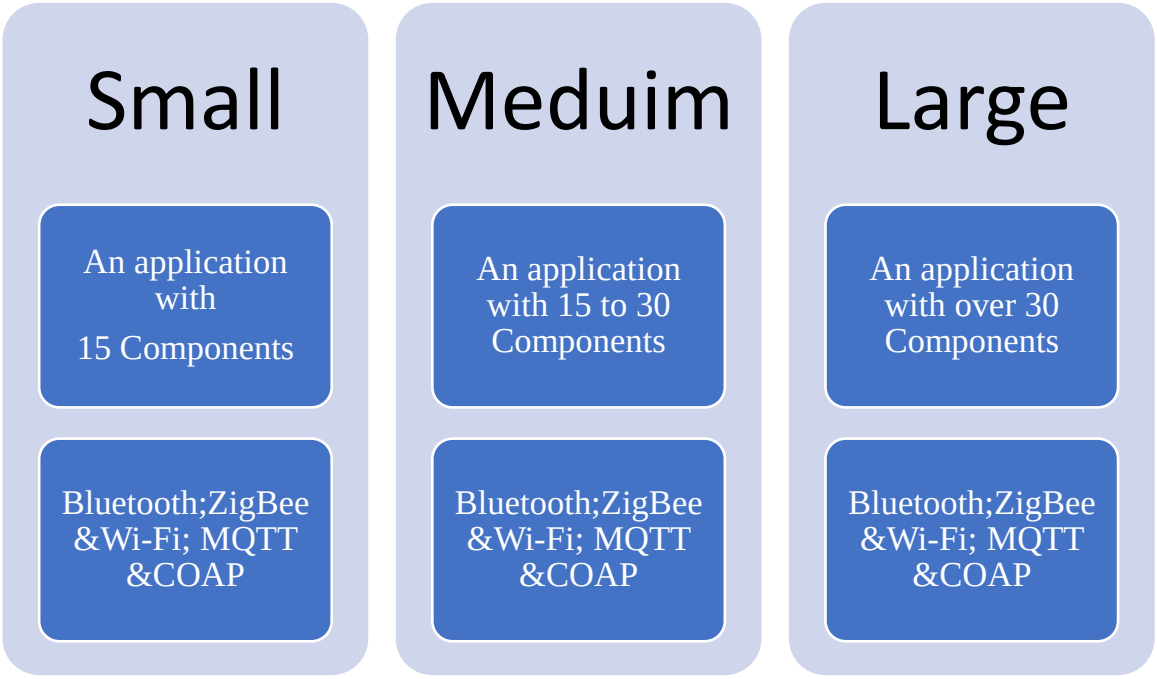


Figure 7.2: Variation of Components in an IoT Application

IoT applications can be in varied according to the number of components (7.2) in the application. In the analysis conducted in chapter 4 of this thesis, an explicit categorisation was conducted based on the findings in the components used in the construction of the applications. The result of this findings leads to the classification and scaling of IoT application types as small, medium and large. In assessing the dependability of an IoT application it is relevant to put the size of the application into consideration. The size of the application is a factor that affects the dependability of an application, dependability varies with the number and type of components used in the construction of the application.

7.3. Architecture of Dependability Assessment Framework

The dependability assessment framework was developed with different phases in the assessment of the dependability of an IoT application. The framework is design in three layers. The application size layer, the component layer and the critical dependability parameters of focus. These whole structures are put in place to ensure that the dependability of IoT application is ensure during the design and deployment stage. The effective use of this framework will create an effective assessment of any IoT application irrespective of the application domain. The overall architecture of the framework and its main structure is shown below in figure 7.3.

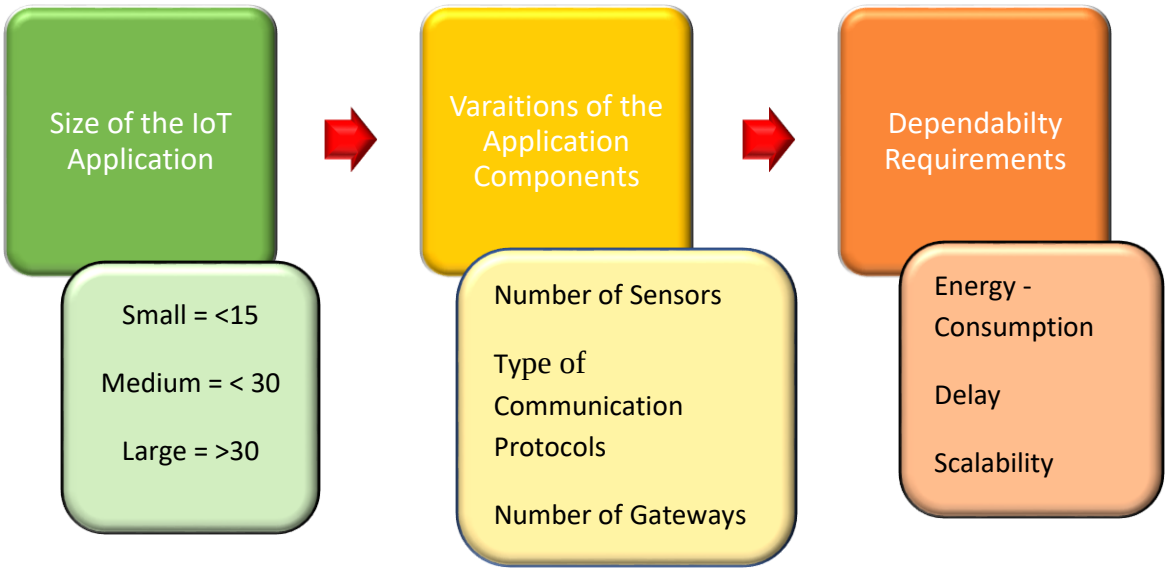


Figure 7.3: Dependability Assessment Framework

The fist layer in the dependability assessment framework is the size of the application, this consist of the scales of measurement. The values of a small scale IoT application not more than 15 components value in the application, while the medium scale is ranging between 15 to 30 component value and the large scale is ranging between 30 and above in its components. The second layer of the framework is the identified components in the application. There is need for variations in these components. The dependability requirements in the third layers, are parameters that describes the characteristics or measurable factors in defining the dependability in an IoT application.

7.4. Application of the Dependability Assessment Framework

The dependability assessment framework indicates the stages and processes involved in assessing the dependability of an IoT application. This stages and processes was achieved from the logical reasoning and deductive analysis of the dependability requirements of an IoT application. In the design of the dependability assessment framework, effectiveness in the application of the framework was put into consideration and steps were followed during the developmental process to ensure that the framework fulfils its intended usage.

7.4.1. Framework Application Process

At this stage, the two main targets are considered during the development of this framework: The system developer and the system analysist. However, is it envisaged, that this

framework can be used by also the end-users in assessing the dependability of an IoT application. The application process of this framework is described below:

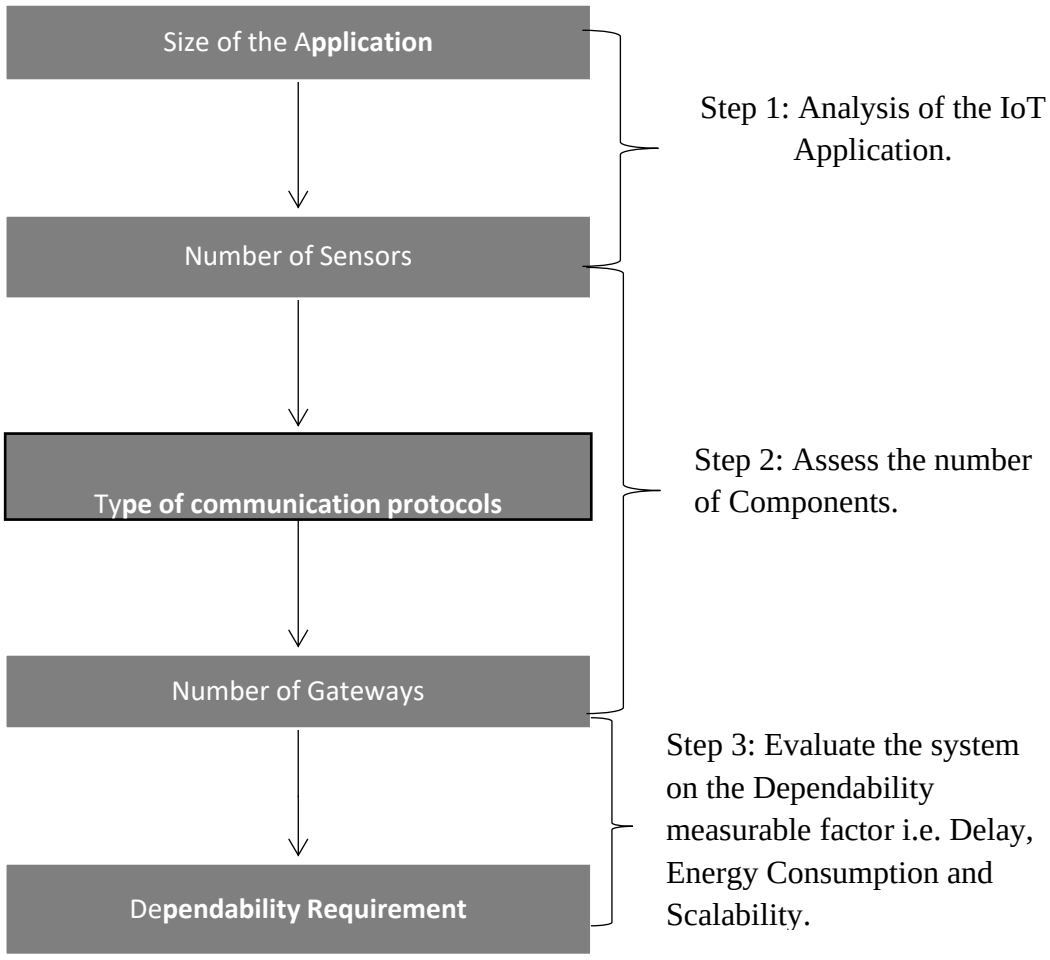


Figure 7.4: Framework Application Process

In figure 7.4 above, the steps and processes in applying the dependability assessment framework is shown. The cases below are representation of the usage of the dependability assessment framework in assessing the dependability of an IoT application. These cases are divided into two cases the system developer and the system analyst:

Case -1 System Developer

A system developer in this case is someone who intends to build an IoT application. The first step in this process is the analysis of the components that is intended to be used in the construction of the IoT application in the case of a system developer using the dependability assessment framework. Assess the number of sensor node that will be in the application, the type of communication protocol that will be used in the application and the number of gateways to be used in the design of the application. After the critical assessment of the

components using the steps in the dependability assessment framework, then the system model can be built by the developer. These whole considerations will ensure that the IoT application will fulfil the dependability requirements, as stated in the third layer of the framework i.e. effectiveness in energy consumption, low delay of packets and scalability of the gateway device.

Case -2 System Analyst

The system analyst, in this case is someone who intends to test an existing IoT application with the provisions of the dependability framework. The first step, the system analyst embarks on is to analyse the existing components in the IoT application. Assess the variations in the number of sensors used in the design of the application, the type of communication protocols used in the construction of the application and the number of gateways contained in the application. In following the processes and stages, contained in the dependability assessment framework the system analyst can ascertain the level of the dependability of the deployed system.

7.5. Summary

In achieving a dependable IoT application, the steps and processes stated in the framework are essential parameters required in assessing the dependability of an IoT application. The size of the application is a major determinant in this framework. This can be ascertained, through an assessment of the number of components contained in the application and the type of communication protocol. The dependability requirements in the framework are essential characteristics, required in the effective service delivery. This framework is intended to be tested using the exemplary cases as stated above. For the designing of an IoT application and in assessing an existing IoT application, through simulation experiments and use case evaluation

Chapter 8. Evaluation of Dependability Assessment Framework

8.1. Introduction

This chapter presents the evaluation and findings of the dependability assessment framework. Section (8.2) consists of the experimental design used in setting up the test, which include the network topology and the simulation environment. Section (8.3) shows the scales and parameters used in the measurements. This is followed by Sections (8.4), (8.5) and (8.6) showing the series of experiments conducted and the results of the findings. Thereafter, an analysis of the findings is presented in section (8.7). A use case evaluation is performed in Section (8.8) using the provision of the dependability assessment framework. Figure (8.1) below depicts the processes of the framework evaluation process with the steps involved.

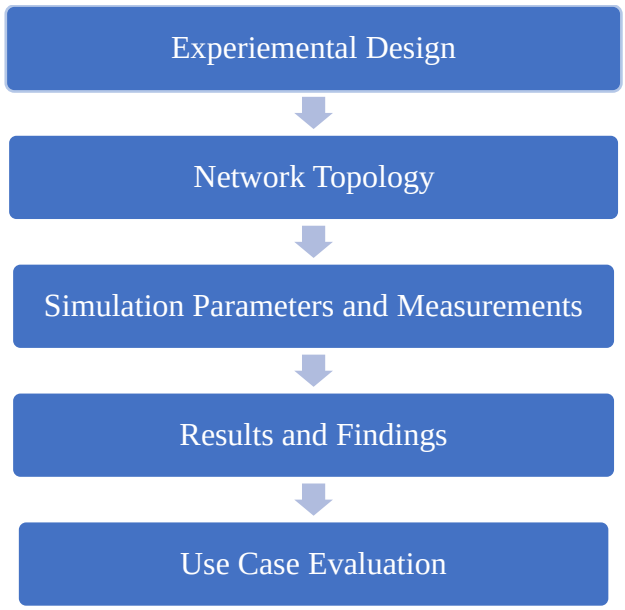


Figure 8.1: Processes involved in the evaluation of the dependability framework

The dependability evaluation framework is evaluated using the processes stated in the above figure (8.1). The evaluation process begins with a consideration on the design of the experiment with a consideration on the network topology to be used in the simulation environment. The test parameters for the simulation experiment are derived from the assumptions and analysis conducted in chapter five of this thesis. Thereafter, the results of

the simulation are presented with an analysis of the findings. This will be explored in the subsequent sections of this chapter.

8.2. Experimental Design

The experimental design in this evaluation was focused on three key areas: the experimental scenario, the experimental variables: the number of sensors, the communication protocols and the gateway. The experimental measurements, are the dependability requirements of an IoT application as stated in the dependability assessment framework.

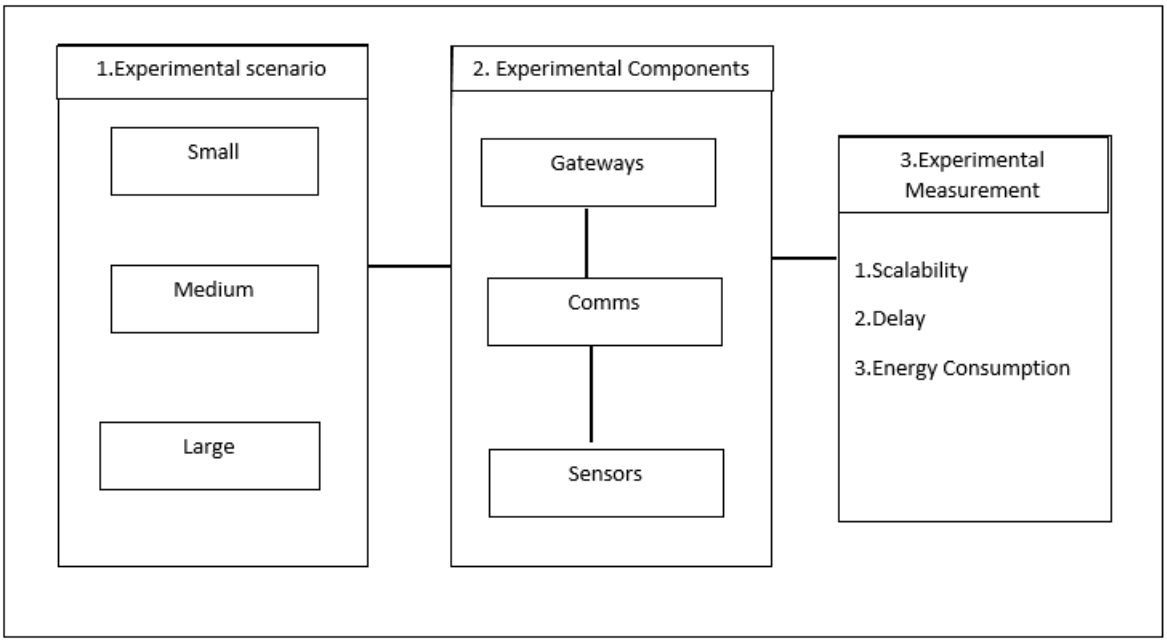


Figure 8.2: Experimental design

The experiments were conducted on the three scales of IoT applications: small, medium and large scale type of IoT application. As indicated in the dependability assessment framework, a small scale IoT application consists of not more than 15 components, while the medium scale is an application with not more than 30 components and finally a large scale IoT application, is an application with over 30 components as shown in the dependability assessment framework. The sensor nodes were used to acquire the packets in the simulation environment. The communications protocols highlighted in chapter five were used in the transmission of the packets to the gateway during the simulation. The measurement in this simulation is accessed in line with the dependability requirements of an IoT application.

8.2.1. Network Topology for Experimental Design

The networking standards being used today in IoT can be categorised into three basic network topologies: point-to-point, star, and mesh (Escobar et al 2019). When considering choosing a network topology there is need to understand the networking attributes of each of the chosen topologies. This include the latency, throughput, fault resilience, number of hops, the range and the number of nodes that can be included in a single network application. A typical design of an IoT application involves the effective communication of the sensor nodes to the gateway devices (Beaulah 2017 & Escobar et al 2019).

Experiment in this study involves small, medium and large-scale IoT application deployments. Choosing a suitable topology is challenging as it can be assumed that different topologies might be suitable in different application scale (Zhang et al 2012). However, according to Namiot & Sneps-Sneppe (2014) the star topology is the most suitable network topology for the effective deployment of the various sizes of IoT application. The performance of a star network is consistent, predictable and fast (low latency and high throughput). In a star network, unlike the mesh network, a data packet typically travels one hop or two hops to reach its destination, yielding a very low and predicable network latency. Secondly, there is high overall network reliability due to the ease with which faults in devices can be isolated. Each sensor devices utilises its own, single link to the hub. This makes the isolation of individual devices straightforward and makes it easy to detect faults and to remove failing network components (Dong et al 2015 & Masi 2018).

Mesh network is a similar network topology often utilised in the design of IoT. However, in mesh topology, sensors do not only capture and disseminate data, but also serves as relays for other nodes creating a collaboration with the neighbouring nodes to propagate the data through the network. The nodes that are configure in a mesh network are deployed in such way, that every node is within a transmission range. Data packets in a mesh network pass through multiple sensor/routers to reach the gateway node (Liu et al 2017). Therefore, failure in the corresponding node can lead to the entire failure in the application. Adding a new device due to failure of a device can lead to complication in the operation of the system, unlike star network when the sensor nodes communicate directly to the gateway creating a high level of functional dependency of the sensor nodes in the application. In such scenarios, a failure occurs in the sensory device, the other nodes can continue correspondence to the gateway as there is no intermediary node, but rather a direct communication (Yang et al 2016 &Liu et al 2018).

Furthermore, the star topology helps to avoid data collision as it contains less hops and even if one sensor node fails, the others can continue to sense the available data, the use of star topology gives the avenue of continuity of service, irrespective of a failure of one or more devices in the application (Invidia et al 2019).

The star configuration is also compatible with most of the popularly deployed communications protocols that have been utilised in the design of IoT application (McGrath 2014 & Invidia et al 2019). The use of star topology in this experimental setup will help in maintaining the scope of this study. However, IoT network can be in an unstructured or in a structured network (Mamta 2014). An unstructured network does not have a fixed topology, while a structured network has a fixed topology. In the case of this network design, a fixed structured network was created using the star topology as the sensors were placed in a fixed location.

8.2.2. Simulation Environment

The simulation environment for the test performed is achieved using the Omnet++ simulation component-based framework. OMNeT++ is an open-source discrete event simulation environment. The network component is referred to as modules which are written in C++ language. The module vector consists of the sensor nodes. The module also represents the protocol in the communication layer which is organised to construct the host and network device. In addition, the module also has the function for holding data, facilitating module interaction and mobility of the network device. The modules are interconnected through gates and exchange information by passing messages through these gates. The message represents the data, packets or control signal during communication. The modules are then assembled into a larger component to become a network environment using network description language.

There are existing frameworks, built in OMNeT++, which provide modules for various protocols. The advantage of these frameworks is that they enable new models to be developed by modifying the existing frameworks. Among the frameworks is the Castalia, inet, mixim and inetmanet which can be used for the simulation of an IoT network. In the experiment, the inetmanet framework was used in the development of the experimental scenarios. The construction of the parameters were achieved through the ned file (.ned), message definition (.msg), configuration (.ini) file and module description (.cc and .h) files. Section 8.3 below shows a detailed description of the test scale and parameters that were used in the simulation.

8.3. Detailed Description of The Test Scales and Parameters.

The scale for the test carried out are divided into small, medium and large-scale applications. This is to create an in-depth investigation into all the cases and to create a correlation that examines the relationships and difference between the cases researched through a series of test to get accurate results that portray the true characteristics of the operation of an IoT application. The parameters used in this test are set to create a distinction in the output of the test conducted. The subsections below are the descriptions of the application scales and the parameters used in the simulation set up.

8.3.1. Small Scale Application Setup

In this simulation set up 10 sensor nodes are assumed to measure changes in the environment and send the acquisition packets to the gateway through the various communication protocols. A total of 1000 packets will be used in the measurement scenario during the simulation. The main reason why 1000 packets are used in this simulation is to create a scope of measurement in the test to achieve a concise results. Figure 8.3 below is a representation of the configuration setup using the star topology for a small-scale application.

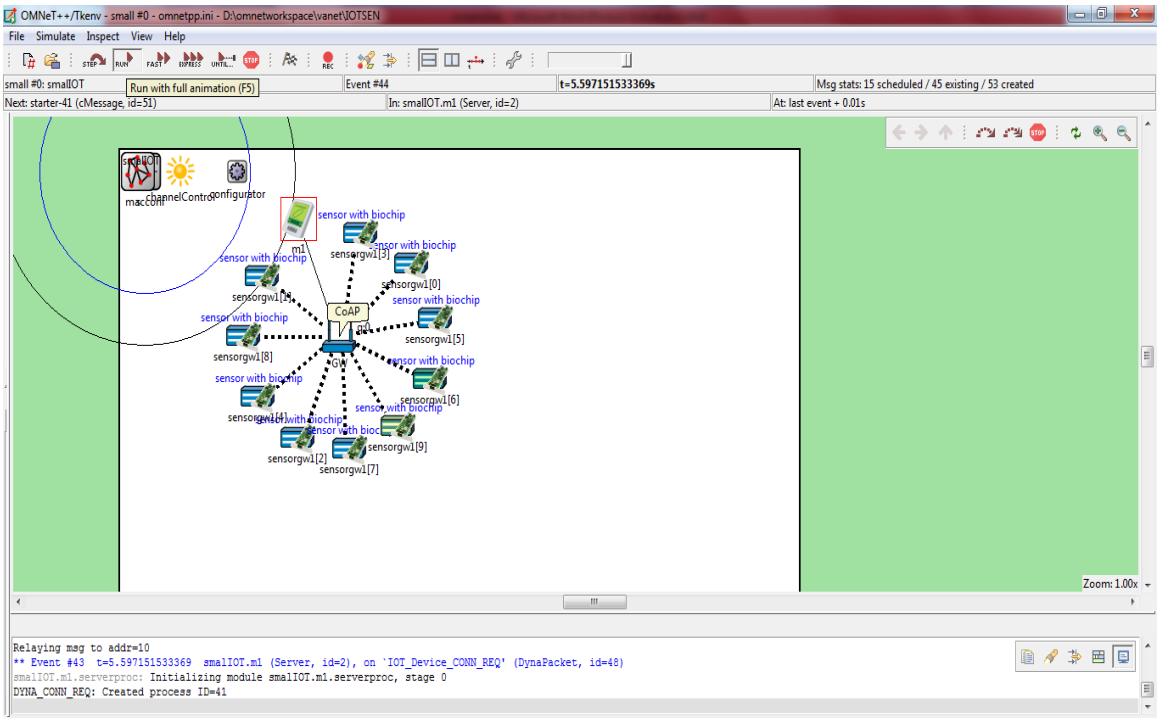


Figure 8.3: A representation of the small-scale configuration

The sensor nodes communicate with the gateway nodes through the communications protocols used in the setup. The communications protocols used in these tests, are Bluetooth,

ZigBee, Wi-Fi, MQTT and CoAP. These communications protocols were used in the transmission of the packets from the sensor node to the gateway device.

Table 8.1. Parameters and values for the simulation experiment

Parameters	Values
Number of sensors	10 sensors
Communication protocols	Bluetooth, ZigBee, WiFi, MQTT and COAP
Number of gateways	1 gateway
Total number of packets	Maximum packets 1000

Table 8.1 shows the parameters and values used to obtain the required metrics of measurements. The main input parameters and values in this setup are the number of sensors, the number of gateways and the total number of packets contained in the entire application scenario. The results of this experiment are determined by the measurements and metric that was calculated using this setup. This design created a consistent measurements during the simulation experiment.

8.3.2. Medium-Scale Application Setup

In this setup, between 20 and 25 sensor nodes is assumed to monitor data and send the acquisition data to the gateway through the various communication protocols. In this experimental setup, the unit of measurement was achieved at 2000 to 2500 packets. This was done to create a scope of measurement to be able to get consistency in the result output as random packet transfers in experiments will generate random results; therefore, the stipulated number of packets will lead to an accuracy in the output result. The figure (8.4), below shows a representation of the configuration.

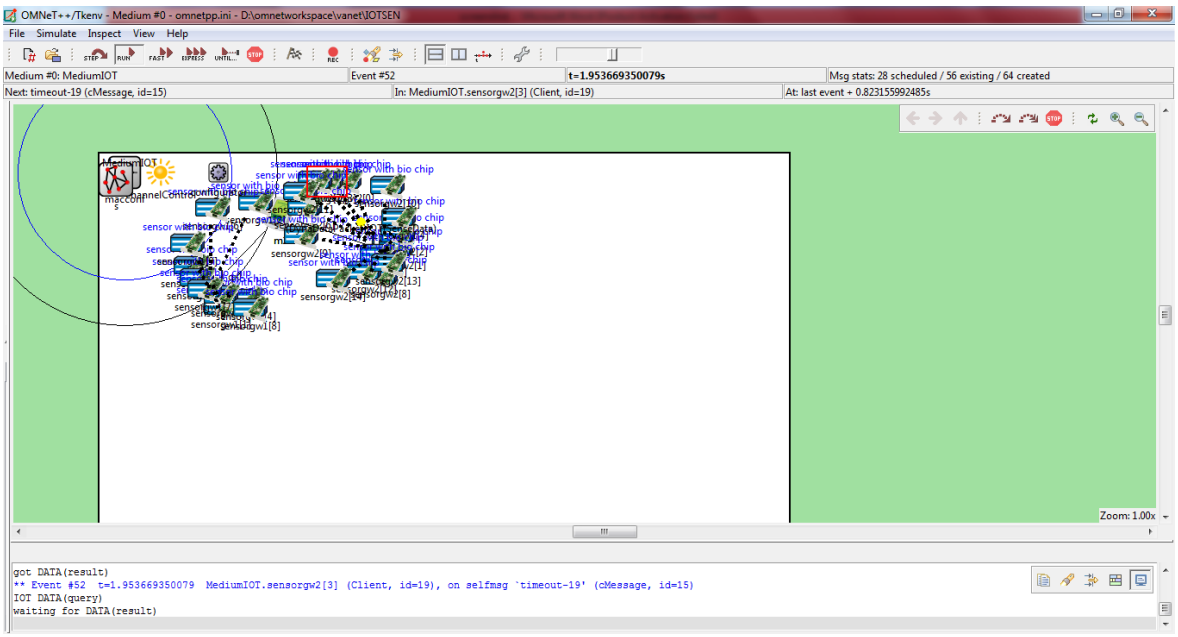


Figure 8.4: A representation of the medium-scale configuration

The structure consists of the primary component such as the sensors and gateway which is connected through a subset of sensors; through clustering, the sensors can communicate with the gateway using the wireless communication protocols and the management of these components is achieved through the design of the network through the star topology form. It is highly acknowledged that there are other types of network topology setup which include the mesh, tree and so on as the application demands, but to address the research needs of this research thesis, the sensors need a more direct and interrupted network design; so, for this reason, the star topology was used in the setup. The sensor nodes perform primary tasks of the device such as data acquisition in forms on packets from the physical layers and transmission of the acquired packets is accomplished through the established communications protocols. Table 8.2 below, is a representation of the parameters and values used for the simulation experiment.

Table 8.2. Parameters and values for the simulation experiment

Parameters	**Values**	
	Scale 1	**Scale 2**
Number of sensors	20 sensors	25 sensors
Communication protocols	Bluetooth, ZigBee, WiFi, MQTT and COAP	Bluetooth, ZigBee, WiFi, MQTT and COAP
Number of gateways	2 gateways	2 gateways

Total number of packets	2000 packets	2500 packets

The parameters and values used for designing these simulation experiments consist of the components of an IoT application (sensors nodes, the gateways and the communications protocols). The ranging of the sensors and gateways used in the application was a major determinant of the results output when calculating the metrics of measurements in this test.

8.3.3. Large-Scale Application Setup

In the design of this test, the sensor nodes that were deployed in the simulation environment to monitor the environment to get the required data was between 50 and 70. The number of packets for the sensing in the simulation experiment was 5000 packets for a total of 50 sensor nodes and 7000 packets for 70 sensor nodes. The number of gateways contained in both scenarios was 5 gateways. The sensor nodes transmit these packets to the gateway using the established communication protocols that have been implemented in the test environment as shown in figure 8.5 below.

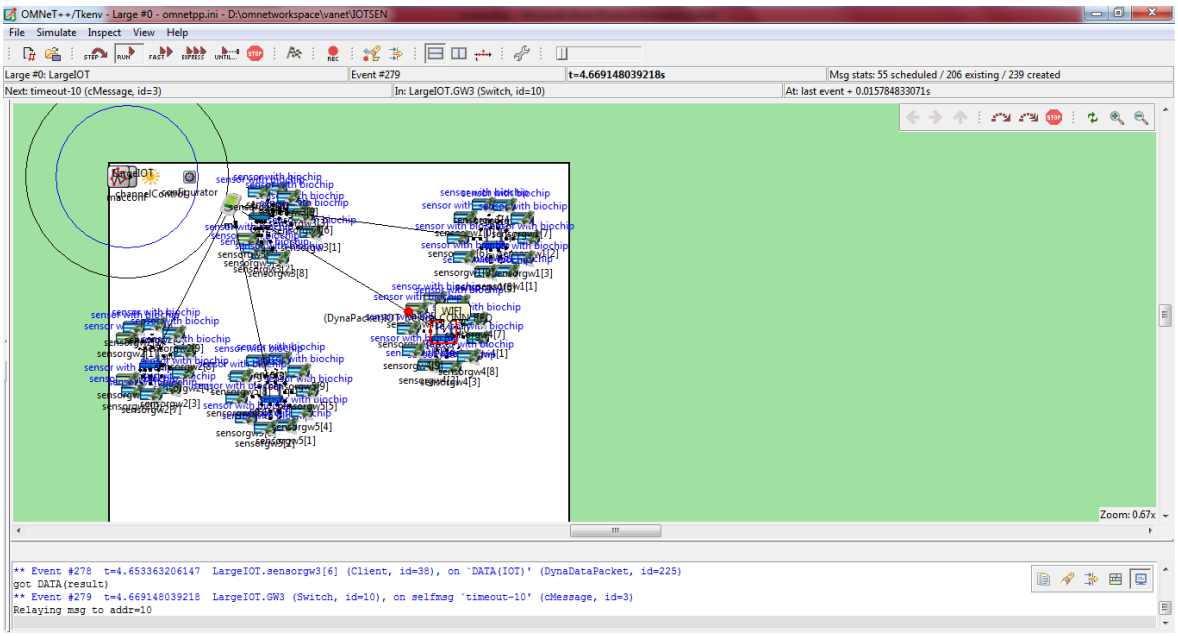


Figure 8.5: A representation of the large-scale configuration

The configuration of this setup comprises of five clusters the sensors which are classed as physical objects in the network, are configured to the gateways which receive the transmitted packets through the various communications protocols. The communication network represents the connectivity between the sensors and the gateway. The physical layer of the platform incorporates wireless sensor nodes, each of which encompasses a set of sensors linked wirelessly to a mobile device. The parameters and values used for this test are represented in table 8.3 below.

Table 8.3. Parameters and values for the simulation experiment

Parameters	**Values**	
	Scale 1	**Scale 2**
Number of sensors	50 sensors	70 sensors
Communications protocols	Bluetooth, ZigBee, WiFi, MQTT and COAP	Bluetooth, ZigBee, WiFi, MQTT and COAP
Number of gateways	5 gateways	5 gateways
Total number of packets	5000 packets	7000 packets

These values consist of the number of sensors used in the design of the application, the number of gateways used in the application and the total number of packets that are available during this test. The communications protocols highlighted above are used in the transmission of the packets. The measurements and metrics as stated in the dependability assessment framework are created to achieve the results of the experiment. The sections below present the results and findings of the test.

8.4. Experiment on Energy Consumption

The goal of this experiment is to measure the energy consumption of the various communication protocols for the transmission of packets from the sensors to the gateway device. This experiment shows that the energy consumption of a sensor node could adversely be affected by the type of communication protocol been used for the data transfer, as sensor's major form of communication is through protocols. Therefore, for sensor's not to fail, and be dependable during its operation there is high need for a low energy consumption protocol.

The energy consumption for this experimental setup was measured as shown below. A similar measurement of energy consumption in wireless sensor network was performed in the research study of (Li et al 2014).

The total energy consumption for sending a packet is measured with the following:

$$Epacket = Etransmit + Ereceive$$

Considering the energy consumption of sending a packet from the sensor node to the gateway device (A to B) through the communication protocol:

Where,

Epacket is the energy consumed by the packet,

$Etransmit$ is the energy used by communication protocol in transmitting the packets,

$Ereceive$ is the Energy at receipt.

8.4.1. Results and Analysis of the Energy consumption

The results of this experiment show the energy consumption of the various communications protocols during packet transmission from the sensor nodes to the destination gateway. The results of this experiment show the effectiveness of a communication protocols in regard to the energy consumption of the sensor node. The energy consumed in sending of the packets was ascertained. The transmission energy was set to maximum for all the protocols. The gateway packet acknowledgement was required. The distance between the slave and the master was fixed at 30cm for the small scale, 50 cm for the medium scale and 100 cm for the large-scale experiment. The packets of measurement, ranges from 1000, 2000, 2,500, 5000 and 7000 respectively for each test scale. Encryption of packets was disabled; the energy supply was unlimited. The active energy of the nodes and the gateway during transmission were measured and analysed. The sleep state of the nodes and the master was not required. The simulation results for this experiment are presented below:

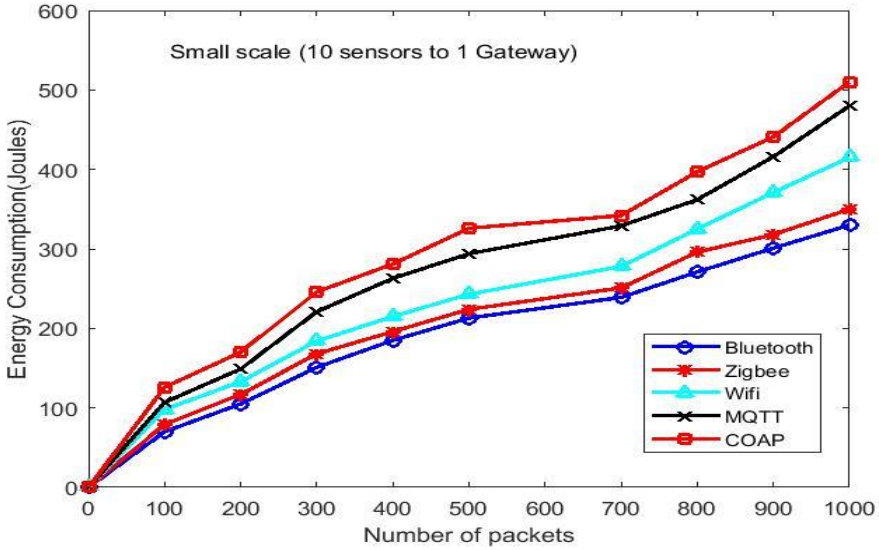


Figure 8.6: Experiment results of small-scale IoT application

The results in (Fig 8.6), show the energy consumption of the various communication protocols that were used different during the transfer of packets from 10 sensor nodes to 1 gateway. It is clear that Bluetooth has the lowest level energy consumption using of 330 (J) in sending 1000 packets, while ZigBee uses a total of 350 (J), Wi-Fi uses 415 (J), MQTT uses 480 (J) and COAP uses the highest energy consumption of 510 (J) in sending 1000 packets. When it comes to small scale IoT application between 0 to 10 sensors with less amount of packet transfer, Bluetooth has the lowest level of energy consumption as compared to the other communications protocols utilised in this experiment.

Bluetooth and ZigBee have much in common. Both protocols are of IEEE 802.15 "wireless personal-area networks," or WPANs and run in the 2.4-GHz unlicensed frequency band, which uses low energy consumption. Bluetooth (IEEE 802.15.1), ZigBee (IEEE 802.15.4) and Wi-Fi (IEEE 802.11) are the three emerging wireless technology for short range and low energy wireless communications while MQTT and COAP are specifically designed for large scale applications.

Bluetooth is a wireless network communication protocol specifically designed to provide short range, low power wireless connections and allow devices to form ad hoc personal area networks (PANs) with other equipped devices in the network infrastructure. Bluetooth application is specifically for a short range of 10 meters or optionally a medium-range of 100 meters radio link of data transmission with a maximum capacity of 720 kbps per channel and throughput of 1Mbit per seconds.

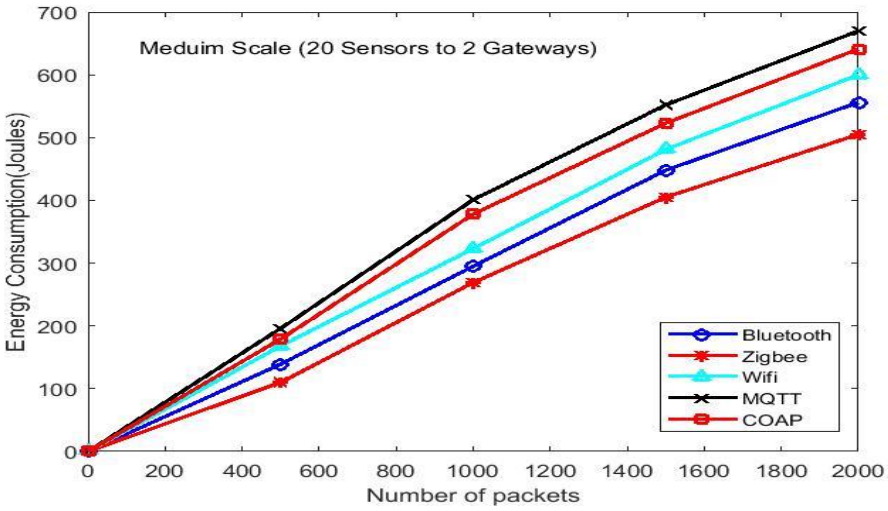


Figure 8.7: Experiment on energy consumption on medium-scale 1

From the observations of the results as shown in (Fig 8.7), on a medium scale IoT application between 0 to 20 sensors connected to two gateway receiver in the application, ZigBee uses a total of 496 (J) of energy in processing 2000 packets slightly lower than Bluetooth which uses a total of 536 (J) in processing the same amount of packets while Wi-Fi uses 600 (J) and MQTT and COAP using a range of energy consumption between 670 to 720 joules which is on the high side in the regards to the amount of packets transfer. MQTT and COAP are communications specifically designed for large scale application.

ZigBee communication protocol tends to performs better in regards to energy consumption during packet transfer unlike the other communication protocols, ZigBee is a local area network and can operate a typical 0 dBm low power ZigBee radio transmitter, unlike Bluetooth which is more of an ad hoc personal area networks (PANs). ZigBee uses a direct sequence spread spectrum (DSSS) technology of working frequency between 868MHz, 915MHz and 2.4GHz and is efficient for packet communications with small volume and of lower data transfer efficiency. It is not only intended to connect to devices directly around a user within a short range but can also connect need a wider range with makes it an ideal protocol for home automation and smart lighting.

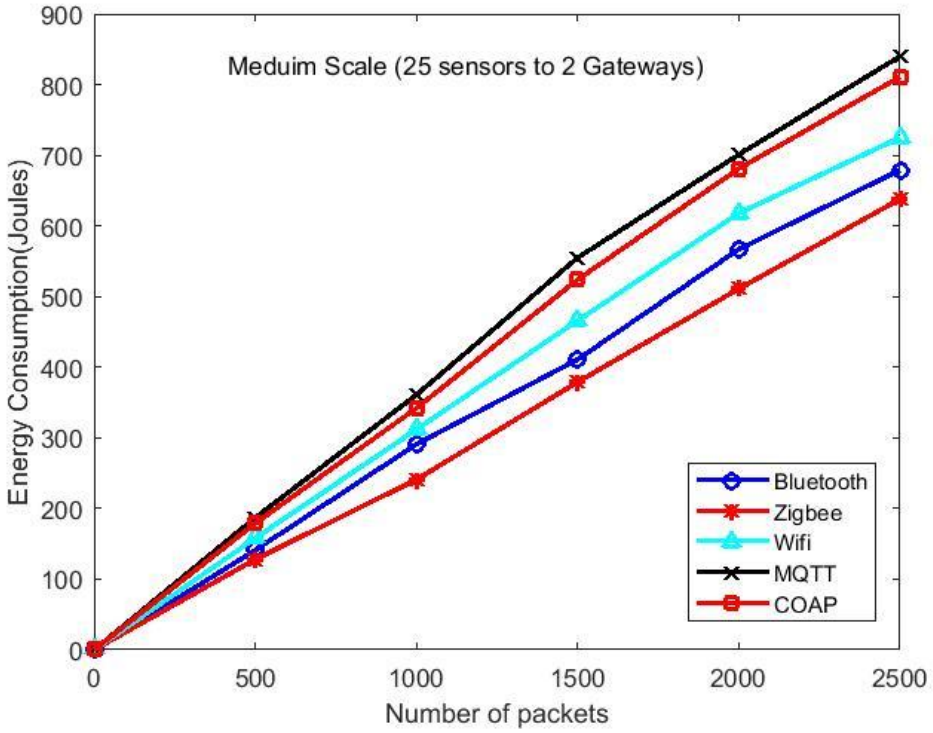


Figure 8.8: Experiment on energy consumption on medium-scale 2

Form the observation of the results presented in (Fig 8.8), a total of 25 sensors nodes were configure to 2 gateway receiver, ZigBee performed better than the other communication protocols in the transmission of 2500 packets. ZigBee used a total of energy of 638 (J) in sending 2,500 while Bluetooth uses 672 (J), Wi-Fi uses 725 (J), MQTT uses 811 (J) and COAP uses 840 (J) during the system operation. ZigBee supports a data rate of 250 kps which is lower in energy consumption compared to Bluetooth that supports a data rate of approximately 24Mbps which is a higher data rate compared to ZigBee.

The results of (FIG 8.6 & 8.7), shows that the lower the data rate, the lower the energy consumption in a small scale, because when the packets were limited to 1000 in the application. Bluetooth performed better in regards to consistency in energy consumption but if more packets are increased it will consumes higher than ZigBee. In contrast ZigBee represents 250kbps data rate. It has lower data rate, low energy consumption and works with increased number of packets. And for Wi-Fi, the data rate is 54 Mbps, higher data rate, higher energy consumption than Bluetooth and ZigBee, WiFi is largely used to provide high speed to the internet access or local area network devices. Wi-Fi, MQTT and COAP provides higher throughput and covers a great distance and need higher energy, on the other hand ZigBee and Bluetooth provide lower throughput and uses low energy.

The research studies of (Khan 2010), indicate that despite the wider range of ZigBee communication protocols it still fairly limited and isn't the best choice for highly and critical instrumented installation like industrial IoT applications, and can cause bottlenecks when configured with multiple or large sensor nodes. This will be explored further in the course of this research study through series of experiments.

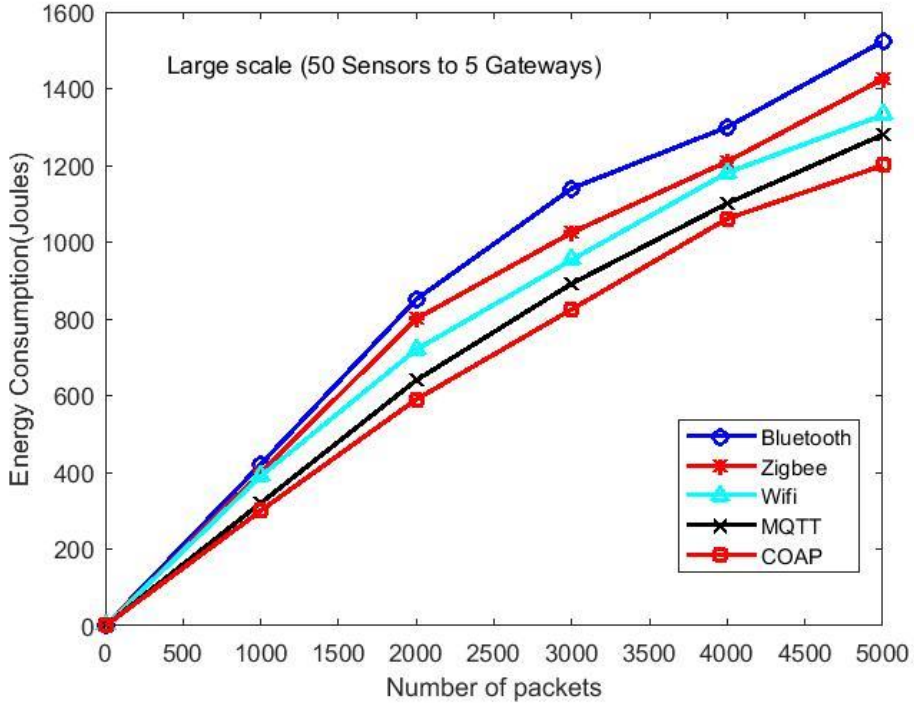


Figure 8.9: Experiment on energy consumption on large-scale 1

The result of the experiment in (Fig 8.9), on 50 sensor nodes to 5 gateways in sending a total of 5000 packets, shows that the energy consumption of COAP is optimised, which is slightly lower than the energy consumption of MQTT and a higher energy consumption of Wi-Fi, ZigBee and Bluetooth. The energy consumption of COAP, is lower than the energy consumption of ZigBee and Bluetooth as compare to the small and medium scale applications. From the observations in this experiment there is a great margin due to the fact that COAP and MQTT are communication protocols specifically designed for IoT and Machine to Machine applications with a very high throughput and data rate for large transmission with a large volume of packets as compared to ZigBee and Bluetooth which are specifically design for small volume of packets and small scale applications with short distances.

In wireless communication, in general when the transmission distance is short there is low amount of transmission energy spent by the node, the shorter the transmission distance the reduction in the energy consumption. This a major contributing factor of the effective performance of Bluetooth and ZigBee in the small and medium scale experiment. However, when considering wireless multi-hop network or large-scale application as shown in this experimental result (Fig 8.9), a communication protocol with a shorter transmission distance increases the total energy consumption, since the transmission requires large node and increased hop count between the sensors and the gateway (Chen et al 2014).

Using a high data rate communication protocol like the case of COAP and MQTT decreased the transmission time of the packets, which in turn decrease the energy consumption. A higher data rate communication protocols generally has a shorter maximum transmission time and thus decreases the hop count for data transmission. A decrease in hop count would lower energy consumption since the number of packet transmitted will get to the destination gateway in less amount of time.

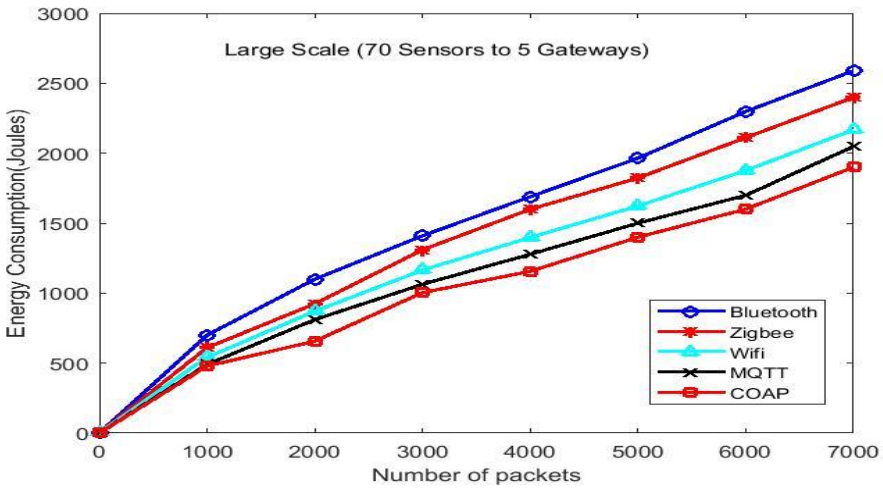


Figure 8.10: Experiment on energy consumption on large-scale 2

The results presented in (Fig 8.10), show a total of 70 sensors nodes configured to 5 gateway receiver, showing the reliability and efficiency of the various communication protocols in regards to the energy consumption in the transmission of the packets from the source node to the destination. COAP uses an energy of 2011 (J) in the transmission of the 7000 packets while MQTT uses 2087(J), Wi-Fi uses 2196 (J), ZigBee uses 2248(J) and Bluetooth uses 2298 (J) of consumed energy during the system operation.

The observations of the results (Fig 8.7 and Fig 8.8) creates a degree of certainty, that higher data rate communication protocols like COAP and MQTT reduces energy consumption for large scale IoT application which is in contrast to small and medium scale where lower data rate communication protocols tend to produce lower energy consumption. When there is high density of communication components there is need for high data rate communications protocols to increase the transmission time of the packets and thus reduce the energy consumption, but where there is low density of communication components then the need for a low data rate communications protocols like ZigBee, Bluetooth and Wi-Fi arises as shown in the above experiments. From this experiment it is observed that every communications protocols has an advantage as well as disadvantages.

In the experiment conducted shows some variations in the rate of energy consumption of the entire application using the various communications protocols, with the different application scales in regards to the sensors used to send packets to the gateways. In this experiment there is an observation that the more the number of workload in the application the more the amount of energy spent in the sending the packets and also that the capability of each communication protocol and their data rate has an impact in the energy consumption in the application in regards to efficient and reliable packet transmission in IoT application. Therefore, in designing an IoT application it is critical to check the communications protocols in regards to the number of components used in designing the application in creating an efficient and reliable transmission of packets to avoid failures and minimising energy during the operation of the system for the system to be dependable. The result of the large scale experiments shows that the less transmission time delay, the higher the quality and efficiency in the energy usage of an IoT application. Table 8.4 is a summary of the results and findings.

Table 8.4. Summary of the Results and Findings

COMMS	ZIGBEE	WiFi	MQTT	BLUETOOTH	COAP
Small scale					
No of Nodes	10	10	10	10	10
No of Gateway	1	1	1	1	1
Available energy	Max	Max	Max	Max	Max
Total number of Packet	1000	1000	1000	1000	1000
Exp results	350	415	480	330	510
	Low	Medium	High	Low	High

Medium Scale

	ZigBee		WiFi	MQTT	Bluetooth				COAP	
No of Nodes	20	25	20	25	20	25	20	25	20	25
No of Gateway	2	2	2		2		2		2	
Available Energy	Max	Max	Max	Max	Max		Max	Max	Max	Max
Total Number of Packets	2000	2500	2000	2500	2000	2500	2000	2500	2000	2500
Exp results	496	638	600	725	671	811	534	672	700	840
	Low	Low	Medium	Medium	High	High	Low	Low	High	High

Large scale

	ZigBee	WiFi	MQTT	Bluetooth	COAP

No of Nodes	50	70	50	70	50	70	50	70	50	70
No of gateways	5	5	5	5	5	5	5	5	5	5
Available energy	Max	Max	Max	Max	Max	Max	Max	Max	Max	Max
Total number of packets	5000	7000	5000	7000	5000	7000	5000	7000	5000	7000
Exp result	1425	2248	1331	2196	1280	2087	1523	2298	1200	2011
	High	High	Medium	Medium	Low	Low	High	High	Low	Low

8.5. Experiment on Transmission Delay in IoT

It is of immense importance to measure the performance parameters of Ad Hoc networks, personal area network and wide area network in regard to the end to end delay of the application in optimising the overall network performance of a system. Minimizing delay is also one of the vital conditions to ensure reliability of IoT application. Delay is the amount of time taken by the nodes to transmit the data packets from the source node to the destination through the use of communications protocols. Time synchronisation and time stamped are used in this experiment to measure delay in the entire application network. This is a lightweight approach to measure packet delay through the time of transmission of sending the packets to the gateway (response time), we define the delay as the time taken from the packet is generated at the source node to the time that the packet is received at the gateway (Parameswari and Sasilatha 2016).

Delay is measured with the following:

$$Delay = \frac{Time}{Load}$$

Where Time, is the transmission time taken to process the packet

And Load, is the number of components in the system, which can be represented as the transmission packets (Liu et al, 2013).

8.5.1. Results and Analysis of the Delay Metrics

This simulation results shows an investigation of the level of delay in the various scales of an IoT application ranging from the small, medium and large scale. This simulation results shows the delay and performance of the various communications protocols in regards to the number sensors and gateways used during packets transfer. The accessible parameters in this simulation experiment was made equal to create a degree of certainty and to ascertain the authenticity of the results.

Each transmission had one 8-byte data packet of arbitrary values with a fixed total number of packets. The transmit power was set to maximum. The gateway packet acknowledgement was required. The distance between the slave and the master was fixed at 30cm for the small scale, 50 cm for the medium scale and 100 cm for the large-scale experiment. The packets used for the measurement of the scale, ranges from 1000, 2000, 2,500, 5000 and 7000 respectively. Encryption was disabled and the energy supply was unlimited. The time of delivery of the packets during transmission was recorded. The simulation results for this experiment are presented below.

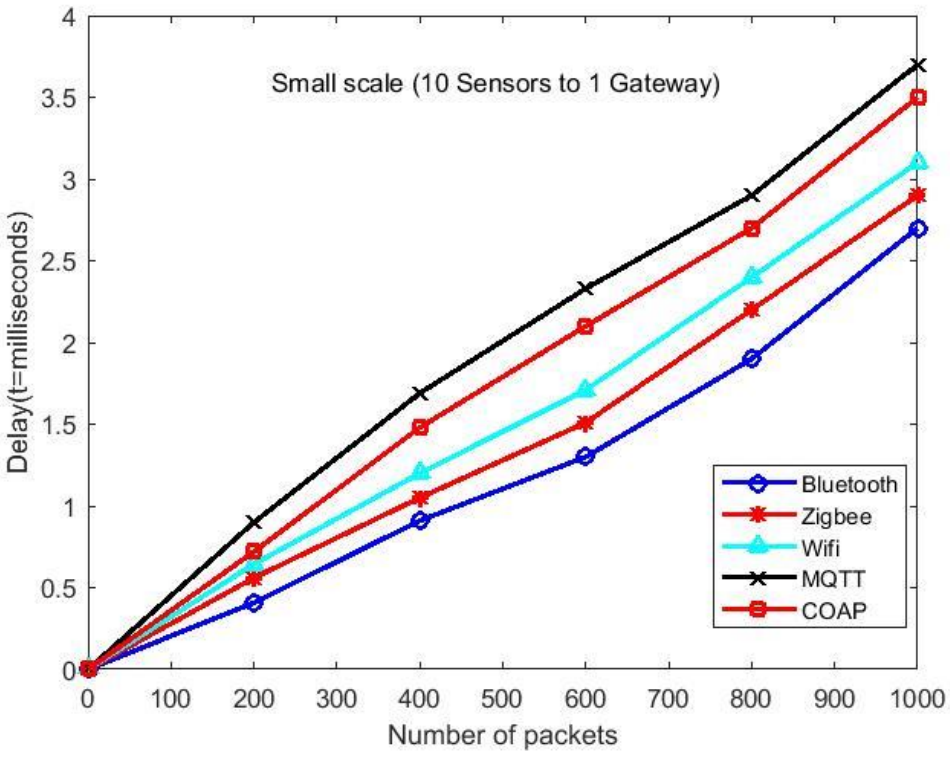


Figure 8.11: Experiment on delay using a small-scale scenario

The results in (Fig 8.11), shows the number of packets transmitted by the communication protocols with the time of delivery. From the results of this experiment it is observed that some communications protocols has higher throughput with less amount of delay for packets delivery as compared to the other protocols. Bluetooth was able to send a total of 1000 packets within 2.6 milliseconds, while ZigBee uses 2.7 milliseconds in sending the same amount of packets which is slightly higher in regards to the delay in packets transfer, as in the case of WiFi, COAP and MQTT this were recorded to have higher variations in time of sending the same amount of packets from the source node to the destination.

Bluetooth has the capability of an increase bandwidth of 2 Mbps. By doubling the amount of data that devices can transfer, this reduces the time required for transmitting and receiving data in an application and specifically design for short range system with low amount of packets and uses a 2.4 GHz ISM band with 40 channels for broadcasting purposes and packets transmission. Bluetooth is a short-range radio link intended to replace the cables connecting portable or fixed electronic devices. The key features are robustness, low complexity, and low power and is effective for small scale IoT applications.

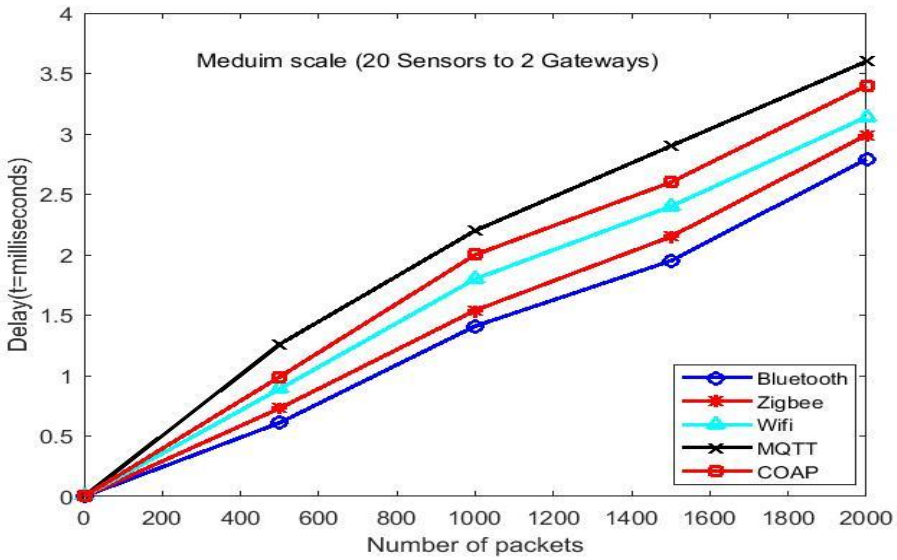


Figure 8.12: Experiment on delay using a medium-scale scenario

The results presented in (Fig 8.12), shows a total of 20 sensors nodes configured to 2 gateway receiver, showing the amount of transmission time for the packets to get to the gateway. From the observation in the results of the simulation as presented above its indicates that when it comes to small scale IoT application with less amount of packets, Bluetooth has a great tendency of high performance in effective packet delivery than the other communications protocols used in the experiment. Bluetooth was able to deliver a total of

2000 packets within 4.9 milliseconds which is quite a reasonable and slightly lower than ZigBee communication protocol which deliver the same amount of packet within 5 milliseconds. Bluetooth and ZigBee are communication protocols specifically design for short range application as compared to Wi-Fi, MQTT and COAP.

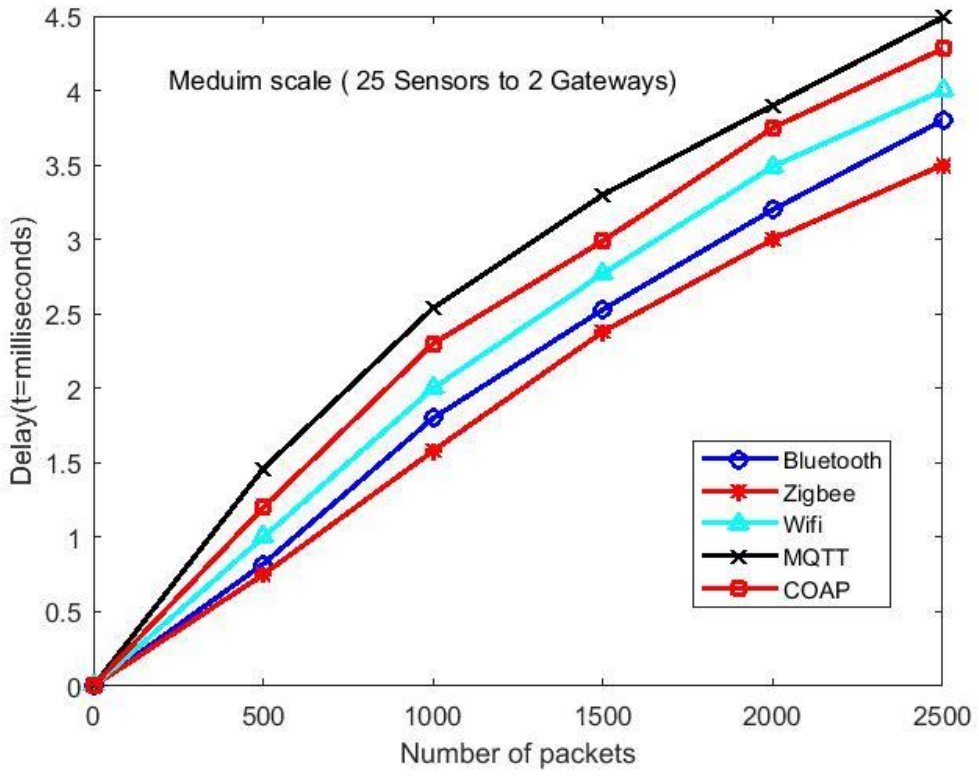


Figure 8.13: Experiment on delay using a medium-scale scenario

From the results of the experiment in (Fig 8.13), shows that when it comes to a larger amount of packets, ZigBee performs better in regards to the amount of time it takes to deliver the packet to the gateway as compared to the previous experiment where Bluetooth has a minimal amount of delay. This indicates that ZigBee is more effective with larger amounts of devices with a medium scale type of application than Bluetooth. Though they have similar throughput, ZigBee is more effective in multi-hop transmissions while Bluetooth is more effective in an adhoc network or in a one way connection.

The results of the experiment indicate that ZigBee and Bluetooth are more reliable and works faster in low dense IoT applications in regards to higher throughput than Wi-Fi, MQTT and COAP. Throughput is the rate of successful packet delivery over a communication channel the lower the throughput, the worse the network is performing. The IoT rely on successful

packet delivery to communicate with each other so if packets aren't reaching their destination the end result is going to be a poor service quality.

However, increase in the components of an IoT application will increase the transmission range and will inevitably cause higher interference which leads to the lower throughput. Thus, there is a trade-off between reducing the delay and improving the throughput of the network.

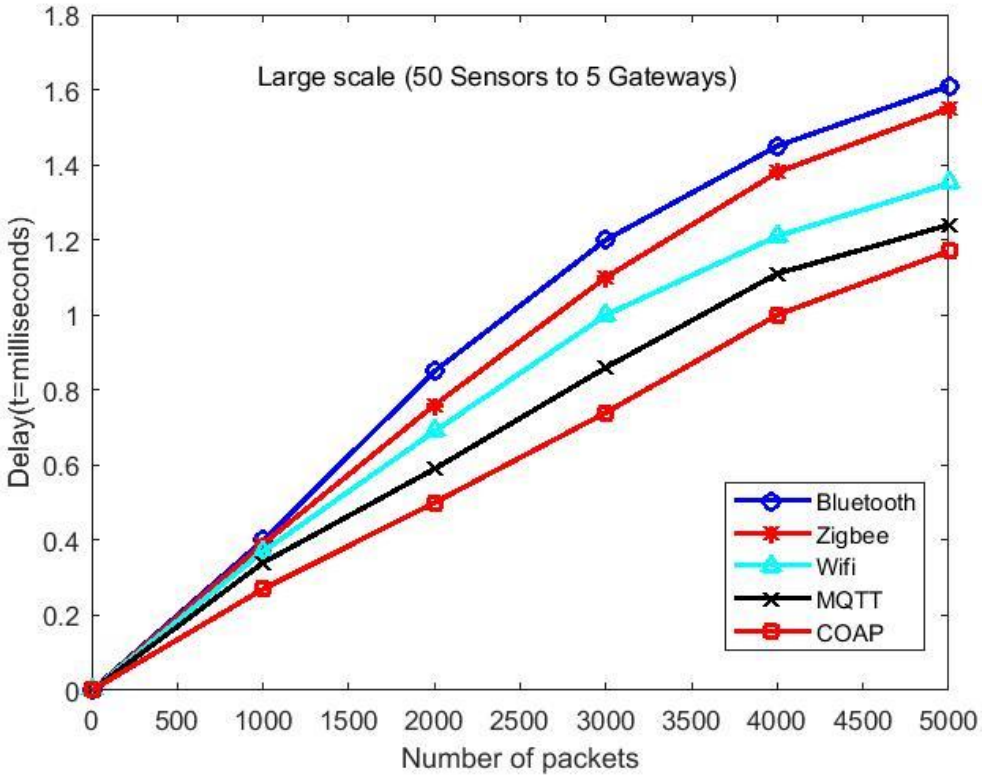


Figure 8.14: Experiment on delay using a large-scale scenario

From the observation of the result of the experiment in (Fig 8.14), on 50 sensor nodes to 5 gateway in sending a total of 5000 packets, shows that COAP has a better throughput of 1.5 milliseconds with less delay as compared to MQTT and Wifi with a lower level of delay as compare to ZigBee and Bluetooth. From the experimental results there is an indication of a great margin due to the fact that COAP and MQTT are specially design for wide area network which comprises of a large amount of devices in the application as compared to ZigBee and Bluetooth that are mainly for routing in a short distance with less devices.

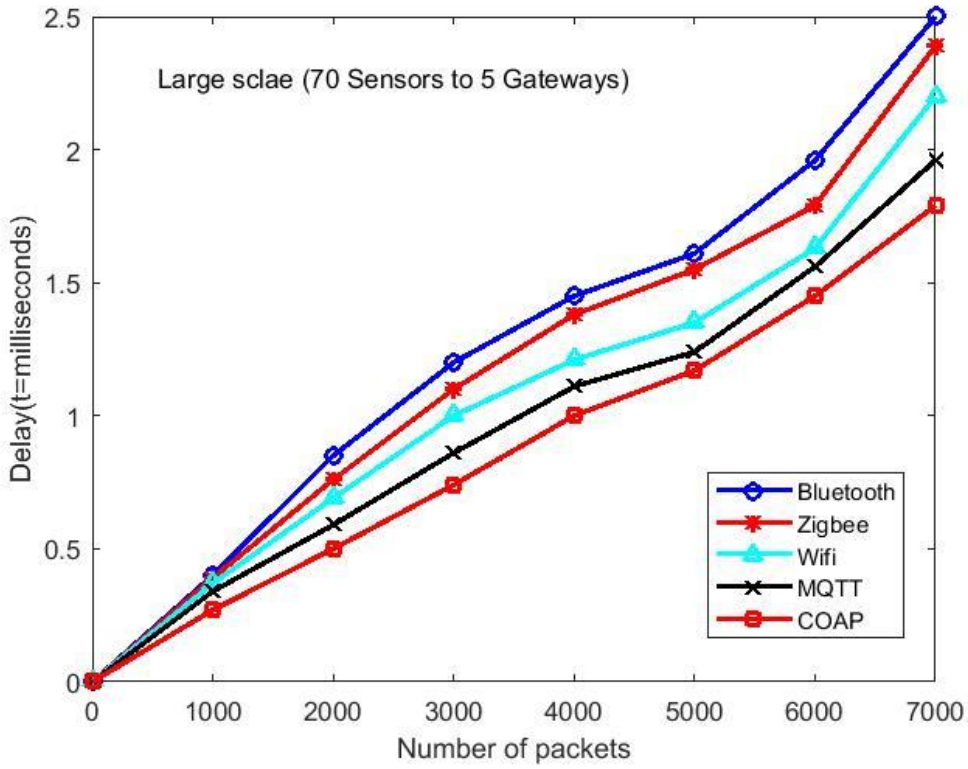


Figure 8.15: Experiment on delay using a large-scale scenario

The results of the experiment in (Fig 8.15), on 70 sensors nodes to 5 gateway in sending 7000 packets adds to the findings of (Fig 8.14) that COAP has a higher throughput when it comes to large scale IoT application with MQTT which has a slight deference in the performance. Wifi is another recommended communication protocols that has a considerable performance in large scale IoT application as to compare to Bluetooth and ZigBee in large scale application set up. This experimental results confirm that for faster communication of packets in a large deployments of the component of an IoT application the communication protocol has a great impact to the successful operation. Table 7.5 below is a representation of the analysis of the results and findings

Table 8.5. Summary of the Results and Findings on the Experiment on Delay

COMMS	**ZigBee**	**WiFi**	**MQTT**	**BLUETOOTH**	**COAP**
Small scale					
No of Nodes	10	10	10	10	10
No of Gateway	1	1	1	1	1
Total num of Packet	1000	1000	1000	1000	1000
Exp results	2.7	2.8	3.0	2.6	2.9
	Low	Low	Medium	Low	Medium

Medium Scale										
	ZigBee		**WiFi**		**MQTT**		**BLUETOOTH**		**COAP**	
No of Nodes	20	25	20	25	20	25	20	25	20	25
No of Gateway	2	2	2	2	2	2	2	2	2	2
Total Number of Packets	2000	2500	2000	2500	2000	2500	2000	2500	2000	2500
Exp results	5	6.3	5.1	6.2	5.3	6.5	4.9	6.6	5.2	6.4
Measured with Time (miliseconds)	Low	Low	Medium	Low	Medium	Medium	Low	Medium	Medium	Low

Large scale					
	ZigBee	**WiFi**	**MQTT**	**Bluetooth**	**COAP**

No of Nodes	50	70	50	70	50	70	50	70	50	70
No of gateways	5	5	5	5	5	5	5	5	5	5
Total number of packets	5000	7000	5000	7000	5000	7000	5000	7000	5000	7000
Exp Result	14.3	19.8	13.8	19.3	13.5	18.4	14.7	20.2	13.3	18.2
	High	Medium	Medium	Medium	Low	Low	High	High	Low	Low

8.6. Experiment on Scalability of IoT Application

The goal of this experiment is to measure the scalability of an IoT application focusing on the energy consumption and the delay of the application as the sensor nodes and gateway increases in the system. This experiment will show the system behaviour in delivery packets as the number of nodes and gateway increases. The scalability of the routing protocols, the sensor nodes and the gateway used in the transmission of the packets to the gateway in this experiment is a critical issue due to the extremely high node numbers and relatively high node density as the experimental scale progresses, hence there is need for a gateway management of the amount of packets been transmitted to the gateway to ensure efficiency in the energy consumption and the level of delay experience during the system operation.

8.6.1. Results and Analysis of the Scalability Metrics

This experiment a major investigation into the scalability issues in IoT application focusing on the energy consumption and the delay of an IoT application as the sensor nodes and gateways in the system increases. In achieving this measurement a major focus was done on the communication protocol used in the transmission of the packets as the components in the application increases. This experimental approach is designed to minimise the energy consumption and delay through the implementation of additional gateways in the experiment as sensor nodes increases the packets increases.

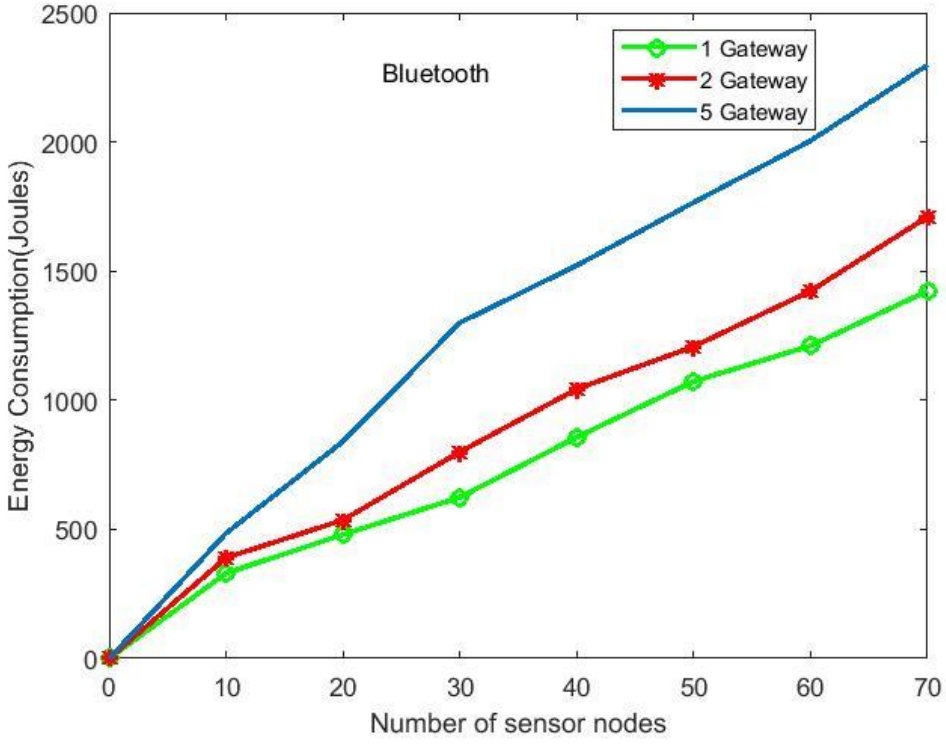


Figure 8.16: Experiment on energy consumption using Bluetooth

The results of the experiment as shown in (Fig 8.16), shows that energy consumption of an IoT application using Bluetooth communication protocols increases, as the sensor nodes increases in the application. The energy consumption of the entire application increases and this has a variation with the gateway that is been added in the application. The results shows that for 10 sensor nodes to 1 gateway the energy consumed is 330 (J), when the gateways progresses to 2 with same amount of sensor nodes (10) the energy consumption increased to 390 (J) and when the number of gateways increases to 5 the energy consumption increased to 495 (J).

While the energy consumption keep increases as more gateway devices are added to the application, the time of delivery of the packets to the gateway decreases. Showing a decrease in the delay of the packets to the gateway. This shows that adding more gateways to the sensor node will increase the energy consumption in Bluetooth communication protocol but decrease the delay time of the packets, although it is with great intent to vary the impact of the communication protocols on their energy consumption and delay as the application increases in their component size this will be achieved through series of experiments in this section. Figure (8.17), below is the results of the delay experienced using same parameters in regard to the number of sensors and gateway used in figure (8.16) in measuring energy

consumption. The reason for this comparison is to ascertain the impact of the components of IoT and their dependability.

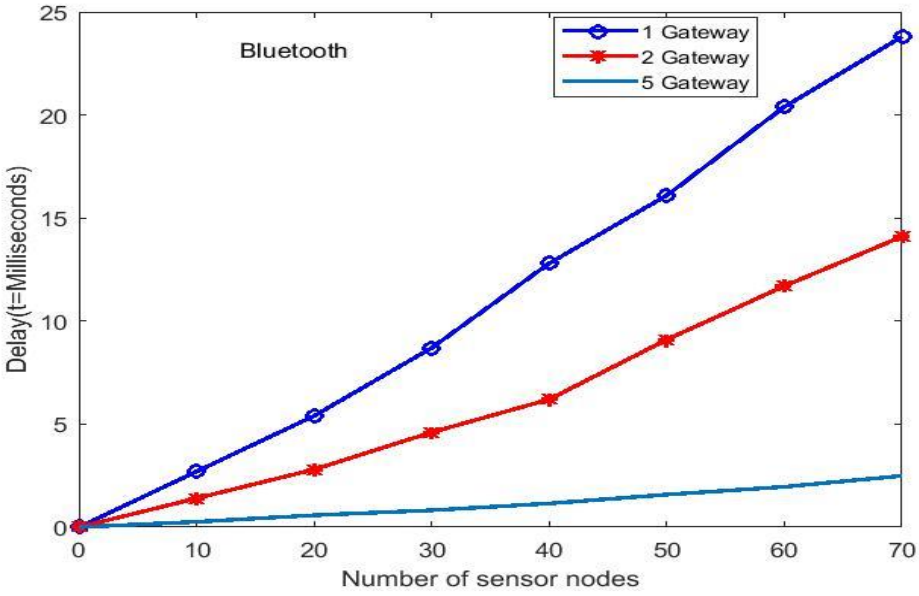


Figure 8.17: Experiment on delay using Bluetooth

As showed in (Fig 8.17), on the experiment on delay using Bluetooth, for 10 sensor nodes using 1 gateway is 2.7 milliseconds in sending a total of 1000 packets, when an additional gateway was added to the sensor node making it 2 number of gateway devices the delay in time of delivery the packets to the gateway reduced to 1.4 milliseconds and when an additional 5 gateway devices was configured to the 10 sensor nodes the delay in time of receiving the packet in the gateway drastically reduced to 0.26 milliseconds this shows that as more gateways are added to the number of sensor nodes the delay in packet transmission and receiving time automatically reduces.

From the observation in the experiment conducted in (Fig 8.6) and (Fig 8.7) shows that when more gateway devices are added to the sensor nodes the level of energy consumption increases but the delay in the transmission of the packets decreases. As stated in dependability assessment framework of this thesis the major components that constitutes an IoT application is the sensors, communication protocols and gateway devices. This result shows that sensor nodes and gateway has an impact in energy consumption and delay in an IoT application. There is an intention to investigate further through series of experiments to ascertain whether the communication protocol used in designing the application also has an impact in the energy consumption and the delay in the application. Therefore subsequent experiments are conducted below with the other communication protocols.

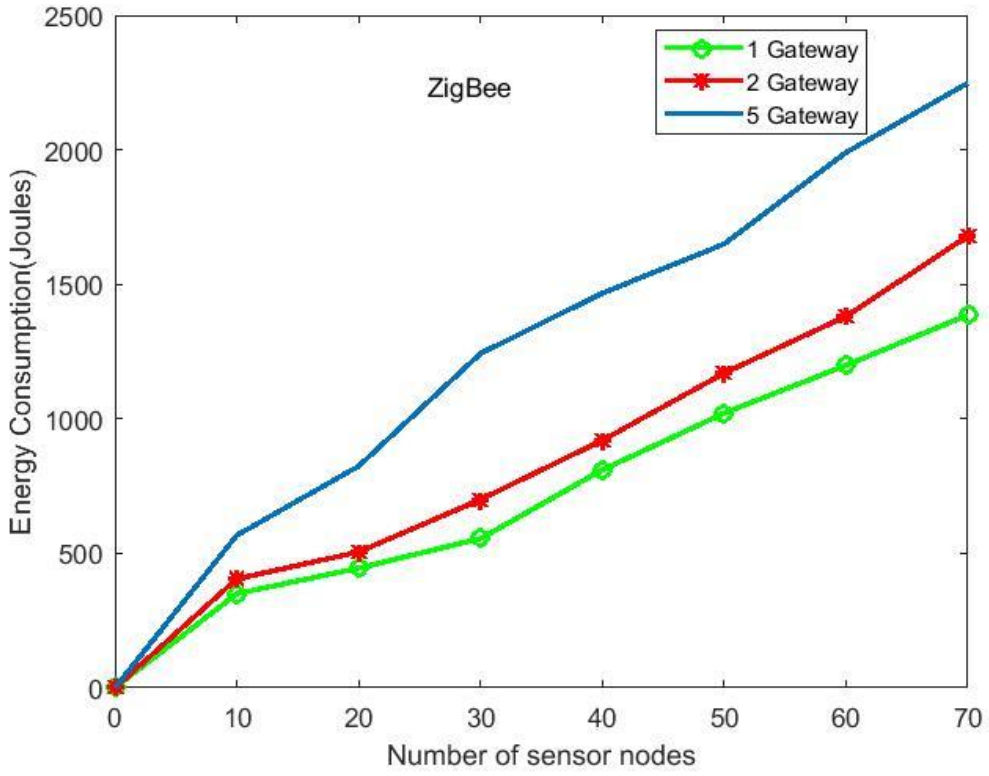


Figure 8.18: Experiment on energy consumption using ZigBee

The results on the experiment on energy consumption using ZigBee communication protocols as shown in (Fig 8.18) above, shows that using the same parameters of 10 sensors which are constant as the gateway varies, same as the experiment on energy consumption using Bluetooth as shown in (Fig 9.8), above. The results of the experiment on Zigbee indicates that for 10 sensor nodes to 1 gateway the level of energy consumption is 350 (J) slightly higher as compared to that of Bluetooth which was 330 (J). As the number of gateway devices progresses to 2 for the same amount of 10 sensor nodes, the energy consumed in the transmission of the packets to the gateway increases to 405 (J) and for Bluetooth on the same parameters uses lesser energy of 390 (J).

However, when the sensor nodes were increased to a fixed constant of 20 sensor nodes with a variation in the gateway devices. It was observed that for 20 sensor nodes to 1 gateway the energy consumption using ZigBee is 445 (J) which is slightly lower than that of Bluetooth which uses the same parameters of 20 sensor nodes to 1 gateway but the energy consumption is 480 (J). From the observation of the results it is clear that the addition of gateway devices to the sensor nodes actually consumes more energy during the transmission of packets, but this varies with the implementation of the particular communication protocol used in the

construction of the application, as an IoT application consist of three main components the sensor nodes, the communications protocol and the gateway devices.

From the result analysis of Bluetooth and ZigBee it shows that a communication protocol can be efficient in terms of energy consumption on a small-scale application, but when it gets to a large scale IoT application it could fail during the transmission of packets which could lead to the failure of the entire IoT application. Also, from the observation of the results on energy consumption it shows that as the more gateways are added to the sensor nodes there is a decrease in the transmission time in which the packets arrived the destination gateway. This shows a great impact that the addition of gateway devices to sensor nodes will reduce delay in an IoT application but could vary in terms of energy consumption as some communications protocols consumes less energy during packet transmission than others. An analysis of the experiment on delay using same parameter as the energy consumption is represented below in (Fig 8.19).

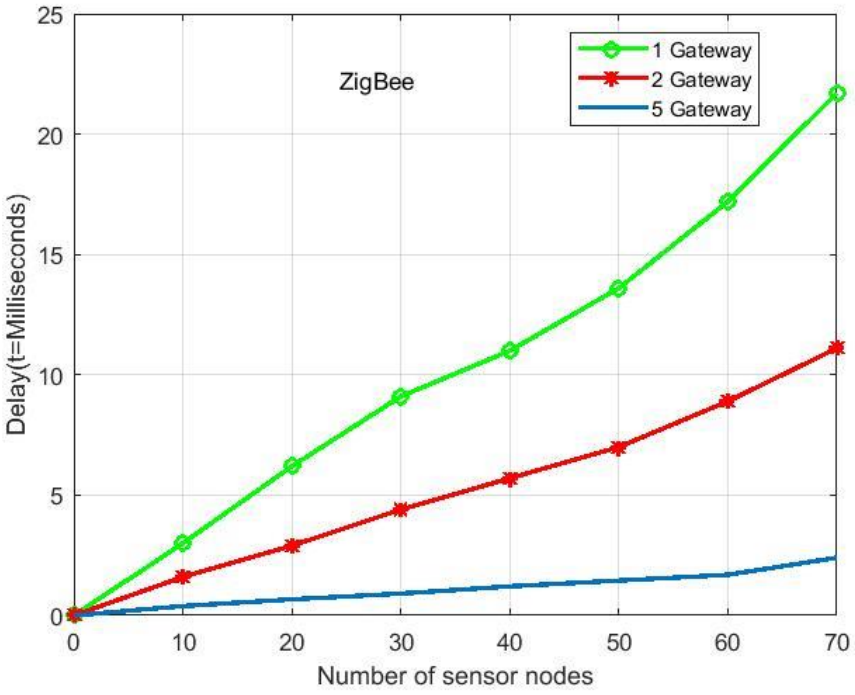


Figure 8.19: Experiment on delay using ZigBee

The results of the experiment on delay using ZigBee communication protocol, it is observed that for the level of delay in using 10 sensor node and 1 gateway is 2.8 milliseconds, when an additional 2 gateway is to added to the sensor node the level of delay experience was reduced 1.6 milliseconds, and when an additional 5 gateway is connected to the same 10 sensor node the level of delay reduces to 0.39 milliseconds this shows that as the gateway

increases, the level of delay in the transmission of the packets to the gateway reduces in time. As compared to the experiment on the energy consumption of ZigBee where the gateway increases the energy consumption increases.

Also from the observations in the results of (Fig 8.19), shows the variations in terms of the level of delay experienced by ZigBee protocol as compared to that of Bluetooth, it is noticed that as the sensor nodes progresses to 20 with 1 gateway, the level of delay in ZigBee is 5.4 milliseconds but for Bluetooth on the same parameters is 6.2 milliseconds and the energy used in transmitting the packets for ZigBee is 455 (J), while for Bluetooth is 480 (J). This indicate that various communications protocols has their level of efficiency and performance when it comes to delay and energy consumption in IoT.

When the number of gateway progresses from 2 to 5 gateways in the experiment conducted, it shows a lesser delay time during the communication of packets from the sensor nodes to the gateway, but a higher level of energy consumption as shown in (Fig 8.18). This shows that the lower the delay time in transmitting the packets to gateway, the higher the energy consumption in the application. The addition of more gateway devices increases the level of energy as reflected in the case of ZigBee and Bluetooth (Fig 8.16 and Fig 8.18). This need to be further explored through series of experimental study to ascertain the consequences of delay and energy consumption in an IoT application as the application becomes larger. This will be reflected upon in subsequent session of this analysis.

From the experiments conducted on delay and energy consumption of an IoT application on a small scale application it is certain that the addition of more gateways to the sensor nodes will reduce the delay in the application but increase the energy consumption as shown in (Fig 8.17 and Fig 19) when 2 gateways where added to the same constant sensor nodes of 20 the level of energy was increased as compared to when 1 gateway was used. This needs to be further explored in the subsequent experiment using the other communications protocols that often utilised (as stated in chapter 5 of this thesis) in the development of an IoT application to see their impact.

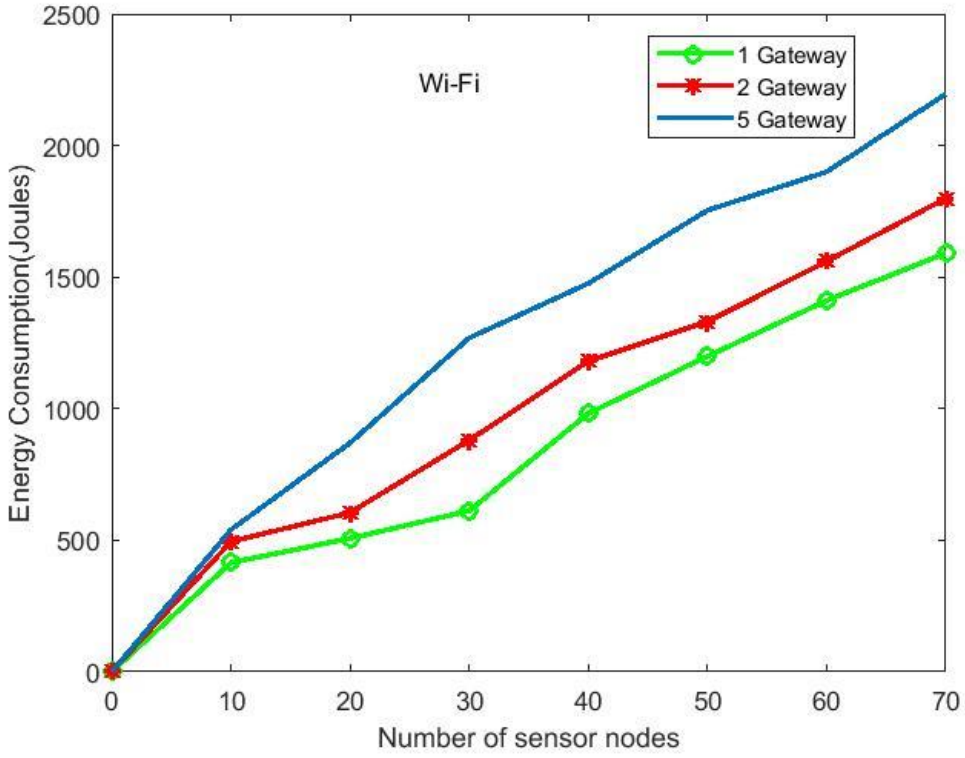


Figure 8. 20: Experiment on energy consumption using Wi-Fi

As shown in (Fig 8.20), the experiment conducted using Wi-Fi shows that for 10 sensor nodes to 1 gateway the level of energy consumption is 415 (J) and when the number of gateways increases to 2, the energy consumption was increased to 495 (J) and when an additional 5 gateways was added to the same sensor nodes (10) the energy consumption used in the transmission of the packets was 540 (J).

When comparing Wi-Fi to ZigBee and Bluetooth, it was discovered that Bluetooth and ZigBee uses a lesser amount of energy when the same number of devices are configured in a small scale scenario between 10 to 20 sensor nodes as compared to Wi-Fi but when the application scale increases in the number of sensor nodes and gateways it was discovered that a reverse was the case as Wi-Fi tend to consume less amount of energy of 1005 (J) which is lower than ZigBee and Wi-Fi. But when the sensor nodes was made 50 with the variations in the number of gateway devices, the results of the experiment clearly indicate that there was an increase in the energy consumption from 1110 to 1596 joules respectively.

The results of this experiment in line with the previous experiments conducted on ZigBee and Bluetooth clearly shows that as more gateway devices are added to the application there is an increase in the level of energy consumption in application but there was a variation in

the results as compared with the other communication protocols used in the previous experiment showing that the communication protocols also have an impact in the level of energy been used in the communication of packets to the gateway. However, from the experiment conducted so far it certain that the addition of more gateways to the sensor nodes will reduce the delay in transmission of the packets.

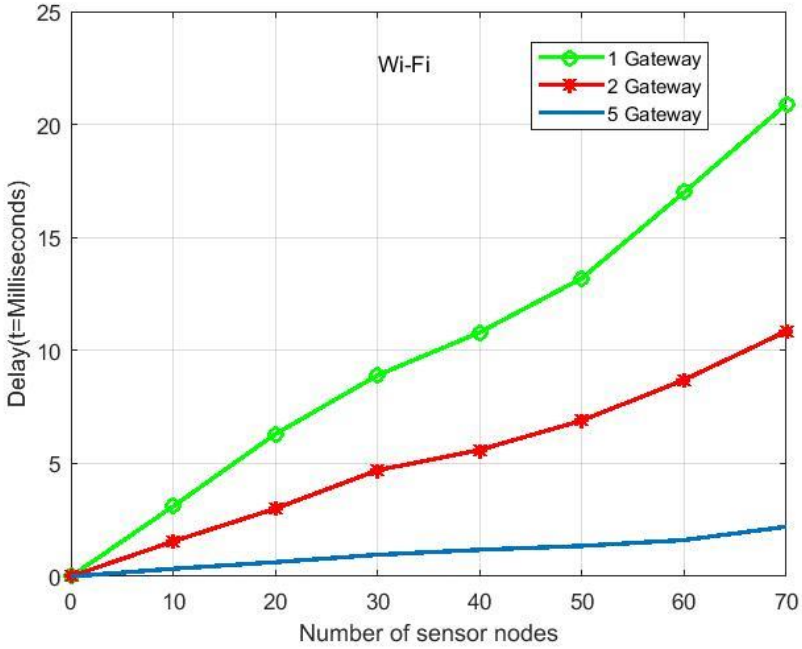


Figure 8.21: Experiment on delay using Wi-Fi

From the results of the experiment on delay using Wi-Fi communication protocol using 10 sensors nodes to 1 gateway the level of delay is 3.1 millisecond when an additional 2 gateway was added the level of delay dropped to 1.55 milliseconds and when it comes to 5 gateway the level of delay drops drastically 0.34 milliseconds. This shows that as the gateway increases, the level of delay in the transmission of the packets to the gateway reduces in time as shown in the (Fig 8.21) from the above experiment it is certain that the addition of gateway to the sensor nodes will reduce the delay in packet transmission time but when it comes to energy consumption the addition of gateways will increase the energy consumption.

The results of this experiements conducted so far shows that to improve the scalability property of an IoT application, the connectivity of the communication protocols between the senor node and the gateway has a major impact in the system operation and in the transmission of packets within the require time as information in the IoT environment is of

vital importance and the timely transmission of this information is very vital to ensure the reliability and dependability of the IoT system. WiFi communication protocols has got higher network connectivity in large scale IoT application when transmiting packets from the sensor nodes to the gateway with less receiving time as compared to bluetooth and ZigBee.

From the observation in the results it is certain that the energy consumption of Wi-Fi, when it comes to large scale of sensors nodes of over 50 devices in the application is lower than the energy consumption of Bluetooth and ZigBee. This shows that Wi-Fi is more efficient in regards to energy consumption of large-scale sensor nodes than Bluetooth and ZigBee. Wi-Fi has more reliability in the construction of large scale IoT application when it comes to efficiency in energy usage and the improvement in the delay time of the gateway receiver. However, further investigation into other communication protocols through series of experiments is needed to ascertain whether other communications protocols has more positive impact in minimising energy consumption and reducing the delay in time during packets transmission.

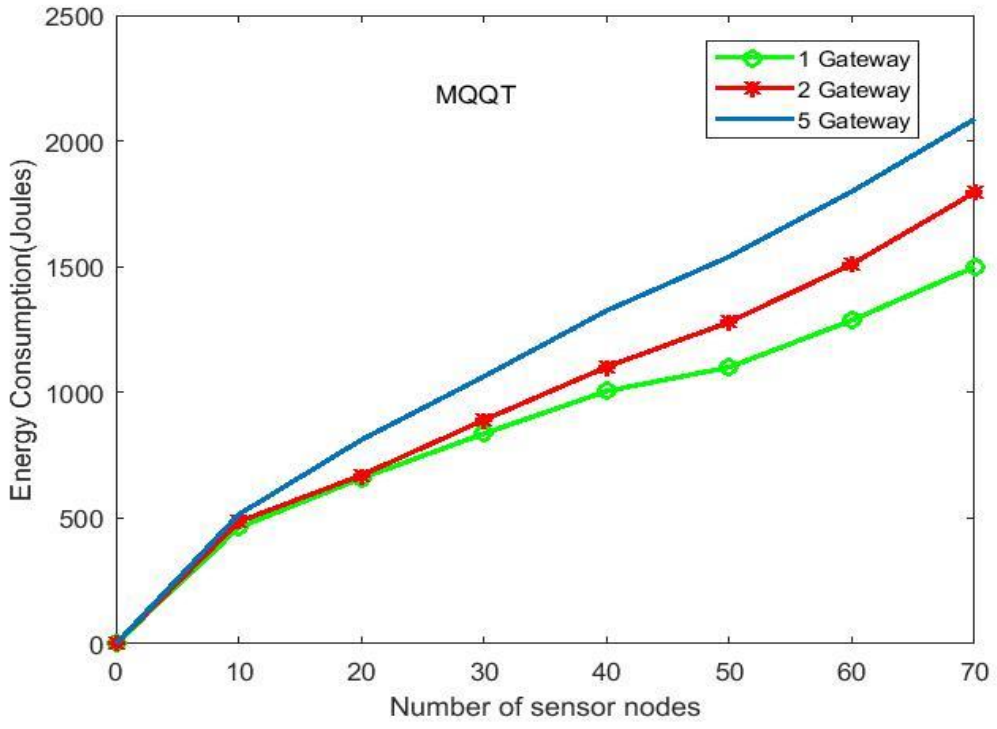


Figure 8.22: Experiment on energy consumption using MQTT

The results of (Fig 8.22), indicate that for 10 sensor nodes to 1 gateway the level of energy consumption is 462 (J), which is higher as compared to Bluetooth, ZigBee and Wi-Fi which

varies between 330 (J), 350 (J) and 415 (J) respectively. The results of the experiment on energy consumption using MQTT communication protocols as shown in (Fig 8.22) above, shows that using the same parameters of 10 sensors which is constant as the gateway devices varies between 2 and 5 as the experimental research progresses has different levels of energy consumption between 495 (J) and 540 (J) when using the same communication protocols (MQTT). However, as shown in the experiemental result in (Fig 8.16 and Fig 8.18) when it comes to small scale applications with less amount of sensor nodes and gateway devices ZigBee and Bluetooth is more effiecient in regards to the energy consumption of the application.

From the observation of the results, when the number of sensor nodes increases to 70 nodes to 5 gateways devices the energy consumption in the application was lower as compared to the experiment of ZigBee, Bluetooth and Wi-Fi. This experiement indicates that the communication protocols has a great impact in the scalability of an IoT application. Therefore for large scale IoT application with more sensor nodes and gateways. MQTT can more reliable in terms of energy consumption than bluetooth and ZigBee. As sensors nodes and gateway devices consumes energy during transmission a commuincation protocols with high throughput in wide range can be used to transmit the packets with less amount of time (delay) in large scale application which will reduce the delay and energy comsumption in the application.

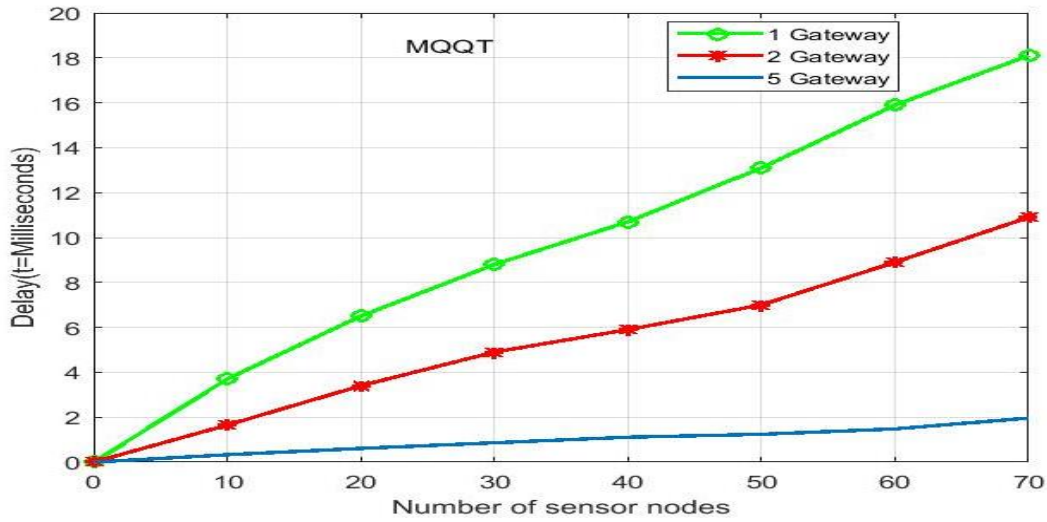


Figure 8.23: Experiment on delay using MQTT

The results in (Fig 8.23), on the experiment conducted on delay using MQTT communication protocol the level of delay experienced in the simulation experiment shows that for a total of 10 sensors to 1 gateway using MQTT, the transmission time for the packets to get to the gateway is 3.7 milliseconds, when an additional gateways were added to the same number of sensor nodes there was a lower level of delay between 1.65 and 0.33 milliseconds respectively. This shows that the number of gateways that are configured to the sensor nodes has an impact in the reliable transmission and delivery of packets. When compared to the previous experimental results of Bluetooth, Zigbee and Wi-Fi it was observed that the result of MQTT on the time in transmission of the packets to the gateway was slightly higher in regards to small scale IoT applications with less number of sensor nodes and gateway.

As the experimental scale progresses to a large-scale application consisting of 70 sensors the level of delay was reduced as compared to the experimental results using Bluetooth, Zigbee and Wi-Fi. This indicates that when it comes to energy consumption and delay of large scale IoT application. MQTT is a more reliable protocol than Zigbee, Bluetooth and Wi-Fi. Bluetooth and Zigbee are communication protocols specifically design for short range IoT application. However more research is intended to be performed into the comparison of MQTT and COAP communications protocols on their level of delay and energy consumption.

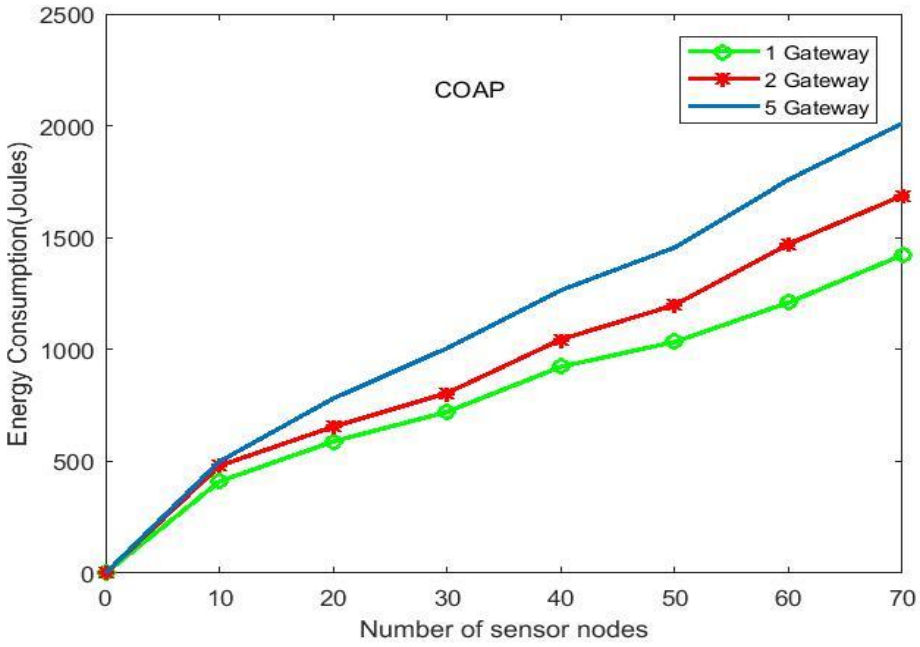


Figure 8.24: Experiment on energy consumption using COAP

As shown in figure (8.24), the experiment conducted using COAP shows that for 10 sensor nodes to 1 gateway the level of energy consumption consume during the transmission of packets is 410 (J), when an additional 2 gateway was added to the fix sensor node of the energy consumed is 480 (J) and when the number of gateways progresses to 5 gateway devices , the energy consumption was increased to 499(J) which is slighly lower than MQTT and Wi-Fi, when it comes to energy consumption of a small scale application.

As the experiment progresses to a large application with over 70 sensor nodes the level of energy consumption was 1422 (J) for 1 gateway, 1688 (J) for 2 gateway and 2011 (J) for 5 gateways which much reduced in term of energy consumption as compared to Bluetooth, with the same parameters is 1523 (J), 1800 (J) and 2298 (J) while the experiment results of ZigBee on the same scale of 70 sensor nodes is 1499 (J), 1799 (J) and 2248 (J) respectively.

From this experimental result it is certain that as the sensor nodes increases in an IoT application with the gateway devices. This has an impact on the energy consumption of the application as an addition of more devices will thus increase energy consumption but varies with various communications protocols. The observation of the results of (Fig 8.24) shows that when there is high number of sensor nodes and gateway devices in an IoT application the need for high data rate communications protocol like COAP and MQTT is needed for effective transmission of packets with less delay and reduce the energy consumption of the application.

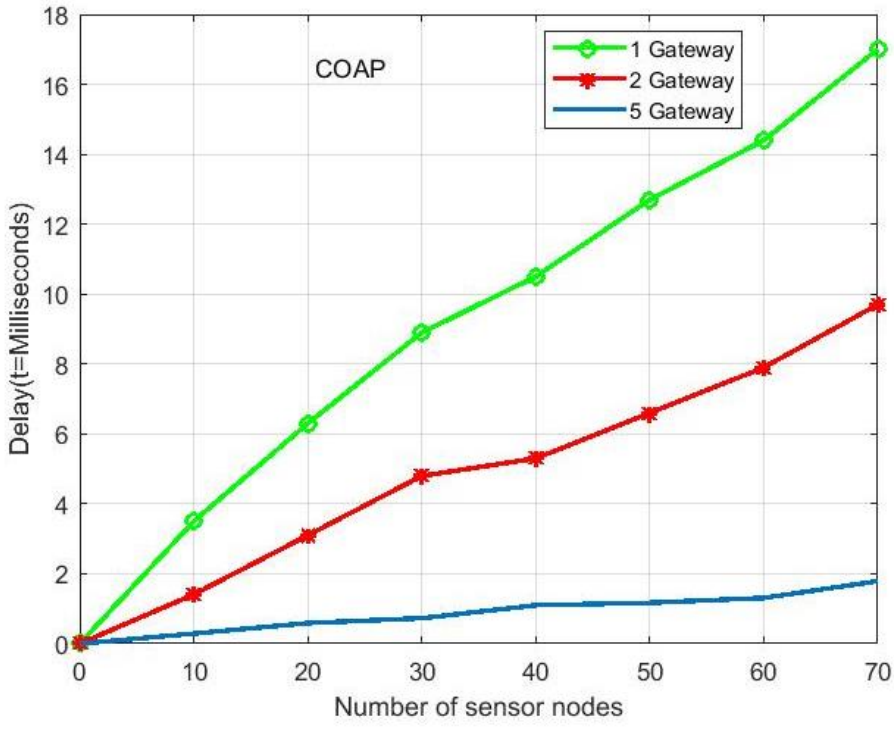


Figure 8.25: Experiment on delay using COAP

As showed in (Fig 8.25), on the experiment on delay using COAP, for 10 sensor nodes using 1 gateway is 3.5 milliseconds, which is higher than Bluetooth which took 2.7 milliseconds in sending the same amount of 1000 packets to the gateway device, when an addition of 2 gateway devices was added to the 10 sensor nodes the packets took a lower time of 1.4 milliseconds in delivery the packets to the gateway and when an additional 5 gateway devices was configured to the 10 sensor nodes the delay in time of receiving the packet in the gateway was reduced to 0.29 milliseconds this shows that the addition of subsequent gateway devices to the sensor nodes which thus reduce the delay time in the packet transmission an IoT application.

It is important to add more gateways to reduce delay in IoT application, but this increases energy consumption. But this varies with communications protocols used in designing the IoT application. As some communications protocol tend consumes more energy and has increased delay level of delay as compared to other communication protocols. From the results of the experiment it shows that COAP has a better throughput with less delay as compared to MQTT and Wifi with a lower level of delay as compare to ZigBee and Bluetooth. From the experimental results there is an indication of a great margin due to the fact that COAP and MQTT are specially design for wide area network which comprises of a large amount of devices in the application as compared to Zigbee and Bluetooth are mainly for routing in a short distance with less devices.

8.7 Analysis of the Findings

The results obtained from the experiment creates the basis of determining the provision of the dependability assessment framework in assessing the dependability of an IoT application. In the experimental scenarios, IoT applications were segregated in scales to consider the sizes of the application, as IoT application comes in various sizes. The small scale consists of application between 1 to 15 component, the medium-scale consist of 15 to 30 component values and the large scale consist of 30 and above component values in the application. The table below shows the summary of the findings.

Table 8.6 Analysis of the findings

IoT Components	Small scale application			Medium scale application			Large Scale application		
sensor	0-15	0-15	0-15	15-30	15-30	15-30	30-70	30-70	30-70
comms	ZigBee/ Bluetooth	Wi-Fi	MQTT / COAP	ZigBee/ Wi-Fi	Bluetooth	MQTT / COAP	MQTT / COAP	WiFi	ZigBee/ Bluetooth
Gateway	1	1	1	2	2	2	5	5	5
Ranking	Low	Medium	High	Low	Medium	High	Low	Medium	High

As shown above in table 8.6, the values of the sensor nodes, communication protocols and number of gateways are ranked in their level of severity. The experimental finding is express below.

```
if (sensor>0 and sensor<=10 and gateway==1):
then use bluetooth or ZigBee"

if (sensor>15 and sensor<=25 and gateway==2):
then use ZigBee or Wifi

if (sensor>30 and sensor<=70 and gateway>=5):
then use COAP or MQQT
```

Figure 8.26: Conditional expression of the provision of the dependability framework

The expression in (Fig 8.26) indicates that between 1 to 10 sensor nodes, to 1 gateway, the recommended communication protocol for the transmission of the packets in this IoT application is Bluetooth or ZigBee. When the number of sensor nodes and gateways in the application increases above 20 and not more than 25 sensor nodes the recommended gateway is 2, the communication protocol that is ideal for this kind of IoT network is either ZigBee or WiFi. Finally, when there is a large scale IoT application where the number of sensor nodes of above 30 and the number of gateways that is ideal for this application is 5, for large types of IoT applications the ideal communication protocol is COAP or MQTT.

8.8. Use Case Evaluation

This section presents an evaluation of the dependability assessment framework based on existing IoT applications. The framework is deployed to evaluate four IoT applications. The applications is selected for this evaluation are the smart healthcare for pathology detection by Amin et al (2019); IoT-based monitoring system for heart diseases patients by Li et al (2017); An application of IoT in weather monitoring and precision Farming by Nagesh et al (2017), and The use of IoT for Car Tracking System by Thomas & Rad (2017). This section demonstrates the how the dependability assessment framework can be deployed and used effectively in assessing IoT applications' dependability.

Case 1: Smart Healthcare for Pathology Detection

Amin et al (2019), designed an IoT healthcare application for detecting human body tissue and diagnosing their health status using interconnected sensors and gateways. The healthcare application includes remote tracking and monitoring of patients, intelligent disease detection, the smart pill dispensing, and remote medical equipment operation and control. This system will help in medical emergencies by providing an immediate response. It is connected with multiple smart healthcare sensors inside, on, and around the human body, receiving and monitoring real-time patents health status.

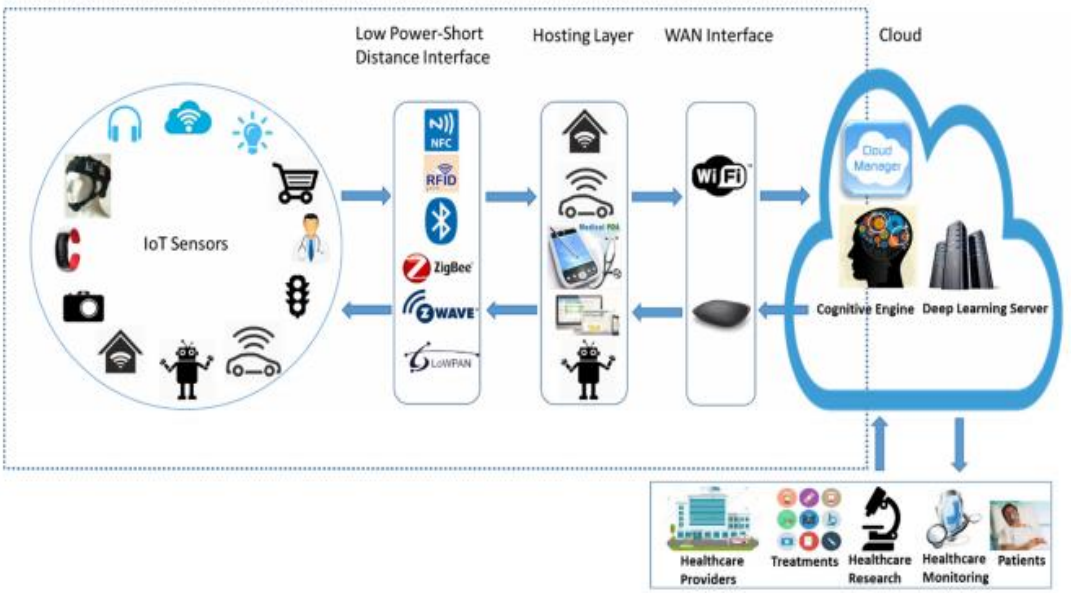


Figure 8.27: System architecture of Smart Healthcare for Pathology Detection (Source Amin et al, 2019)

The architecture of the smart healthcare system as shown in (Fig 8.27), indicates how the multimodal signal acquisition is carried out, through smart IoT sensors. Smart IoT sensors consist of wearable and fixed sensors that can measure medical signals, such as body temperature, heartbeat, blood pressure, voice, facial expressions, body movement, and EEG. Some of these sensors are embedded in the patient's surroundings. These devices can also communicate with other devices using IoT. The LAN consists of short-range communication protocols, such as Bluetooth, LoWPAN, and Zigbee. This layer transmits the acquired signals from the smart IoT sensor to another layer called the hosting layer (Amin et al, 2019).

The hosting layer has different types of smart devices, such as multimedia smartphones, laptops, tablets, and personal digital assistants. Which can store and send signals. These devices store data locally and have dedicated programs for simple computations on the received signals. The users can obtain preliminary and general health feedback using these limited processing devices. Data are transmitted to the cloud processing unit through the WAN layer. The smart devices are connected to the wide-area network (WAN), which sends the data received from the smart devices to the cloud unit. The WAN layer employs advanced communication networks, such as Wi-Fi, 4G, or 5G, to transmit data in real-time to the cloud. The cloud manager in the cloud layer authenticates the patient's data and sends them to the cognitive engine for processing (Amin et al, 2019).

- **System Analysis and Framework Application**

The IoT architecture designed by Amin et al (2019), shows the number of components that were used in the development of the smart healthcare application for pathology detection and monitoring. In analysing the application, the following factors were further considered as stated in the dependability assessment framework. In considering the application size, the total number of sensor nodes were assessed to indicate the scale of the application. The application is considered to be a small-scale application due to the number of sensor nodes of more than ten in values. The type of communication protocol used in the transmission of the data was also considered and finally the total number of gateway contained in the application.

The main area of consideration in the application of Amin et al (2019), was from the smart sensors to the hosting layer as this represents the characteristics of an IoT application. The acquisition of data from the physical world to the digital world (Aston, 2005). The impact, the potential occurrences and outcome of these components and their functions were considered during this analysis of the use case scenario. The outcome of this analysis is shown in the table below.

Table 8.7. Analysis of the IoT application of Amin et al (2019) with dependability assessment framework evaluation

Application Component	**Number of components**	**Severity (Framework Provision)**
Sensor	13 sensor nodes.	**Low**
Communication Protocols	Zigbee/ Bluetooth	**Low**
Gateway	4 smart gateways	**High**

The evaluation of the IoT application of Amin et al (2019) as presented above in table 8.7, indicate that with values of the components used in the design of the application, the application can be classed as a dependable IoT application when evaluated with the provisions of the dependability assessment framework. The application contains 13 sensor nodes with 4 smart gateways and uses ZigBee communication protocols for the transmission of the data.

The values of the sensor nodes in regards to the number of gateways in the application enable the scalability of the gateway to be able to receive the amount of data been transmitted. The addition of the multiple gateways is essential to the successful performance of an IoT network which involves large numbers of sensor nodes.

Case 2: IoT-based Monitoring System for Heart Diseases Patients

Li et al (2017), designs an IoT application for monitoring heart disease. This application can monitor the health status of the patients such as blood pressure, ECG, SpO2 and send the acquired data to the remote physician. This ensures the physician is aware of the patient's heart condition in real-time. The aim of this monitoring system is to assist remote practitioners to be aware about patients' health status and to diagnose or forecast dangerous conditions, satisfying the requirement of medical diagnose of heart diseases. The IoT system in this application consist of the data acquisition part which is referred to as the sensing layer and the data transmission part. The application layer in the system is used by the physician in checking the patient's health status as shown in figure 8.28 below.

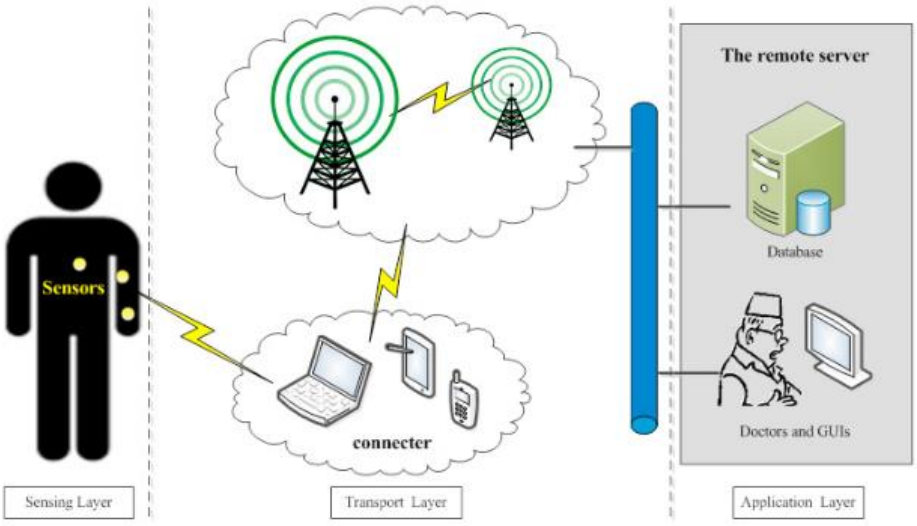


Figure 8.28: The system architecture of IoT-based monitoring system for heart disease patient (Source Li et al 2017)

The system architecture of Li et al (2017), contains the sensing layer which is composed of the sensors nodes placed on the patient's body for monitoring the patients' health status. The data transmission process in the application was divided into two sub-processes. In the first sub-process, the data is sent from the sensors to a connector using a short-range communication protocol. Bluetooth communication protocol is used to communicate the

data from the sensor nodes to the connector gateway. The connector gateway sends the data to the remote side. The connector gateway used in the application of Li et al (2017), comprises of a smartphone application, a personal digital assistant (PDA) and a laptop device.

The connector gateway has the computing abilities suitable gateway for receiving data from the sensor nodes and for remotely transferring the data through other communication technology that is suitable for long-distance communication. The coverage range of the communication protocol is a factor that is critically considered in the design of the application of Li et al (2017). The communication quality is an important element in the design of an IoT application. In comparing the requirements and the characteristics of the communication protocols. Li et al (2017), used Bluetooth communication protocol in the first sub-process in acquiring the data from the sensor nodes in short range distance and cellular technologies like GSM and GPRS were utilised in the second sub-process for the transmission of the data to the remote physician within the wide area network of the application.

- **System analysis and framework application**

A critical analysis of the IoT application of Li et al (2017) conducted, indicate the components that were used in the design of the IoT application for heart disease patients. During the analyses of this application, the number of the sensor nodes used in the design of the application is considered, the communication protocols used in the application is also a point of consideration and the number of gateway devices in the application was also assessed to see if it meets the criteria and the provisions stated of the dependability assessment framework.

The application of Li et al (2017) is more of an IoT application that has to do with the patient and the remote physician. The primary aim of the design of this application is to monitor the patient's vital health status to administer the necessary assistance. This application was further classified by the author to have two sub-processes in the operation. However, the first sub-process which has to do with the sensor nodes to the connector gateway is the IoT application network infrastructure in the architecture as presented above.

The communication between the gateway connector and the remote physician is more of the data processing and intelligence monitoring. An IoT application is the connectivity between the sensor nodes and the connector gateway as the physical parameters can be accessed at this point either for development, feedback or processing. Hence, the point of evaluation in

this application with the provisions of the dependability assessment framework as stated earlier in will be on the first sub-processes from the sensor nodes to the connector gateway.

In this evaluation, the components and their functions are assessed with dependability assessment framework. All these parameters are evaluated with the provisions of the dependability assessment framework. Table 8.8 below is a representation of the outcome of the analysis.

Table 8.8. Analysis of the IoT application of Li et al (2017) with dependability assessment framework evaluation

Application Component	**Number of components**	**Severity (Framework Provision)**
Sensor	3 sensor nodes.	**Low**
Communication Protocols	Bluetooth	**Low**
Gateway	3 smart gateways	**High**

As shown above, in table 8.8, the application Li et al (2017) can be classed as a dependable IoT application as it meets the requirements of the provision of the dependability assessment framework. The values of sensors in this application is 3 sensor nodes, using Bluetooth communication to this point the application can be classed as dependable in regards to less delay but assessing the number of gateway devices used in the design of the application will thus increase the energy consumption in the application.

Case 3: Application of IoT in weather monitoring and precision Farming

Nagesh et al (2017) developed a smart farming application with remote sensors by which the agrarian harvests will be checked continuously. The sensor nodes in the application of Nagesh et al (2017) monitors the humidity, temperature, moisture and irrigation of the crops. These sensor data are used for directing and encompassing agronomic varieties to explore new conceivable answers for more prominent yield.

The goal of this application to provide long term sustainable solution for automation of agriculture. This IoT based solution to obtain continuous knowledge from the crops and also to develop and deploy smart technology for agriculture sector which benefits in improving

agricultural and environmental sustainability, promoting increased crop productivity, crop traceability and safety of the agricultural yield.

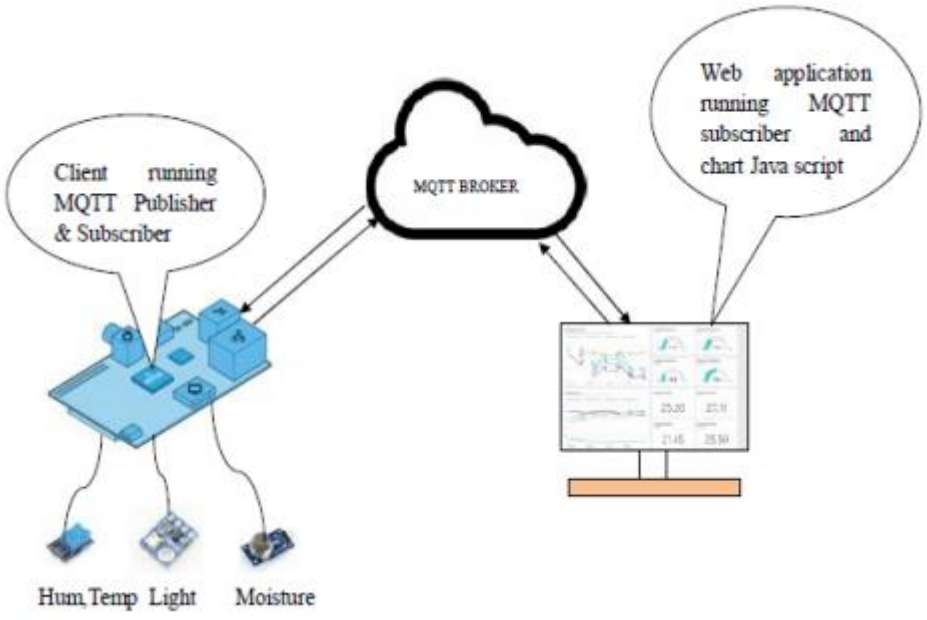


Figure 8.29: Architectural model of weather monitoring and precision farming (Source: Nagesh et al 2017)

The architectural model of Nagesh et al (2017) shows the components used in the design of the system. The sensors are installed in a single raspberry Pi computer board with interfaced for the digital pins. The sensors are configured in the raspberry pi board with a python script language. The Raspberry-Pi configuration board sends the sensor data to a central server over publish/subscribe messaging protocol MQTT. The focal server is designed within the sensor nodes to facilitate the network configuration and routing. The role is to send multicast information to the MQTT server. MQTT server is called a broker and the clients are simply the connected sensors and gateway in the application. The sensor data is then sent to web application layer which contains an SQL server where the information is analysed and processed.

- **System analysis and framework application**

In the analysis of the above application the following main parameters where considered the number of the sensor nodes contained in the application, the communications protocol used in the transmission of the agricultural data from the sensor nodes and the number of gateways contained in the application. From the analysis of the components that make up the IoT

application of Nagesh et al (2017), it is established that the system comprises of three sensor nodes, MQTT communication protocols and one broker gateway.

After, a detailed understanding of the components and functions of the IoT application of Nagesh et al (2017). The provisions of the dependability assessment framework is applied in the evaluation of the components used in the design of this application to ascertain if the application meets the criteria set out in the framework. The components used in the design of an IoT application has a great impact on the dependability of the operations of an IoT application. The output of the analysis of this application is shown below in table 8.9.

Table 8.9. Analysis of the IoT application of Nagesh et al (2017) with dependability assessment framework evaluation

Application Component	**Number of components**	**Severity (Framework Provision)**
Sensor	3 sensor nodes.	**Low**
Communication Protocols	MQTT	**High**
Gateway	1 broker (sink)	**Low**

- **Results Analysis**

From the results and analysis conducted, the IoT applications evaluated can be classed as a dependable IoT application as it meets the provisions of the dependability assessment framework. This use case evaluation has demonstrated the use of the framework in the evaluation of an existing IoT application to see if it is fit for purpose.

Case 4 The use of internet of things in Car Tracking System

The IoT technology promises a broad range of exciting products and services, car tracking technology as part of the broad range of technological concept under the IoT paradigm. The car tracking technology involves deploying some basic IoT components into the tracking of important transportation component; the basic principle behind any technological concept involves delivery of high quality product that conforms to specifications (Thomas & Rad

2017).Considering the importance of this technology, any failure resulting from operations in critical scenario can be detrimental

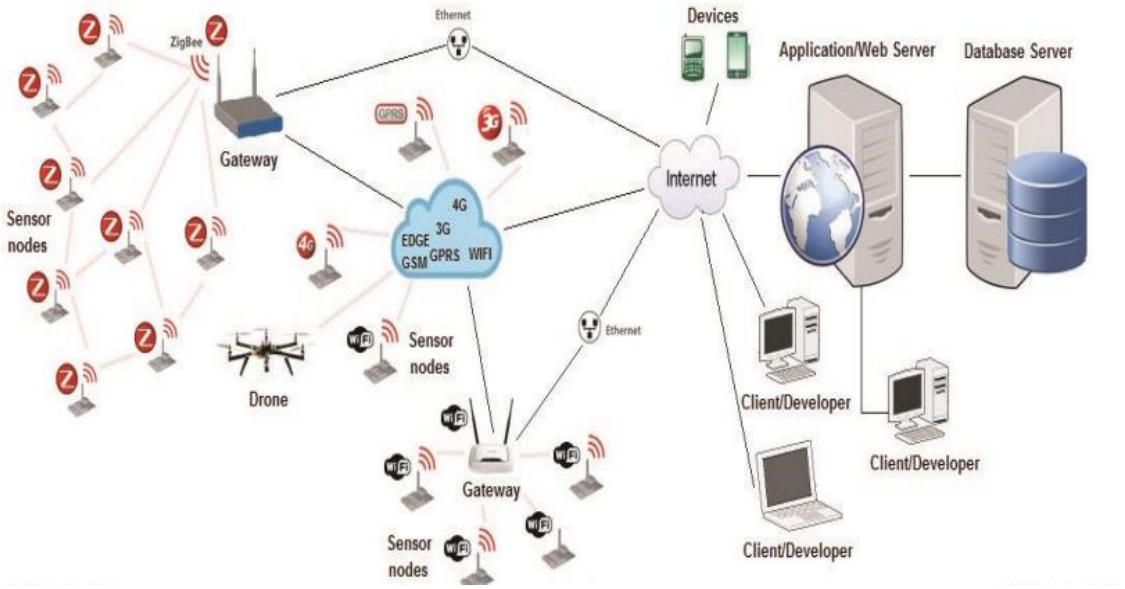


Figure 8.30: The use of IoT for Car Tracking System (Source: Thomas & Rad 2017)

As shown in (Fig 8.30), the basic fundamental concept of the IoT revolves around object identification, tracking, locating and also a proper network management platform through which data is dispersed to sensing objects through Bluetooth communication protocol. This process links the object the location composes of the time, longitude, latitude, and altitude. All of these data is necessitated in tracking and controlling a car in real-time. The intelligent process creates the enhance data sharing. It is paramount for the technological components which constitute the car tracking technology and the network infrastructure to be highly dependable to ensure that the packets transmitted through the network are delivered on a timely basis without any interruptions, delay or failures (Thomas & Rad 2017).

- **System analysis and framework application**

After a critical analysis of the IoT application of Thomas & Rad (2017), the components identified in the application was a total of 17 sensor node with one gateway using Zigbee communication protocol from the transmission of the data. The communication protocols used in the application is also a point of consideration and the number of gateway devices in the application was also assessed to see if it meets the criteria and the provisions stated of the dependability assessment framework.

Table 9. Analysis of the IoT application of Thomas & Rad (2017) with dependability assessment framework evaluation

Application Component	Number of components	Severity (Framework Provision)
Sensor	17 sensor nodes.	**Low**
Communication Protocols	Zigbee	**Low**
Gateway	1	**Low**

- **Results analysis**

The results of the analysis of the IoT application of Thomas & Rad (2017) shows that this application can be class as a dependable IoT application considering the components used in the design of the application in line with the criterial of the dependability assessment framework. The application of this use case evaluation has showed the effective usage of the dependability assessment framework in the assessment of the dependability of an IoT application.

8.9. Summary

The dependability assessment framework was evaluated on the two critical areas of application. The first phase of the evaluation, was on the development of an IoT application. This was achieved through simulation and experiments, the results of the finding of this evaluation indicates a high knowledge of the components dependability during the system operation. The results of the experiments also created a level certainty in the selection of the right component when developing an IoT application. The second phase, of the framework evaluation, is on existing IoT application, the findings of this approach shows that the dependability assessment framework can be effectively used in assessing the dependability of an existing IoT application.

9. Conclusion

9.1. Research Achievements

This research is focused on the enhancement of dependability in IoT driven application. As IoT is becoming an integral part of human life's, the dependability of this application is a challenging issue that needs to be addressed in the development of IoT and successful achievement of the services provided by this concept. This research study is able to develop a dependability assessment framework for assessing the dependability of IoT application. The achievements of the research questions in this study is shown in table 9.1.

Table 9.1. Achievement of Research Questions

Research Questions	**Achievement**
What constitutes dependability of IoT applications?	The constituents of dependability in IoT was ascertained through this research study.
How can the dependability be assessed and ensured in IoT applications?	A dependability assessment framework was constructed to assess the impact of the components used in the design of IoT applications.

The main aim of this research study is to develop a framework that can used for assessing the dependability in IoT application. In achieving the aim of this thesis, a detailed literature review on the dependability in computer systems was conducted to create an understanding of dependability in IoT. An analysis of the components that make up an IoT application was conducted. These components were evaluated using the fault tree analysis method to assess the dependability of each of these components during the system operation. This evaluation method of the components creates a certainty that the three major components in an IoT application needs to be assessed in creating a dependable system. Table 9.2 below resents the achievements of this research study.

Table 9.2. Achievements of the Aims and Objectives of this research study

Research Aim	Achievement
Aim:	Develop a dependability assessment framework for IoT application.
Research Objective:	**Fulfilment**
Objective 1: To conduct an in-depth investigation of literature to understand the concept of dependability in relation to IoT applications.	An in-depth literature review was conducted to understand the concept of dependability in relation to IoT application in achieving of this objective, the key dependability requirements of an IoT was identified.
Objective 2: To critically analyse the characteristics and the functional requirement of existing IoT applications**.**	A critical analysis was conducted on existing IoT applications to assess the components that are used in the design of the application. This process created an understanding of the structures and systematic functional constituents of an IoT application. The key characteristics of an IoT application were identified in the analysis leading to the classification of small, medium and large scale type of IoT application in regards to the number of component in the application.
Objective 3: To critically analyse the failures in the components of an IoT application.	The fault tree analysis method was used in analysing the components failures in an IoT application. This process created the

	understanding of the importance of the component and the root cause of the failures in the components that make up an IoT application. The criticality of the failures in the components to the operation of an IoT application was ascertained during this process.
Objective 4: To develop of a dependability assessment framework for IoT applications.	In achieving this objective, three layers was considered in the design of the framework, a the first layer consists of the categorisation of IoT application into sizes the values that make up the classification was put into consideration. The second layer of consideration in the framework, consists of the components that make up an IoT application. In assessing the dependability of an IoT application, there is need for a variation of the components used in the design of the application in regards to their relative performance. The third layer of the framework is based on the assessment of the application in line with the established dependability requirement.
Objective 5: To evaluate the framework for its effectiveness and viability.	The dependability assessment framework is deployed by both the system analysist and the system developer to test for its effectiveness and viability. This process was done to ensure the practicability of the framework in assessing real time IoT application. In achieving this objective two major techniques was applied, the use of

	simulation environment and the use case evaluation method.

The developed framework provides a practical way that can be adapted by a developer in building a dependable IoT application in assessing the dependability requirements of the components in IoT application in order to enhance the systems operation. The dependability assessment creates the requirement for the effective operation of the IoT system. This framework also enables system analysts to initiate the evaluation and assessment of an IoT application using the processes stated in the framework. This result and finding of this assessment can used for comparing the best components and whether a specific component is dependable and if it fits the purpose for the design of the IoT application. The evaluation result highlights the capability of the components.

Overall, this thesis presents a framework that offers the processes that enable both the developer and system analyst to assess the specific dependability requirement of the components of an IoT application and making critical decisions in choosing the right components in designing an IoT application. Therefore, making sure that the system is dependable and operates within the specified time and providing a service that can justifiably trusted. The original contribution to knowledge of this research study are resented in table 9.3.

Table 9.3. Original Contribution to Knowledge

Research contribution 1: A detailed understanding of dependability in IoT, with clear measurable attributes

Achievement:

This research study was able to create an understanding of dependability with the clear measurable attributes from other related areas of computer science which can be applied to IoT applications. This include a concise understanding of the factors that needs to be addressed when considering dependability in an IoT application.

The dependability of an IoT application can therefore be defined as the ability of the application to provide a service that can be justifiably trusted. The main attributes of dependability in IoT is reliability and availability, where availability is the readiness of correct service and reliability is the continuity of service. The availability of an IoT application is the ability of the system to deliver the required service within the required time. A failure in a component of an IoT application will adversely affect the service delivery of the application.

The concept of availability is directly related to the concept of reliability, the reliability of an IoT application is paramount to its successful operation, this can be reflected in the timely packet transmission and delivery and in energy efficiency. In IoT, sensors cooperatively sense, collect and process information in the monitoring environment or area, there is need for real time acquisition and timely processing of information. Reliability in data transmission is a key determinant of the dependability in an IoT application.

However, there is a limited understanding of dependability in the energy efficiency in IoT. This research study was able to create an understanding of processes that contribute to the reliability in energy efficiency in IoT. An increase in IoT devices will potentially produce a large amount of data that will be transmitted through the communication network. Therefore reducing the power demand of these devices through effective and reliable transmission network in processing the data as quickly as possible from the sensory devices will adversely improve the reliability of the application.

Research Contribution 2: A framework for assessing dependability in IoT application

Achievement:

The main contribution of this research study is the development dependability assessment framework for IoT application assessing dependability of an IoT application. The objective of this framework is to provide practical ways to assess the dependability of an IoT application. The intended purpose of the design structures and processes contained in the framework is to enable system developers and analysist to assess the dependability of an IoT application during the design stage and before deployment of existing IoT applications in ensuring effective service delivery. The two main the targets that are considered to be the users of this framework are the system developer and the system analysist. However, is it envisaged, that this framework can be used by also the end-users in assessing the dependability of an IoT application.

This framework consists of the processes and structures that creates a logical understanding of the processes involved in assessing the dependability of an IoT application. In assessing the dependability of an IoT application the size of the application is a factor to be considered. IoT applications are complex systems with variations in the number of components used in the design, an assessment of the application components will create an understanding of the application's operation. Identifying the variations in an IoT application is considered as an important factor in the ensuring the dependability of an IoT application. The dependability requirements in the framework are essential characteristics, required in the effective service delivery.

9.2. Research Limitations

9.2.1. Lack of existing research materials specific to the concept of IoT

The area of IoT is relatively new and lacks a huge amount of research materials as compared to other areas of research in computer science. This has a great impact on the progress of this research study, as most available related research is more focus on wireless sensor network and distributed systems. Another limitation in this research study is finding the right simulation tool that is compatible with the components of IoT application was a challenge.

9.2.2. Lack of Simulation Tools

The lack of simulation tool in the area of IoT limits the configuration used in the development of the simulation environment despite numerous attempts was made in finding an IoT specific simulator, the existing WSN simulators was utilised due to its usage in the application of IoT by other researchers in the literature.

9.3. Recommendations and Future Work

9.3.1. A quantitative Analysis of IoT Application Components using the Fault Tree Approach:

The quantitative analysis of IoT application using the fault tree is recommended to get the probability of the severity of the failures of the components. In quantitative analysis, the probability of the occurrence of the top event and other quantitative reliability indexes such as the important measures are mathematically calculated, given the failure rate of the individual system components. The results of the quantitative analysis give an indication about the system reliability and also help in determine which components failures are more critical so as to place more consideration during the system development. However, this is mostly relevant when using a fault tree in assessing the components of a system during the developmental stage.

9.3.2. A continuous Evaluation of the Dependability assessment Framework

A continuous evaluation of the dependability assessment framework for IoT application is recommended as the concept of IoT expands. This will ensure the robustness of the framework to suit the dependability requirements of the diverse range of IoT application as the concept continues to develop.

References

Abohamama S A, Alrahmawy F M & Elsoud M A (2017) Improving the dependability of cloud environment for hosting real time applications. Ain Shams Engineering Journal.

Adesina O A , Agbele K K & Nyongesa O H (2011) Using Wearable sensors for Remote Healthcare Monitoring System. Journal of Sensor Technology 2011.

Alfian G, Syafrudin M, Ijaz F M, Syaekhoni A M, Fitriyani L N & Rhee J (2018) A Personalized Healthcare Monitoring System for Diabetic Patients by Utilizing BLE-Based Sensors and Real-Time Data Processing. Journal of Sensors.

Al-Taee A M, Al-Nuai,Y W, Al-Ataby A, Muhsin J Z & Abood N S (2015) Mobile Health Platform for Diabetes Management Based on the Internet-of-Things. IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)

Angelo D G, Ferretti S & Ghini V (2016) Simulation of the Internet of Things. Proceedings of the IEEE 2016 International Conference on High Performance Computing and Simulation (HPCS 2016)

Angelo D G, Ferretti S & Ghini V (2017) Modelling the Internet of Things: a simulation perspective. Proceedings of the IEEE 2017 International Conference on High Performance Computing and Simulation (HPCS 2017)

Angelo D G, Ferretti S & Ghini V (2017) Multi-level simulation of Internet of Things on smart territories. Simulation Modelling Practice and Theory.

Angelo G, Ferretti S & Ghini V (2016), Simulation of the internet of Things.

Ashton K (2009) That ’internet of things’ thing in the real world, things matter more than ideas. RFID Journal,

Atkins (2013) A Cloud Service for End-User Participation Concerning the Internet of Things. In Signal-Image Technology & Internet-Based Systems (SITIS), International Conference on. IEEE.

Atzori L, Iera A, & Morabito G, (2010) The internet of things: A survey, Comput. Netw., vol. 54, no. 15, pp. 2787–2805.

Aublin, P L, Mokhtar B S & Quéma V (2013) RBFT: Redundant Byzantine Fault Tolerance. 33rd IEEE International Conference on Distributed Computing Systems. International Conference on Distributed Computing Systems

Avizienis A, Laprie J.C , Randell B & Landwehr C. (2004) Basic Concepts and Taxonomy of Dependable and Secure Computing, IEEE Trans. on Dependable and Secure Computing, vol. 1, no. 1, pp. 11-33

Avizienis A, Laprie C J & Randell B (2001) Fundamental Concepts of Computer System Dependability. IARP/IEEE-RAS Workshop on Robot Dependability: Technological Challenge of Dependable Robots in Human Environments – Seoul, Korea,

Ayestaran I, Agirre I, Nicolas F C & Perez J (2014) Simulated Fault Injection for the Validation of Fault Tolerance Mechanisms in Dependable Time-Triggered Systems

Bader A, Ghazzai H, Kadri A & Alouini S M (2016) Front-End Intelligence for Large-Scale Application-Oriented Internet-of-Things. 6 IEEE. Translations and content mining.

Balaji P, Buntinas D, Kimpe D (2012) Scalable computing and communications: theory and practice. In: Zomaya Y, Khan SU Wang L (eds) Fault tolerance techniques for scalable computing. John Wiley & Sons Publishing, Hoboken, New Jersey

Balandina E, Balandin S, Balandina E, Koucheryavy Y, Balandin S & Mouromtsev D (2015) IoT Use Cases in Healthcare and Tourism. IEEE 17th Conference on Business Informatics

Bandyopadhyay D & Sen J (2011) Internet of things: Applications and challenges in technology and standardization," Wireless Pers. Commun., vol. 58, no.

Bauer E & Adams R (2012) Reliability and availability of cloud computing. John Wiley & Sons. 2012.

Beaulah S P & Chelliah P.R. (2017) Networking Topologies and Communication Technologies for the IoT Era. In: Mahmood Z. (eds) Connected Environments for the Internet of Things. Computer Communications and Networks. Springer, Cham

Bernaschi M, Cacace F, Pescape A, & Za S. (2013) Analysis and experimentation over heterogeneous wireless networks. In Tridentcom. IEEE, 2005.

Boano A C, Romer K, Tsiftes N , Voigt T , Zuniga M, Langendoen K, Brown J, Roedig U, Veiga A, Socorro R, Vilajosana X & Monton M, (2015) Research by Experimentation for dependability on the Internet of Things. Seventh Framework Programme.

Bondavalli A, Clin D M, Latella D, Majzik I, Patariza A & Savola G (2001) Dependability analysis in the early phases of UML-based System design. International journal of computer systems science & Engineering

Botta A, Donato D W, Persico & Pescape A (2014) On the integration of cloud computing and Internet of Things Internatinal conference on future Internet of things and cloud

Botta, A. Pescape & Ventre G (2008) Quality of service statistics over heterogeneous networks: Analysis and applications. European Journal of Operational Research.

Bouguera T, Diouris J, Chaillout J, Jaouadi R & Andrieux G (2018) Energy Consumption Model for Sensor Nodes Basedon LoRa and LoRaWAN

Brambilla G, Picone M, Cirani S Amoretti M & Zanichelli F (2014) A simulation platform for large-scale internet of things scenarios in urban environments. Proceedings of the First International Conference on IoT in Urban Space

Cavallo F, Aquilano M & Arvati M (2014) An Ambient Assisted Living Approach in Designing Domiciliary Services Combined with innovative Technologies for patients with Alzheimer's Disease: A Case Study. American Journal of Alzheimer's Disease and other Dementias

Chao C H (2011) Internet of things and cloud computing for future internet. In Ubiquitous Intelligence and Computing.

Chen H, Jia X & Li Heng (2011) A brief introduction to IoT gateway

Cinque M and Curran K, (2013) On dependability issues in ambient intelligence systems. Pervasive and ubiquitous technology. Innovations for Ambient Intelligence. Ulster, UK. 2013.

Clement A, Wong E, Alvisi L, Dahlin M & Marchetti M (2009) Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults. Symposium on Networked Systems Design and Implementation.

Coetzee L & Eksteen J (2011) the internet of things- promise for the future? An Introduction. International Information management Corporation. South Africa

Copie A, Fortis F T, & Munteanu I V (2013) Benchmarking Cloud Databases for the Requirements of the Internet of Things. In 34th International Conference on Information Technology Interfaces.

Cvijikj P I & Michahelles F (2011) The Toolkit Approach for End-user Participation in the Internet of Things. In Architecting the Internet of Things. Springer.

Deng J, Han R & Mishra S (2006) Secure code distribution in dynamically programmable wireless sensor networks in Proc. ACM/IEEE Int. Conf Inf. Process. Sens. Netw. (IPSN) pp. 292–300.

Dobre C & Xhafa F (2013) Intelligent services for Big Data science. Future Generation Computer Systems.

Dohr A, Modre-Osprian R, Drobics M, Hayn D, & Schreier G (2010) The Internet of Things for Ambient Assisted Living. International Conference on Information Technology.

Dong Z, Wang Z, Xie O, Emelumadu W, Lin W & Rojas-Cessa R (2015) An experimental study of small world network model for wireless networks. in Proceedings of the 36th IEEE Sarnoff Symposium (Sarnoff ’15)

Driscoll, K., Hall, B., Sivencrona, H., & Zumsteg, P. (2003) Byzantine Fault Tolerance, from Theory to Reality. *SAFECOMP*.

Egwutuoha IP (2014) A proactive fault tolerance framework for high performance computing (HPC) systems in the cloud. PhD thesis, University of Sydney

Elghazel W, Bahi J, Guyeux C, Medjaher K & Zerhouni N (2015) Dependability of wireless sensor networks for industrial prognostics and health management

Elhayatmy G, Dey N & Ashour S A (2017) Internet of Things Based Wireless Body Area Network in Healthcare

Elkhodr M, Shahrestani S & Cheung H (2013) The Internet of Things: Vision & Challenges .School of Computing Engineering and Mathematics. University of Western Sydney. Sydney, Australia

Enokido T, Yan L, Xiao B, Kim D, Dai Y & Yang T L (2005) Embedded and Ubiquitous Computing EUC 2005 Workshops. Springer

Escobar M J J, Matamoros M O, Espinosa Q H, Padilla T R & Gutiérrez R G A (2019) Optimizing a Centralized Control Topology of an IoT Network Based on Hilbert Space. Internet of Things (IoT) for Automated and Smart Applications

European Commission (2008) Internet of things in 2020 road map for the future. Working Group RFID of the ETP EPOSS,

Fernandez F & Pallis C G (2014) Opportunities and Challenges of the Internet of Things for healthcare. International Conference on Wireless Mobile Communications and HealthCare.

Flocchini P & Gasieniec L (2006) Structural Information and Communication Complexity: 13th International Colloquium. SIROCCO

Fruhwirth T, Krammer L & Kastner W (2015) Dependability demands and state of arts in the Internet of Things. Emerging Technologies & Factory Automation (ETFA), IEEE Conference. Austria .

Gazis V, Goertz M, Huber M, Leonardi A, Mathioudakis K, Wiesmaier A & Zeiger F (2015) Short Paper: IoT:Challenges, Projects, Architectures. 18th International Conference on Intelligence in Next Generation Networks

Gerkey P B & Mataric J. M (2002) A market-based formulation of sensor-actuator network coordination, in: Proc. of the AAAI Spring Symposium on Intelligent Embedded and Distributed Systems, CA, pp. 21–26

Gershenfeld N (1999) When things Start to think. Henry Holt and Co. USA

Gia N T, Rahmani M A, Dhaou B I & Westerlund T (2017) An IoT based continuous glucose monitoring system: A Feasibility study. 8th International Conference on Ambient Systems, Networks and Technologies (ANT-2017)

Gu Y, Wu CQ, Liu X, Yu D (2013) Distributed throughput optimization for large-scale scientific workflows under fault-tolerance constraint. J Grid Comput 11(3):361–379

Gubbi J, Buyya R, Marusic S & Palaniswami M (2013) Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems.

Gui N, Sun H, Florio V D, and Blondia C (nd) A Service-oriented Infrastructure Approach for Mutual Assistance Communities . Interdisciplinary institute for BroadBand technology, Ghent-Ledeberg, Belgium.

Guimaraes FP, C'elestin P, Batista DM, Rodrigues GN, de Melo ACMA, (2013) A framework for adaptive fault-tolerant execution of workflows in the grid: empirical and theoretical analysis. J Grid Comput

Guinard D (2009) Towards the web of things: Web mashups for embedded devices. ACM

Guo B, Zhang D, and Wang Z (nd) living with Internet of Things: The Emergence Of Embedded Intelligence.

Hadzilacos V (1985) Issues of fault tolerance in concurrent computations (databases, reliability, transactions, agreement protocols, distributed computing). Ph.D. Dissertation. Harvard University, USA. Order Number: AAI8520209.

Haenggi M (2002) Mobile sensor-actuator networks: opportunities and challenges. Cellular Neural Networks and their Applications IEEE Int. Germany pp. 283–290

Haider S & Nazir B (2016) Fault tolerance in computational grids: perspectives, challenges, and issues. SpringerPlus

Hank P, Muller S, Vermesan O & Van Den J (2013) Automotive ¨ ethernet: in-vehicle networking and smart mobility. In Proceedings of the Conference on Design, Automation and Test in Europe, pages 1735–1739. EDA Consortium.

He T, Stankovic J, Lu T, & Abdelzaher T (2003) A real-time routing protocol for sensor networks, in: Proc. IEEE Int. Conf. on Distributed Computing Systems (ICDCS) USA, pp. 46–55

Henkel J, Pagani S, Amrouch H, Bauer L & Samie F (2017) Ultra-Low Power and Dependability for IoT Devices. Design, Automation & Test in Europe Conference & Exhibition

Horst S, Strese H, Loroff C, Hull J, and Schmidt S. (2006) Europe Is Facing a Demographic Change Ambient Assisted Living Offers Solutions. Berlin

Hossain M M, Fatouhi M & Hasan R (2015) Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things. IEEE World Congress. New York.

Hou W (2013) Integrated reliability and availability analysis of networks with software failures and hardware failures. Industrial and Management Systems Engineering, South Florida.

Hu F, Xie D & Shen (2013) On the application of the internet of things in the field of medical and health care. School of information Engineering. IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing.

Hua-Dong M (2011) Internet of things: Objectives and Scientific Challenges. Journal of computer science and technology.

Issarny V & Zarras A (2003) Software Architecture and Dependability. In: Bernardo M., Inverardi P. (eds) Formal Methods for Software Architectures.

Jamil. Y. Khan & Mehmet R. Yuce (2010) Wireless Body Area Network (WBAN) for Medical Applications, New Developments in Biomedical Engineering, Domenico Campolo.

Jiang S (2015) Internet of Things (loT): technologies and applications. International Conference on Advances in ICT for Emerging Regions (ICTer)

Joshi R k, Bunker G, Jahanian F, Moorsel V A & Weinman J (2009) Dependability in the Cloud: Challenges and Opportunities. Dependable Systems & Networks

Kabanda G (1994) A stochastic model of dependability of computer systems in Zimbabwe. Transactions on Information and Communications Technologies.

Kaul L & Goudar I I R (2016) Internet of things and Big Data – challenges. International Conference on Green Engineering and Technologies (IC-GET)

Kecskemeti G,Casale G, Jha N D, Lyon J & Ranjan R (2017) Modelling and simulation challenges In Internet of things. IEEE Cloud computing

Kempf J, Arkko J, Beheshti N & Yedavali K (2011) Thoughts on reliability in the Internet of Things. Journal of Interconnecting smart objects with the Internet workshop, vol.1, pp.1-4 Prague.

Kharchenko , V, Lian A K & Andrzej R (2019) Dependable IoT for Human and Industry: Modeling, Architecting, Implementation. Rivers Publications.

Kharchenko , V, Lian A K & Andrzej R (2019) Dependable IoT for Human and Industry: Modeling, Architecting, Implementation. Rivers Publications.

Korkalainen M, Sallinen M, Karkkainen N & Tukeva P (2009) Survey of Wireless Sensor Networks Simulation Tools for Demanding Applications. Fifth International Conference on Networking and Services.

Kranenburg V & Bassi A (2012) IoT Challenges. Communications in Mobile Computing.

Kranenburg V R (2007) The Internet of Things: A Critique of Ambient Technology and the All-Seeing Network of RFID. Amsterdam, The Netherlands: Institute of Network Cultures, 2007.

Kranenburg V R, Anzelmo E, Bassi A, Caprio D, Dodson S & Ratto M (2011) The internet of things in Proc. 1st Berlin Symp. Internet Soc., Berlin, Germany, 2011

Lamport L (2016). The Byzantine Generals Problem. ACM Transactions on Programming Languages and Systems. SRI International.

Lamport L, Shostak R & Pease M (1982) The Byzantine generals problem. ACM Trans. Program. Lang. Syst.

Laprie J D (2004) Dependable Computing: Concepts, Challenges, Directions

Laprie C J (1995) Dependability of Computer Systems: Concepts, Limits, Improvements.

Li X, Lu R, Liang X, & Shen S X, Chen J, Zhejiang L & Lin X(2011) Smart Community: An Internet of Things Application IEEE Communications Magazine

Lin J, Chelliah P.R, Hsu M & Hou J (2019) Efficient Fault-Tolerant Routing in IoT Wireless Sensor Networks Based on Bipartite-Flow Graph Modeling," in *IEEE Access*, vol. 7, pp.

Liu Y, Tong K, Qiu X, Liu Y & Ding X (2017) Wireless Mesh Networks in IoT Networks. A new soultion for IoT Networks

Liu, C H, Yang B & Liu T (2014) Efficient naming, addressing and profile services in Internet of Thing sensory environments. Ad Hoc Networks.

Liu, J., Chen, M., Yang, T., & Wu, J. (2018) IoT Hierarchical Topology Strategy and Intelligentize Evaluation System of Diesel Engine in Complexity Environment. Sensors (Basel, Switzerland),

Macedo D & Silva I (2014) A dependability evaluation for Internet of Things incorporating redundancy aspects.

Macedo D & Silva I (2014) A dependability evaluation for Internet of Things incorporating redundancy aspects. Networking, Sensing and Control (ICNSC). IEEE. Miami pp 417-422

Maier M, Montreal Q C & Levesque M, (2014) Dependable fiber-wireless(FiWi) access networks and their role in a sustainable third industrial revolution economy. IEEE Reliability society.

Masi D G (2018) The impact of topology on Internet of Things: A multidisciplinary review. Advances in Science and Engineering Technology International Conferences (ASET), *Abu Dhabi,*

Meissner S, (2012) Internet of Things – Architecture. IoT-A Project Deliverable D2.5 - Adaptive, fault tolerant orchestration of distributed IoT service interactions.

Mesbahi, M.R., Rahmani, A.M. & Hosseinzadeh, M (2018) Reliability and high availability in cloud computing environments: a reference roadmap. *Hum. Cent. Comput. Inf. Sci.* **8,** 20.

Michel R (1997) Real-Time Dependable Decisions in Timed Asynchronous Distributed Systems. Proceedings of the Third Workshop on Object-Oriented Real-Time Dependable Systems. IEEE.

Moon Y-H, Youn C-H (2015) Multi hybrid job scheduling for fault-tolerant distributed computing in policy-constrained resource networks. Comput Netw 82:81–95

Motlagh H N, Bagaa M & Taleb T (2017) UAV-Based IoT Platform: A Crowd Surveillance Use Case. IEEE Communications.

Mukhopadhyay S C & Suryadevara (2014) Internet of Things: Challenges and Opportunities. Internet of Things. Smart Sensors, Measurement and Instrumentation, vol 9. Springer, Cham

Nazir B, Qureshi K, Manuel P (2012) Replication based fault tolerant job scheduling strategy for economy driven grid. J Supercomput 62

Nordrum A (2016) Popular internet of things forecast of 50 billion devices by 2020 is outdated. IEEE SPECTRUM Of computer science and Engineering, University of Bologna, Bologna , Italy.

O'Regan G (2006) Mathematical Approaches to Software Quality (Springer Verlag, London, 2006)

O'Regan G (2017) Software Reliability and Dependability. Springer International Publishing

Ouerhani N, Pazos N, Aeberli M & Muller M (2016) IoT-Based Dynamic Street Light Control for Smart Cities Use Cases

Pan Y & Hu N (2014) Research on Dependability of Cloud Computing Systems. International Conference on Reliability, Maintainability and Safety (ICRMS)

Panda S (2017) Security Issues and Challenges in Internet of Things. Silicon Institude of technology.India

Partha P R (2016) Towards Internet of Things Based Society. International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)-2016

Perera C, Zaslavsky A, Christen P & Georgakopoulos D (2014). Context Aware Computing for The Internet of Things: A Survey. IEEE Communications Surveys & Tutorials Volume: 16, Issue: 1

Podofillini L, Sudret B, Stojadinovic B, Zio E & Kroger W (2015) Safety and Reliability of complex Engineered systems. Taylor & Francis Group London

Power A & Kotonya G (2019) Providing Fault Tolerance via Complex Event Processing and Machine Learning for IoT Systems. In 9th International Conference on the Internet of Things

Pradeebha R , Srinidhi R , Vanitha D M &, P.Veeralakshmi M.E (2018) Remote Monitoring Of Patient's Healthcare Using IoT And Android Application. International journal of recent trends in engineering & research.

Quan B C (2014) Cloud Service System Reliability Modeling. Electronic Product Reliability and Environment Testing Vol.32 No.2.

Rabaiei K & Harous S (2016) Internet of Things: Applications and Challenges. 12th International Conference on Innovations in Information Technology (IIT).

Rodrigues N G, Alves V, Silveira R & Laranjera A L (2012) Dependability analysis in the Ambient Assisted Living Domain: An exploratory case study

Rodriques G N, Alves V, Franklin R & Laranjeira L (2010) Dependability Analysis in the Ambient Assisted Living Domain": An Exploratory Case Study.

Sarkat C, Nambi U A, Prasad V R, Rahim A, Neisse R & Baldini G (2015) DIAT: A Scalable Distributed Architecture for IoT. IEEE Internet of Things Journal.

Shakir M (2002) The selection of case studies: Strategies and their applications to IS implementation cases studies. Institute of Information and Mathematical Sciences, Massey University, Albany Campus

Sheng Z, Yang S, Yu Y, Vasilakos V A, McCann A J & Leung K K (2016) A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities.

Silva I, Leandro R , Macedo D & Guedes A L (2013) A dependability evaluation tool for the Internet of Things. Computers and Electrical Engineering. Elsevier. Bazil.

Slater R (1998) Distributed Dependability. Dependable Embedded Systems.
Sommerville I (2011) Software Engineering, 9th edn.

Stankovic A J (2014) Research Directions for the Internet of Things". IEEE, Vol 1.

Stapelberg F R (2009) Handbook of reliability, availability maintainability and safety in Engineering Design. Centre for Infrastructure and Engineering Management Griffith University Gold Coast Campus Queenslans Australia.

Stefan K V, Otto P & Alexandrina M P (2017) Considerations Regarding the Dependability of Internet of Things. 14th International Conference on Engineering of Modern Electric Systems (EMES)

Taherkordi A & Vitenberg R (2014) Dependability in Distributed Systems. INF 5360 spring

Ta-Shma P, Akbar A, Gerson-Golan G, Hadash G, Carrez F, & Moessner K (2017) An Ingestion and Analytics Architecture for IoT applied to Smart City Use Cases. IEEE Internet of Things Journal.

Thakate S, Patil A & Siddiqui A (2016) The Internet of Things – Emerging Technologies, Challenges and Applications International Journal of Computer Applications.

Thomas O M & Rad B B (2017) Reliability Evaluation Metrics for Internet of Things, Car Tracking System: A Review.I.J. Information Technology and Computer Science

Tokuda H, Takuro Y & Nakazawa J (2014) Monitoring Dependability of City-scale IoT Using D-Case. IEEE World Forum on Internet of Things (WF-IoT)

Trinka M & Cerny T (2016) Identity Management of Devices in Internet of Things Environment. Dept. of Computer Science FEE, Czech Technical University

Ullah, Z., Ahmed, I., & Razzaq, K. (2017) DSCB: Dual sink approach using clustering in body area network

Vermesan O & Friess P (2014) Internet of things- From Research and Innovation to Market Deployment .

Volochiy B, Yakovyna V & Mulyak O (2017) Queueing Networks for Availability and Safety Assessment of the IoT Data Service. Computer Sciences and Information Technologies (CSIT), 2017 12th International Scientific and Technical Conference

Wang L, Jie W, Chen J (2018) Grid Computing: Infrastructure, Service, and Applications

Warriach U E, Ozcelebi T & Lukkien J J (2014) A First Step Towards a Dependability Framework for Smart Environment Applications, ADAPTIVE. Abidoye P A, Azeez A N,

Warriach U E, Ozcelebi T, & Lukkien J J (2012) A Machine Learning Approach for Identifying and Classifying Faults in Wireless Sensor Networks. IEEE 15th International Conference on Computational Science and Engineering

Warriach U E, Ozcelebi T, & Lukkien J J (2014) Fault-Prevention in Smart Environments for Dependable Applications. IEEE Eighth International conference on self adaptive and self organizing systems

Warriach U E, Ozcelebi T, & Lukkien J J (2015) Proactive Dependability framework for smart Environment Applications.

Woo W M, Lee W J & Park H K (2018) A reliable IoT system for personal healthcare devices. Future Generation Computer Systems 78

Xu D L, He W & Li S (2014) Internet of Things in Industries: A Survey. IEEE transactions on industrial informatics, vol. 10, no. 4

Xu X, Chen T & Minami M (2012) intelligent fault prediction system based on internet of things. Computers & Mathematics with Applications

Yan Z, Zhang P, & Vasilakos V A (2014) A survey on trust management for Internet of things. Journal of network and Computer

Yang M & Liu C (2016) Research of SNMP-based network topology discovery algorithm in IOT. Proceedings of the 2016 4th International Conference on Machinery, Materials and Computing Technology

Zaslavsky A, Christen P & Georgakopoulos D, (2014) Context Aware Computing for the Internet of Things: A Survey. IEEE Communications surveys & Tutorial vol 2.

Zhang D, Zhu Y, Zhao C & Dai W (2012) A new constructing approach for a weighted topology of wireless sensor networks based on local-world theory for the Internet of Things (IOT). Computers and Mathematics with Applications. Elsevier.

Zhao F (2010) Sensors meet the cloud: Planetary-scale distributed sensing and decision making. In Cognitive Informatics (ICCI), 9th IEEE International Conference.

Zhou S, Lin K, Na J, Chuang C & Shih C (2015) Supporting Service Adaptation in Fault Tolerant Internet of Things," IEEE 8th International Conference on Service-Oriented Computing and Applications (SOCA*)*, Rome

Appendix- A. Case Analysis

Case 1 – Remote healthcare monitoring system

In the research of Abidoye et al (2011), sensor nodes are integrated into the human body through the construction of WBAN technology called Medical Super Sensor (MSS). This sensor has more memory, processing and communication capabilities than other sensor nodes. MSS uses a radio frequency to communicate with other body sensors and ZigBee is used as a communication protocol to communicate with the Personal Server (Abidoye et al, 2011).

Abidoye et al (2011), considered Bluetooth and ZigBee technologies. Bluetooth supports maximum of seven active slaves as a model of communication (i.e. sensors to be controlled by one master, personal server). The second technology is ZigBee/IEEE 802.15.4 standard. It has a short range, low power consumption, low cost technology, capable of handling large sensor networks up to 65,000 nodes and reliable data transfer. It supports a maximum of 250kbps using Industrial, Scientific and Medical (ISM) free band i.e. 2.4 GHz. Therefore, they adopted ZigBee for transmitting physiological signals from WBAN to the patient server. Their reason for adopting this method was for security issues, scalability and Interoperability issues between the devices and their communication protocols (Abidoye et al 2011).

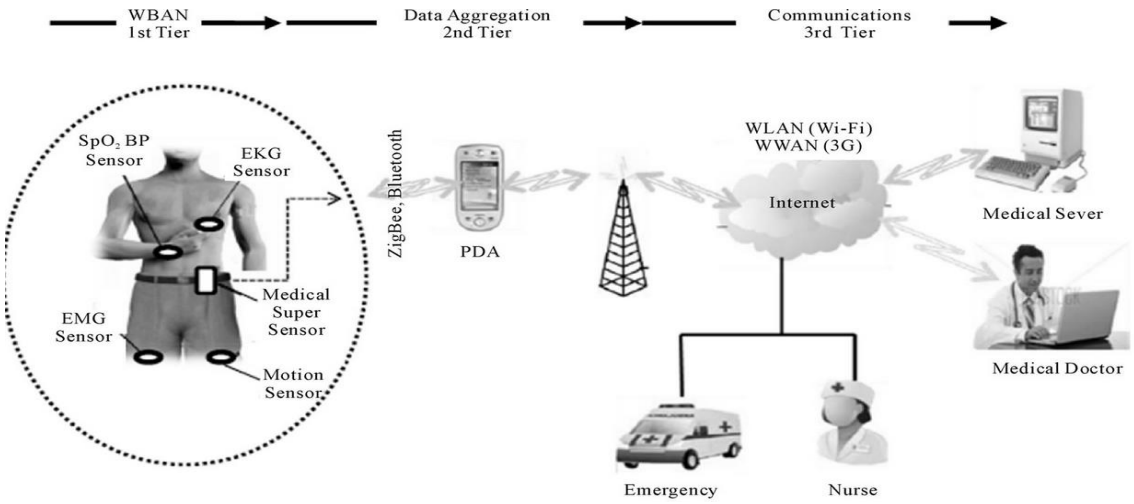


Figure 1: Architecture of wearable sensors for remote healthcare monitoring systems (Source: Abidoye et al 2011).

The method not only reduces transmission delay of physiological vital signs but also improves its bandwidth utilization. The role of wireless technology in healthcare

applications is expected to become more important with an increase in deployment of mobile devices and wireless networks (Abidoye et al 2011).

Case 2 - Internet of Things in Smart Cities

The research of Cenedese et al (2014), shows that the IoT can access a variety of devices such as home appliances, surveillance cameras, monitoring sensors, actuators, with a common de-facto standard for internet communications, is as HTTP, IPv4/v6, and Ethernet. Which are the sensor nodes and IoT components, such as the Constrained Application Protocol (CoAP), IPv6, and 6LoWPAN (Cenedese et al 2014).

The IoT nodes are equipped with a CC2420 transceiver, that implements the IEEE 802.15.4 standard. Routing functionalities are provided by the IPv6 Routing Protocol for Low power and Lossy Networks (RPL). Nodes collectively deliver their data to the gateway, which represents the single point of contact for the external nodes. The gateway hence plays the role of 6LoWPAN border router and RPL root node.

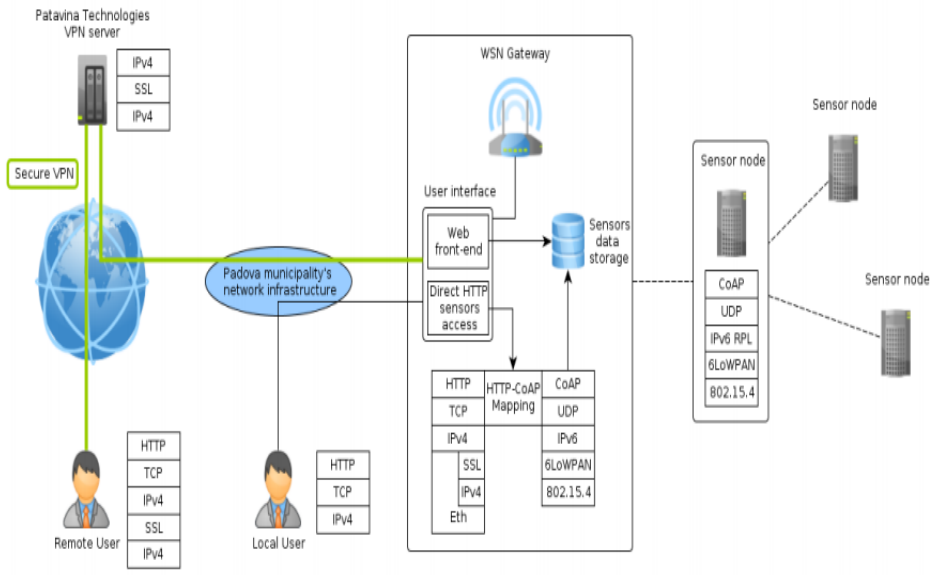


Figure 2: Padova Smart City (Source: Cenedese et al 2014).

This approach makes it possible to develop IoT services that can easily interact with other web services through the adoption the Representational State Transfer (ReST) paradigm. This paradigm, indeed, guarantees strong similarities in the structure of IoT and traditional web services, thus promoting the adoption of IoT by both end users and service developers. The web service approach requires the deployment of suitable protocol layers in the different

elements of the network, as shown in the protocol stacks depicted in the figure above shows the key elements of the system architecture.

Appendix - B. Overview of the Experimental Tool

The OMNeT++, is a C++ based discrete event simulator that is primarily design for modelling communication networks, multiprocessors, distributed and parallel systems (Varga &Hornig 2008). OMNeT++ is an open source tool publicly available under the academic license and a free software used as a non-commercial product for research purposes (Varga 2001 & Korkalainen et al, 2009). The motivation behind the development of OMNeT++ is to produce a powerful open-source discrete event simulation tool that can be used by academics, educational and research-oriented commercial institutions for the simulation of computer networks, distributed and parallel systems. OMNeT++ creates an avenue for an open-source, research-oriented simulation software such as NS-3, NS-2 and other commercial alternative simulation tool like OPNET (Bajaj et al 2000 & Varga 2001).

1.1.The Design of OMNeT++

OMNeT++ is objectively designed to support network simulation of various scales which include small, medium and large-scale experimental scenarios. The sub-sections below are the following main design requirements of OMNeT++:

● The design of OMNeT++ enables large-scale simulation as the models are hierarchical and built from reusable components.

● The simulation software enables visualisation and debugging of the simulation models which reduces the debugging time, which is usually a large percentage of simulation projects.

● The software is modular, customisable and allows embedding simulations into larger applications such as a network planning which brings additional requirements on the memory.

● OMNeT++, has facilities for integrated development environment that facilitates the development of models and effective analysis of the results.

The following sections below, are important aspects of OMNeT++ simulation highlighting the design decisions that aid in the achievement of the goal of this research study.

1.2. Model Structure

The models in OMNeT++ consists of modules that communicate messages. The active modules or simple modules are written in C++, in the simulation class library (Varga & Hornig 2008). The simple modules are grouped into compound modules and the number of hierarchy levels are not limited. Data and messages are sent through the connections that span between modules or directly to their destination modules. The concept of simple and compound modules is similar to DEVS (Chow & Zeigler 1994, & Zeigler 1990) atomic and coupled models. Both simple and compound modules are instances of module types. While describing the model, the user defines module types. The instances of these module types serve as the component for constructing the complex module types.

The modules normally contain various parameters which serves the purpose of passing the configuration data to the simple modules, in defining the network topology model. The various parameters used are string, numeric or Boolean values. Because parameters are represented as objects in the program. The parameters also holding the constants which serves as the source of the random numbers in the model configuration, which interactively prompt the user for the value and for holding expressions that are referencing other parameters. As well as the compound modules passing the expressions of parameters to their submodules (Gimenez et al 2013 & Abo-Zahhad et al 2014).

1.3. The Design of the NED Language

The user of the simulation creates definitions of the descriptions and structures of the model in NED, which contains the modules and their interconnections in OMNeT++'s network topology (Varga 2001). The typical ingredients of a NED description are the simple module declarations, the compound module definitions and the network definitions. The simple module declarations describe the interface of the modules the gates and the various parameters used in the design of the NED. The compound module consists of the declaration of the module's external interface (gates and parameters), and the definition of submodules and their interconnection. A definition in the network creates the compound modules in the self-contained simulation models (Korkalainen et al, 2009).

The NED language is designed to enable large scale deployment, the recent growth in the amount and complexity of OMNeT++-based simulation models and model frameworks made it necessary to improve the NED language as well. In addition to a number of smaller improvements, the following major features have been introduced:

Inheritance: Modules and channels in the NED, are designed in subclasses. Derived modules and channels can be improved with new parameters, gates, and (in the case of compound modules) new submodules and connections. The existing parameters have a specific value, and also the gate size is set in a gate vector. This makes it possible, to take a GenericTCPClientApp module and derive an FTPApp from it by setting certain parameters to a fixed value; or derive a WebClientHost compound module from a BaseHost compound module by adding a WebClientApp submodule and connecting it to the inherited TCP submodule (Gimenez et al 2013 & Abo-Zahhad et al 2014).

Interfaces: Module and channel interfaces are used as a placeholder where normally a module or channel type are stored, and the concrete module or channel type is determined at network setup through the setting of the parameters. Concrete module types are implemented in the interface. The module types ConstSpeedMobility and RandomWayPointMobility are implemented in the IMobility and plugged into a MobileHost, that contains an IMobility submodule (Gimenez et al 2013 & Abo-Zahhad et al 2014).

Metadata annotations: It is possible to annotate module or channel types, parameters, gates and submodules by adding properties. Metadata are not used by the simulation kernel directly, but they can carry extra information for various tools, the runtime environment, or even for other modules in the model (Gimenez et al 2013 & Abo-Zahhad et al 2014). The module's graphical representation (icon, etc) or the prompt string and unit (milliwatt, etc) of a parameter are specified using these properties. The NED language has an equivalent XML representation, that is, NED files can be converted to XML and back without loss of data and comments (Varga 2001).

1.4. OMNeT++ Graphical Editor

The OMNeT++ package includes an Integrated Development Environment (IDE), which contains a graphical editor using NED as its native file format; moreover, the editor can work with arbitrary, even hand-written NED code. The editor is a fully two-way tool, i.e. the user can edit the network topology either graphically or in NED source view, and switch between the two views at any time (Varga & Hornig 2008). This is made possible by design decisions about the NED language itself. First, NED is a declarative language, and as such, it does not use an imperative programming language for defining the internal structure of a compound

module. Allowing arbitrary programming constructs would make it practically infeasible to write two-way graphical editors which could work directly with both generated and hand-made NED files (Gimenez et al 2013 & Abo-Zahhad et al 2014).

Most graphical editors only allow the creation of fixed topologies. However, NED contains declarative constructs (resembling loops and conditionals in imperative languages), which enable parametric topologies. it is possible to create common regular topologies such as ring, grid, star, tree, hypercube, or random interconnection whose parameters (size, etc.) are passed in numeric-valued parameters. The potential of parametric topologies and associated design patterns have been investigated in (Varga & Fakhamzadeh 1997). With parametric topologies, NED holds an advantage in many simulation scenarios both over OPNET where only fixed model topologies can be designed, and over NS-2 where building model topology is programmed in the Tcl and often intermixed with the simulation logic, this makes it generally impossible to write graphical editors which could work with existing, hand-written code (Bagrodia et al 2008).

1.5. Separation of Model and Experiments

It is always a good practice to try to separate the different aspects of a simulation as much as possible. The model behaviour in OMNeT++ is captured in C++ files as code, while the model topology and the parameters defining the network topology, is defined in the NED files. This approach allows the user to keep the different aspects of the model in different places which in turn allows having a model and support. It is important in a generic simulation scenario the behaviours of the different inputs (Varga & Hornig 2008).

The INI files are used to store these values. INI files provide a great way to specify how these parameters change and enables the running of the simulation for each parameter combination (Varga & Hornig 2008). The generated simulation results can be easily harvested and processed by the built-in analysis tool. That was explored, in the result analysis chapter 8, of this research study.

1.6. Simple Module Programming Model

The simple modules are the active elements in a model. They are atomic elements in the module hierarchy: they cannot be divided any further. The simple modules in OMNeT++ are programmed in C++, using the OMNeT++ simulation class library (Varga & Hornig 2008). OMNeT++ provides an Integrated C++ Development Environment so it is possible to write, run and debug the code without leaving the OMNeT++ IDE. The simulation kernel

does not distinguish between messages and events as events are also represented as messages (Gimenez et al 2013 & Abo-Zahhad et al 2014).

Simple modules in OMNeT ++ are programmed using the process-interaction method. The user implements the functionality of a simple module by sub classing the cSimpleModule class. The functionality is added through various alternative programming models: which include coroutine-based and event-processing function. When using co-routine-based programming, the module code runs in its own (non-pre-emptively scheduled) thread, which receives control from the simulation kernel each time the module receives an event (=message). The function containing the co-routine code typically never return, usually it contains an infinite loop with send and receive response (Varga & Hornig 2008).

1.7. Design of the Simulation Library

The OMNeT++ provides a rich object library for simple module implementation. There are several distinguishing factors between this library and other general-purpose simulation libraries. The OMNeT++ class library provides reflection functionality which makes it possible to implement high-level debugging and tracing capability, as well as automatic animation (Varga & Hornig 2008). There are issues of memory leaks, pointer aliasing and other memory allocation problems in C++ programs if not well coded. OMNeT++ alleviates this problem by tracking object ownership and detecting bugs caused by the pointers and misuse of shared objects.

The requirements for ease of use, modularity, open data interfaces and support of embedding also heavily influenced the design of the class library. The consistent use of object-oriented techniques makes the simulation relatively easy to understand its internals, which is a useful property for both debugging and educational use (Gimenez et al 2013). It has become common to do large scale network simulations with OMNeT++, with several ten thousand or more network nodes (Abo-Zahhad et al 2014). To address this requirement, aggressive memory optimization has been implemented in the simulation kernel, based on shared objects and copy-on-write semantics (Varga & Hornig 2008).

1.8. Contents of the Simulation Library

This section provides an overview of the catalog of the classes in the OMNeT++ simulation class library. The classes were designed to cover most of the common simulation tasks. OMNeT++ has the ability to generate random numbers from several independent streams. The common distributions are supported, and it is possible to add new distributions programmed by the user. It is also possible to load the user distributions defined by

histograms. The class library of OMNeT++ offers queues and various other container classes. Queues can also operate as priority queues. Messages are objects which may hold arbitrary data structures and other objects through aggregation or inheritance and can also embed other messages. OMNeT++ supports routing traffic in the network. This feature provides the ability to explore actual network topologies extract it into a graph data structure (Varga & Hornig 2008).

1.9. Parallel Simulation Support

OMNeT++ also has support for parallel simulation execution. Very large simulations may benefit from the parallel distributed simulation (PDES) feature, either by getting speedup, or by distributing memory requirements (Varga & Hornig 2008).

1.10. Real-Time Simulation, Network Emulation

OMNeT++ Network emulation, together with real-time simulation and hardware-in-the-loop like functionality, is available because the event scheduler in the simulation kernel is pluggable. The OMNeT++ distribution contains a demo of real-time simulation and a simplistic network emulation (Varga & Hornig 2008).

1.11. Organising and Performing Experiments

The ultimate goal of running a simulation is to obtain results and to get some insight into the system by analysing the results. A thorough simulation study produces both small and large amount of realistic data, which are organised to produce results in a meaningful way. OMNeT++ simulation runs generates results around the following concepts: the study model, experiment, measurement, replication and the actual run (Varga & Hornig 2008).

OMNeT++ supports the execution of the whole or partial experiments as a single batch. After specifying the model (executable file + NED files) and the experiment parameters (in the INI file) one can further refine the measurement of interest once the simulation batch is executed, the progress is monitored from the IDE.