# THE FORMATION OF A COLLABORATIVE NETWORK BY MOBILE AGENTS IN THE UBIQUITOUS ENVIRONMENT



LEE THIAN SENG

UNIVERSITI KEBANGSAAN MALAYSIA

# THE FORMATION OF A COLLABORATIVE NETWORK BY MOBILE AGENTS IN THE UBIQUITOUS ENVIRONMENT

LEE THIAN SENG



THESIS SUBMITTED IN FULFILMENT FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

INSTITUTE OF VISUAL INFORMATICS
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2023

# PEMBENTUKAN RANGKAIAN KERJASAMA OLEH EJEN MUDAH ALIH DALAM PERSEKITARAN UBIQUITOUS

LEE THIAN SENG



TESIS YANG DIKEMUKAKAN UNTUK MEMPEROLEH
IJAZAH DOKTOR FALSAFAH

INSTITUT INFORMATIK VISUAL
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2023

UNIVERSITI KEBANGSAAN MALAYSIA
*The National University of Malaysia*

# PERAKUAN TESIS SARJANA / DOKTOR FALSAFAH
## *(DECLARATION OF MASTER / DOCTOR OF PHILOSOPHY THESIS)*

| **A. MAKLUMAT PELAJAR/*STUDENT DETAILS*** | |
|---|---|
| Nama<br>*Name* | Lee Thian Seng |
| No. Pendaftaran<br>*Registration No.* | P78664 |
| Fakulti/Institut<br>*Faculty/Institute* | Institut Informatik Visual (IVI) |
| *Program Pengajian*<br>*Program of Study* | ~~Sarjana~~ / Doktor Falsafah \| ~~*Masters*~~ */ Doctor of Philosophy* |
| Tajuk Tesis<br>*Thesis Title* | The Formation of A Collaborative Network By Mobile Agents in The Ubiquitous Environment |
| Mel-e/*Email*: thianseng01@yahoo.com | No. Telefon/*Tel.No: 0163687161* |

**B. PERAKUAN / *DECLARATION***

Merujuk kepada Dasar Harta Intelek UKM 2018, **tesis adalah hak milik UKM** seperti keterangan dibawah:

4.2.1 Harta Intelek yang direka cipta oleh pelajar termasuk tesis dan semua hasil penyelidikan ditulis oleh Pelajar UKM dalam tempoh pengajiannya di UKM direka cipta, dibangunkan atau dihasilkan menggunakan kemudahan, bahan, dana, atau sumber lain adalah hak milik UKM melainkan dinyatakan sebaliknya dalam Dasar ini atau dipersetujui sebaliknya oleh UKM dan Pelajar/Penaja.

*Referring to the UKM Intellectual Property Policy 2018, **the thesis is the property of UKM** as described below:*

*4.2.1 Intellectual property created by the student, including the thesis and all research output produced by the UKM Student during their studies at UKM, including designed, developed or produced output using facilities, materials, funds, or other resources are the property of UKM unless otherwise stated in this Policy or otherwise agreed by UKM and the Student/Sponsor.*

| | | |
|---|---|---|
| | **RAHSIA**<br>***CONFIDENTIAL*** | Mengandungi maklumat rahsia di bawah AKTA RAHSIA RASMI 1972<br>*Consisting of classified information under the OFFICIAL SECRETS ACT 1972* |
| | **TERHAD**<br>***RESTRICTED*** | Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi / badan di mana penyelidikan dijalankan<br>*Consisting of RESTRICTED information which has been determined by the organisation/body where the research was conducted* |
| ✓ | **AKSES TERBUKA/ TIDAK TERHAD**<br>***OPEN ACCESS/ NON-RESTRICTED*** | Saya membenarkan tesis ini diterbitkan secara akses terbuka, teks penuh atau dibuat salinan untuk tujuan pengajian, pembelajaran & penyelidikan sahaja<br>*I give permission for this thesis to be published through open access, full text or copied only for study, learning, and research purposes.* |

Bagi kategori Akses Terbuka/Tidak Terhad, saya membenarkan tesis (Sarjana/Doktor Falsafah) ini di simpan di Perpustakaan Universiti Kebangsaan Malaysia (UKM)* dengan syarat-syarat kegunaan seperti berikut:

*For the Open Access/Non-Restricted category, I permit this (Master's/Doctoral) Thesis to be kept in the Universiti Kebangsaan Malaysia (UKM) Library with the following conditions:*

1. Perpustakaan UKM mempunyai hak untuk membuat salinan untuk tujuan pengajian,pembelajaran, penyelidikan sahaja.
   *UKM Library has the right to reproduce the thesis only for study, learning and research purposes.*
2. Perpustakaan Universiti Kebangsaan Malaysia dibenarkan membuat satu (1) salinan tesis ini untuk tujuan pertukaran antara institusi pengajian tinggi dan mana-mana badan/ agensi kerajaan, tertakluk kepada terma dan syarat.
   *UKM Library is allowed to make one (1) copy of this thesis for exchange purposes among higher education institutions and any government body/agency, subject to the terms and conditions.*

| | |
|---|---|
| Saya mengakui maklumat & tesis yang diberikan adalah benar & terkini.<br>*I acknowledge the information & the thesis provided are correct & current.*<br><br>Tandatangan Pelajar:<br>*Student's Signature*<br><br>*Tarikh / Date:* | Saya memperakukan maklumat & tesis yang diberikan adalah benar & terkini.<br>*I verify that the information & the thesis provided are correct & current.*<br><br>Tandatangan Penyelia / *Supervisor's Signature*:<br>Nama / *Name*:<br>Cop Rasmi / *Official Stamp*:<br>*Tarikh / Date:* |

**C. PENGESAHAN FAKULTI/INSTITUT | *VERIFICATION OF FACULTY/INSTITUTE***

Tandatangan/*Signature*:
Nama/*Name*:
Tarikh/*Date*:
Cop Rasmi/*Official Stamp*:

**Kuatkuasa:15 Julai 2021

## DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

30 August 2023

LEE THIAN SENG
P78664

**ACKNOWLEDGEMENT**

I would like to express my gratitude to my esteemed research supervisor, Prof. Ir. Dr. Riza Sulaiman, and my Co-supervisors, Associate Prof. Dr. Nazlena Mohamad Ali, and Dr. Paul Gardner-Stephen for their invaluable guidance, patience, and tutelage during my PhD study. Their immense knowledge and expertise have encouraged me in all my academic research and daily life.

Additionally, I would like to express appreciation to all my institute members and friends for the time spent together in the lab and in social settings. A special thanks to my fellow close friends, who stood by my side during my downturn.

Finally, I would like to express my infinite gratefulness to all my family members, especially my parents, in-laws, wife, dear son, and my siblings. Without their tremendous understanding and continuous support in the past few years, it would have been impossible for me to complete my study.

## ABSTRAK

Walaupun perkembangan pesat dalam protokol dan teknologi rangkaian telah berlaku selama ini, rangkaian komunikasi berasaskan infrastruktur masih menghadapi cabaran terutamanya pada kawasan yang mempunyai infrastruktur rangkaian terhad, atau semasa krisis seperti bencana alam, pergolakan awam atau peperangan. Oleh itu, model komunikasi alternatif untuk tujuan komunikasi sementara masih diperlukan. Kajian literasi melaporkan bahawa peranti mudah alih dilengkapi dengan keupayaan komunikasi yang canggih tetapi tidak dipergunakan sepenuhnya. Ini mendorong kepada idea pembentukan model komunikasi sementara berdasarkan fungsi Wi-Fi peranti mudah alih yang sedia ada. Pada mulanya, analisis awal telah dijalankan untuk menyiasat operasi protokol Wi-Fi P2P dan kajian lepas yang berkaitan. Ini termasuk kajian skop sistematik untuk mengenal pasti kaedah analisis dan hasil kajian yang telah dilaporkan oleh penyelidik. Selain itu, dokumentasi teknikal juga dikaji untuk mendapatkan pemahaman terperinci tentang operasi asas protokol yang berkaitan. Seterusnya, model pertukaran data baharu direalisasikan dengan menyesuaikan protocol "Service Discovery" yang sedia ada dalam rangka kerja Wi-Fi P2P. Sebuah prototaip berdasarkan model tersebut telah dibangunkan menggunakan Android Studio IDE untuk tujuan ujian penilaian. Akhirnya, beberapa kes ujian telah dibentuk dan eksperimen pada masa nyata telah dijalankan. Prototaip yang dibina telah dipasangkan ke dalam semua telefon pintar untuk unjian dengan muatan mesej yang berbeza (60,120,180, dan 240). Setiap set kes ujian telah dilaksanakan sekurang-kurangnya 100 kali untuk mengumpul data untuk analisis. Alat "Logcat" dalam Android Studio telah digunakan untuk pengumpulan data. Analisis ke atas data yang dikumpul menunjukkan bahawa model tersebut boleh menyokong muatan maksimum 240 bytes setiap mesej. Berdasarkan muatan yang berlainan sebagai kes ujian, purata latensi dan goodput yang didapati ialah 2.4 saat dan 1.6 kbps. Sumbangan utama penyelidikan ini termasuk pengenalan sebuah model komunikasi yang boleh diadaptasikan atau diperkembangkan oleh penyelidik lain dalam bidang kajian yang sama. Selain itu, prosedur sistematik yang digunakan untuk mengukur prestasi model komunikasi dalam kajian ini boleh menjadi garis panduan untuk para penyelidik yang lain untuk tujuan pengukuran yang sama. Sekiranya berlaku kegagalan komunikasi, model yang dicadangkan akan berfungsi sebagai alternatif untuk komunikasi sementara atau kecemasan. Oleh itu, hasil penyelidikan ini akan memberi manfaat kepada pengguna mudah alih yang bertambah di pasaran dan akhirnya meluaskan liputan rangkaian komunikasi alternatif.

## ABSTRACT

Despite the rapid emergence of network protocols and technologies over the years, infrastructure-based communication networks remain volatile, especially in areas with limited network infrastructure or during crises such as natural disasters, civil unrest, or war. Therefore, an alternative and temporary form of communication is needed. The literature has shown that mobile devices are equipped with excellent communication technologies but are underutilized. This leads to the idea of developing a temporary data exchange model by leveraging the mobile device's Wi-Fi interface. Initially, a preliminary analysis was conducted to investigate the operation of the Wi-Fi P2P protocol and related literature. This analysis included a scoping review to identify the analytical methods and experimental results documented by researchers. Furthermore, the technical documentation of the protocols was studied to gain a detailed understanding of the underlying operations. Next, a new data exchange model was realized by exploiting the Wi-Fi P2P service discovery protocol within the Wi-Fi P2P framework. A prototype for testing was developed using the Android Studio IDE. Finally, multiple test cases were conducted on a real-time experimental testbed. The prototype was installed on all smartphones for testing with different message payloads (60, 120, 180, and 240 bytes). Each set of test cases was executed at least 100 times to collect data for analysis. The Logcat tool in Android Studio was used for data collection. Analyzing the collected data revealed that the model can support a maximum payload of 240 bytes per message. When considering various payloads as test cases, the recorded average latency and effective goodput were 2.4 seconds and 1.6 kbps, respectively. The main contribution of this research is a communication model that other researchers can adopt or extend to leverage wireless interfaces with similar characteristics. Furthermore, the systematic procedures used in the experimental testbed can serve as a guide for performance measurement in similar domains. In the event of communication failure, the proposed model will serve as an alternative for temporary or emergency communication. Hence, the outcomes of this research will benefit the rapidly growing mobile user market and ultimately expand the coverage of alternative communication networks.

# TABLE OF CONTENTS

# LIST OF TABLES

## LIST OF ILLUSTRATIONS

## LIST OF ABBREVIATIONS

| | |
|---|---|
| ANQP | Access Network Query Protocol |
| AIDL | Android Interface Definition Language |
| BSS | Basic Service Set |
| BT | Bluetooth |
| CDF | Cumulative Distribution Frequency graph |
| D2D | Device-to-device |
| DLL | Data Link Layer |
| DMG | Directional Multi-Gigabit |
| DNS | Domain Name System |
| DTN | Delay Tolerance Network |
| GAS | Generic Advertisement Service |
| GO | Group Owner |
| GM | Group Member |
| GSMA | Global System for Mobile Communications Association |
| HIDL | HAL Interface Definition Language |
| IBSS | Independent basic service set |
| IE | Information Element (in a frame) |
| IEEE | Institute of Electrical and Electronics Engineers |
| ITU | International Telecommunication Union |
| LMIC | Low- and Middle-Income Countries |
| MAC | Media Access Control |
| MANET | Mobile Ad-hoc Network |
| MMPDU | MAC Management Protocol Data Unit |
| NFC | Near Field Communication |
| NDEF | NFC Data Exchange Format |
| NSD | Network Service Discovery |
| OppNet | Opportunistic Networks |

| | |
|---|---|
| OS | Operating System |
| OSI | Open Systems Interconnection |
| P2P | Peer-to-Peer |
| PHY | Physical Layer (OSI layer) |
| PPDU | Physical Layer Protocol Data |
| PTR | Pointer Record |
| RSSI | Received Signal Strength Indicator |
| RF | Radio Frequency |
| RFC | Request for Comments |
| SNR | Signal-to-Noise Ratio |
| STA | Station |
| TLV | Type Length Value |
| TU | Time Unit (1024 microseconds) |
| UI | User Interface |
| UID | Unique ID |
| WCN | Wireless Collaboration Network |
| WFD | Wi-Fi Direct |
| WPA | Wi-Fi Protected Access |
| WPS | Wi-Fi Protected Setup |
| WSC | Wi-Fi Simple Configuration |

# CHAPTER I

# INTRODUCTION

## 1.1 INTRODUCTION

On 11th March 2020, the World Health Organization declared COVID-19 as a pandemic (World Health Organization, 2020). Countries around the world have implemented measures to slow the spread of the virus, including forbidding cross-country travel and imposing nationwide lockdowns. People are advised to stay at home and are restricted from participating in any forms of social events and gatherings. Brick-and-mortar businesses and various industries have been forced to temporarily halt their operations. Schools and universities have also been temporarily closed to contain the spread of the pandemic. As a result, people from all walks of life have begun to work from home, and students are attending classes and submitting assignments online. The pandemic has drastically changed the lifestyle of almost everyone around the world. Social activities are mostly conducted online, leading to an increased demand for communication and collaboration over networking platforms.

Unlike the pandemic outbreaks, which happened infrequently, natural disasters occur every year around the world. The catastrophic effects of these disaster result not only in economic losses but also in the loss of human life. Major disaster such as the earthquake in Haiti (2010), Japan (2011) and Nepal (2015) have impacted millions of people. During such disasters, a stable communication network is of utmost important, as it serves as the only way for the crisis management team to reach the victims. However, the breakdown of communication network is widely known to occur in almost all extreme conditions (Khaled & Mcheick, 2019; Yash & Nainesh, 2017). This situation increases the duration and complexity of rescue works and evacuation efforts.

Therefore, a standalone communication platform should be available as a backup in case of emergencies. Unlike infrastructure-based communication networks, a standalone communication platform is a channel that is free of any fixed infrastructure. It should be readily available or deployable for emergency usage. When an emergency occurs and existing communication channels fail, rescuers and victims can use this independent communication platform to communicate.

For many years, countries around the world have invested significant resources in establishing stronger and better wireless network infrastructures (Oughton et al., 2023). For instance, the fifth-generation mobile network (5G) has taken cellular technology to another level. The rapid advancement of new networking technology has brought about major changes in the networking and communication sector. Despite the emergence of these new networking technologies, there are still some people who cannot enjoy the benefits, especially in low- and middle-income countries (LMIC).

Base on the ICT statistic report from International Telecommunication Union (ITU), 53.6% of the global population was using the Internet at the end of 2019 (International Telecommunication Union, 2019). In another word, there are still 46.4% of the global population who do not have the access to the Internet. Therefore, there is a need for an affordable networking platform for the underserved population.

Besides personal computers, mobile phones are another medium for accessing the Internet. Reports have shown that smartphones with wireless networking capabilities have become the primary communication tool of people worldwide (International Telecommunication Union, 2019). The mobile Internet has become the main platform for all smartphone users to perform online activities such as communication, information searching, entertainment, financial transactions and more (Bahia & Suardi, 2019).

In recent years, statistical data has indicated a growing trend in the usage of mobile devices' wireless technology (e.g., Wi-Fi) as the primary media to access the Internet (Cisco, 2019). Researchers have introduced proposals, hardware, software,

frameworks, architectures, and protocols to complement infrastructure-based communication network, utilizing wireless technology. These supplementary wireless communication networks intersect with the wireless broadband infrastructure network at certain points to connect to the Internet. Therefore, in the event of an existing infrastructure network breakdown, there will be an alternative network available for the end-users.

## 1.2 BACKGROUND

Since the outbreak of the COVID-19 pandemic, mobile devices have become indispensable tools not only for communication but also for shopping, banking, entertainment, and various other online activities on the Internet. While the pandemic highlighted the importance of reliable mobile networks, natural disasters also occur frequently worldwide, causing significant economic losses and loss of life. In such critical and extreme situations, a strong and stable mobile network becomes even more essential. Moreover, it is well-known that communication breakdowns occur during almost all major disasters, further exacerbating challenges for crisis management teams in rescue and relief work (Abraham, 2011).

According to Beech (2020), the pandemic has pushed up global Internet usage by 70% and streaming by more than 12%. Telecommunication equipment producer Nokia also reported that most countries' telecom networks have experienced increased Internet usage, with peaks being around 20 to 40 percent higher than the previous year, around March 2019 (The Star, 2020). Due to the peak network usage and network instability, end-users might face network connectivity problems, such as slow network bandwidth or even loss of network connection (Scott, 2020). According to the news reported by the New Straits Times (2020), the Malaysian Communications and Multimedia Commission (MCMC) noticed a surge in Internet bandwidth during the government's movement control order (MCO). The sudden spike in bandwidth was due to the increased use of video conferencing, online learning, and online shopping (New Straits Times, 2020). Based on all the up-to-date reports discussed above, it is evident that even with a strong backbone of networking infrastructure, the sudden spike in network traffic might result in network congestion or bandwidth overload in

certain areas (Scott, 2020). The sudden spike in bandwidth during the lockdown was not a surprising phenomenon, as almost everyone was restricted from traveling physically from one place to another. Hence, it is very important to have an alternative communication platform that serves as a backup for people to stay connected, especially during critical moments.

Over the years, the wireless technology of mobile devices has rapidly evolved. In the same decade, new advancements were introduced, including the new 5G cellular network, the updated Bluetooth (5.0) standard, and the new Wi-Fi 6 (802.11ax) protocol. The evolved wireless technologies offer higher bandwidth, broader coverage, or lower latency. In the research field, the advancements in the Mobile Ad-hoc Network (MANET) research area have inspired the possibility of a communication network mainly formed by end-user's devices. Within the MANET research area, both Opportunistic Networks (OppNets) and Vehicular Networks (VANETs) have captured the attention of many researchers in recent years. OppNets researchers primarily focus on forming MANET through the wireless technology of mobile devices. On the other hand, VANET researchers concentrate on forming MANET through the wireless technology of vehicles.

There should be a highly stable and robust mobile communication system, especially when mobile devices' wireless technology and the MANET research area are progressing in parallel. At the very least, there should be an independent communication network available in proximity areas. End-users should have an alternative option for end-to-end devices' communication without solely relying on the service provider's network. An autonomous mobile network would prove invaluable during critical situations like disasters or pandemics. It would enable people from lower-income communities to access communication technology without worrying about unsustainable service charges. Unfortunately, in many countries at the time of writing, this is not the case.

## 1.3 PROBLEM STATEMENT

Statistics have revealed a continuous and significant increase in mobile network usage over the years. However, the current infrastructure-based communication network is facing persistent challenges in maintaining uninterrupted availability. As the number of mobile device users continues to grow, the strain on the existing communication infrastructure intensifies, leading to potential network congestion and reliability issues.

To address these challenges, there is an urgent need for innovative solutions that can act as temporary or alternative communication systems, ensuring continuous connectivity for mobile device users. While various communication models, protocols, and routing devices have been integrated into the network to establish temporary communication systems, none have proven to be universally effective and robust in addressing the issue.

Amidst the ongoing exploration of alternative communication systems, the potential of leveraging the Wi-Fi interfaces of mobile devices has emerged as a promising solution. The ubiquitous nature of mobile devices, coupled with their widespread usage, makes them valuable resources for forming a Wireless Collaboration Network (WCN). By utilizing the Wi-Fi interfaces of these devices, it is possible to create a dynamic and flexible communication network that can adapt to changing network conditions and provide seamless connectivity to users.

Despite the promising potential, the research on developing a comprehensive model for WCNs based on Wi-Fi interfaces is still inconclusive. There is a need for in-depth investigations and rigorous studies to design an efficient and standardized model that can be successfully implemented in mobile devices.

## 1.4 RESEARCH OBJECTIVES

The proposed research aims to fill this gap by undertaking a detailed exploration of WCNs and their viability as a reliable and efficient communication system. By analyzing the advantages and challenges associated with WCNs, the research seeks to

develop a robust model that can optimize the utilization of Wi-Fi interfaces on mobile devices. The model will be designed to ensure seamless connectivity, even in areas with limited infrastructure support, and serve as a sustainable communication solution for various scenarios. Through rigorous experimentation and empirical analysis, the research will strive to provide insights into the capabilities and limitations of WCNs. By examining real-world implementations and conducting performance evaluations, the study aims to offer valuable recommendations for the adoption of WCNs in practical settings. To achieve the main aim, the following objectives must be accomplished.

1) Conduct a preliminary analysis to ascertain the constraints, limitations, and potential solutions pertaining to the establishment of a WCN.
2) Develop a new model for communication operations in WCN by adapting the store-and-forward routing mechanism.
3) Measure the performance of the model by implementing it on mobile devices and assessing the rate of successful data transfer.

## 1.5 RESEARCH SCOPE

### 1.5.1 Research Area

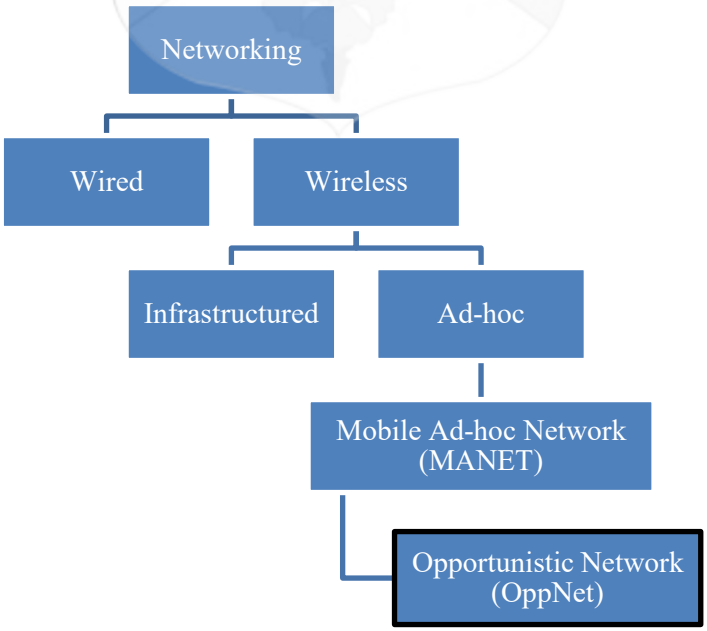



Figure 1.1 The hierarchy of this research domain

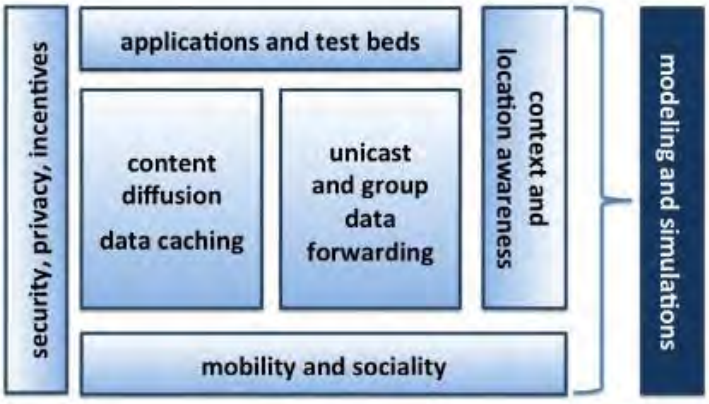



Figure 1.2 Overview of the research issues in the opportunistic networking field.
Source: Boldrini et al. (2014)

Based on an extensive review of literature, this research is positioned within the Opportunistic Network (OppNet) research domain, as depicted in Figure 1.1. Boldrini et al. (2014) summarized the research areas that falls under the OppNet research domain. As illustrated in Figure 1.2, OppNet covers a diverse range of research areas, and this study specifically focuses on people-centric unicast and group data forwarding mechanisms. This research is devoted to designing a data dissemination model that benefits from the mobility of mobile device users. The proposed data dissemination model adopts the store-and-forward routing approach within the delay-tolerant OppNet framework (Conti et al., 2015). Additionally, the research emphasizes applications and testbed deployment, as indicated in Figure 1.2.

The primary objective of this research is to design a data dissemination model that leverages the mobility of mobile device users. To achieve this aim, the developed model has been implemented into a smartphone application and deployed in a testbed to evaluate its performance effectively. The focus areas and their corresponding objectives are summarized in Table 1.1.

Table 1.1 The mapping of research objectives with the research area.

| | Objectives (section 1.4) | Focus area under OppNet research domain (from Figure 1.2) |
|---|---|---|
| Research Aim (section 1.4) | Objective 1 & Objective 2 | Unicast and group data forwarding |
| | Objective 3 | Applications and testbeds |

### 1.5.2 The Wireless Interface for the WCN

By default, the majority of mobile devices are equipped with built-in Bluetooth or Wi-Fi interfaces. While some researchers have concentrated on amalgamating these two wireless interfaces to establish heterogeneous connections within WCN (Bellavista et al. 2010; Dubois et al. 2015; Wang et al. 2016), the prevailing approach involves leveraging the Wi-Fi interface due to its broader signal transmission range and higher bandwidth (Camps-Mur et al., 2013; Gardner-Stephen et al., 2013; Han et al., 2014, 2014; Lee et al., 2018; Liu et al., 2016; Michel Lombera et al., 2014; Raj et al., 2014; Wang et al., 2016). As a result, this study primarily emphasizes the utilization of the Wi-Fi interface within mobile devices. Notably, the mobile device's Wi-Fi interface incorporates two types of Wi-Fi protocols, each designed to facilitate either infrastructure mode or P2P mode wireless connections. This investigation zeroes in on the P2P protocol, as it aligns with the necessity for direct connections among Wi-Fi-enabled mobile devices in the creation of WCNs.

### 1.5.3 The Mobile System Platform

The term 'mobile devices' in a WCN covers a wide range of mobile products. In this work, the scope of mobile devices has been narrowed to focus only on smartphones, considering their ubiquitous nature. There are many smartphone brands in the market, each equipped with different types of operating systems (OS). All smartphones are equipped with at least one or more Wi-Fi interfaces that support basic Wi-Fi connectivity. However, the implementation of the P2P Wi-Fi connectivity function is OS-dependent. For instance, Apple Inc. has integrated the Multipeer Connectivity framework (also known as Apple Wireless Direct Link, AWDL) into their proprietary OS (iOS). On the other hand, Android OS-based devices use the P2P connection standard known as Wi-Fi Direct (WFD).

As of the time of writing, the integration of P2P Wi-Fi connectivity between the two aforementioned OS is not possible. According to a survey report by Holst (2019), Android has been the leading OS since 2011. Furthermore, Android is an open-source OS, and its core library and API of the wireless framework are open and

accessible to developers. Therefore, this research focuses solely on Android's WFD standard.

Over the years, the Android OS has been upgraded and newer version were released to the market. At the point of writing, the latest Android OS version introduced was Android 10 at API level 29. Nonetheless, the number of smartphones that are running the newer Android OS version is still very limited. Figure 1.3 shows the cumulative distribution percentage of Android platform version to the smartphones available in the market. The chart was generated through the Android Studio software (Google LLC & JetBrains, 2020).

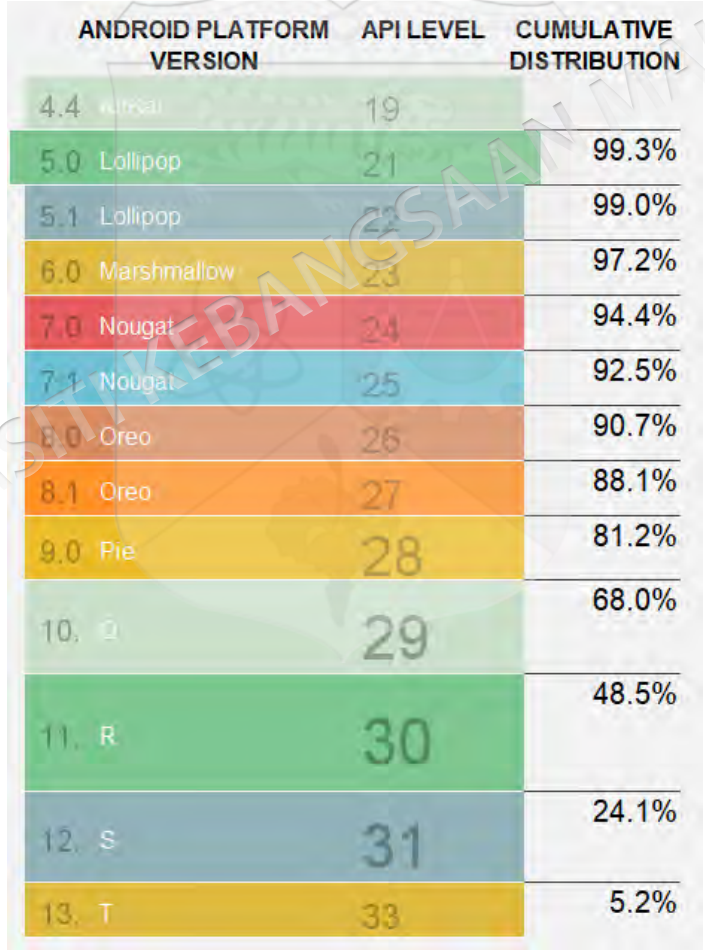



Figure 1.3 Android Platform or API version distribution to the smartphone in the market

Source: "Android Studio Dolphin" (2023)

The formation of a strong WCN requires as much participation of the smartphones as possible. Hence, the proposed model should be built on the Android

platform version which is able to cover most of the smartphones available in the market. The Android platform in between version 4.0 to 5.0 has been chosen as the targeted development platform as the cumulative distribution percentage is close to 100%, as depicted in Figure 1.3.

## 1.6 SIGNIFICANCE & CONTRIBUTION OF THE STUDY

### 1.6.1 Contribution to the Society

The WCN is an ad-hoc P2P network formed by the end user's mobile devices through common wireless interfaces. Ideally, the WCN serve as a supplementary wireless-based communication network that intersects with the infrastructure network at some points to connect to the Internet. The communication system plays a very crucial role, especially in harsh environments like during disasters The formation of the WCN will provide a temporary or emergency communication system in case of breakdowns during calamities. Statistics have shown a rapid increase in the number of mobile users and participation in networking activities every year. Therefore, it is predicted that the research outcome will benefit the rapidly growing mobile market. With more mobile users participating in the WCN, the communication model's coverage area will expand, allowing people to communicate anywhere, anytime.

### 1.6.2 Contribution to the Research Domain

By accomplishing the three objectives (section 1.4), this research has introduced a new communication model. In terms of research domain, this work has contributed to the "unicast and group data forwarding" and "applications and test beds" research area under OppNet (Figure 1.2). The developed communication model can be adopted by other researchers to implement on different types of DTN. Additionally, the algorithms introduced in the model can be extended and applied to new wireless interfaces that may be introduced to mobile devices in the future. The systematic procedures and tools applied to the experiment testbeds in this work can be duplicated and referenced by other researchers conducting experiments with a similar aim.

## 1.7 THESIS ORGANIZATION

This thesis is divided into six chapters, namely Introduction (Chapter 1), Literature Review (Chapter 2), Research Methodology (Chapter 3), Research Outcome and Evaluation (Chapter 4), and Conclusion (Chapter 5). Chapter 1 introduces the background, problem statement, research aim, objectives, scope, and significance of this study. Chapter 2 further explains the extensive and detailed literature review related to this research. The literature in Chapter 2 helps establish a strong research foundation to achieve research goals (as discussed in Chapter 1), while the formation of the research methodology and the procedures for generating research outcome are explained in Chapter 3. The detailed experiment results and evaluation are elaborated in Chapter 4. Finally, Chapter 5 summarizes the research and introduces suggestions for future work.

# CHAPTER II

# LITERATURE REVIEW

## 2.1 INTRODUCTION

As demonstrated in Figure 1.1 the research domain of this research falls under OppNet. OppNet is a sub-domain under MANET, which, in turn, is a sub-domain under ad-hoc network. Ad-hoc networks are networks that form instantaneously between two or more devices. MANET, being a mobile ad-hoc network, is formed by mobile devices. Due to the mobility nature of these devices, the network medium of MANET is wireless. Therefore, the focus of this research is to establish a communication network using wireless technology on mobile devices.

At the beginning of this chapter, a comparison of common wireless technologies in mobile devices was conducted. Based on this comparison, the Wi-Fi and WFD protocols were chosen as the transport for the new model due to their numerous benefits. This leads to the second part of this chapter, which provides a detailed explanation of the WFD protocol. The literature review of past research involved a critical review and categorization of collected literature based on OppNet formation by hardware and software. To specifically understand the relevance of past literature related to WFD technology, the scoping review methodology was applied. The screened and reviewed articles were then grouped into a few categories. The following section delves into the wireless technology in mobile devices.

### 2.1.1 Research Area

An alternative wireless communication network is needed during the critical event. The supplementary wireless communication network is a type of Mobile Ad-hoc Network (MANET). The MANET is an ad-hoc Peer-to-Peer (P2P) network formed by

the mobile devices through the wireless interfaces like Bluetooth and Wi-Fi. Considering the possibility of network disruption and network connection instability nature, the MANET has been categorized as a type of Opportunistic Network (OppNet). An OppNet is an instance of the Delay Tolerance Networking (DTN) paradigm. Like MANET, the mobile devices (nodes) in an OppNets play the role as a router to transmit the data, as well as the end devices that send and receive data.

Unlike MANET, the connection paths between the source and destination devices of OppNet are not known, and there is a possibility of data transmission disruptions. The disruptions might be caused by the network instability, the device being turned off, or running out of battery. The mobility nature of the mobile devices in MANET is a limitation. However, the same nature has become an advantage(opportunistic) in OppNet, increasing the chances of data reaching the targeted devices.

With the DTN paradigm in OppNet, the mobile devices in the OppNet are generally equipped with the store-carry-and-forward functionalities. The data sent from a source will be stored and forwarded by the intermediary devices (mobile devices), repeatedly until the data has reached the destination (Boldrini et al., 2014).

In this research, the research area has been refined from OppNet to Wireless Collaborative Network (WCN) (Lee & Sulaiman, 2019), to emphasize the collaboration of mobile devices in forming a wireless network for data transmission activity. By participating in the WCN, the mobile devices can perform networking activities such as messaging, communicating, and file sharing. By forming the WCN, an alternative to support the shortfall of the infrastructure-based communication network has been established.

### 2.1.2 The Challenges of a Stable Network

After the outbreak of influenza pandemic in 2009, Abraham (2011) wrote a review article titled “Lessons from the pandemic: the need for new tools for risk and outbreak communication”. In the review article, the author explains the importance of having a standard communication protocol and an alternative communication platform

(Abraham, 2011). During a pandemic, the communication platform is crucial as it is the fastest way for communicators (government and authorities) to deliver the latest information to the public. The Internet is the largest communication platform because most mobile devices with web and web-based tools provide valuable channels for people to communicate. As mentioned before, the recent COVID-19 pandemic has changed the lifestyle of almost everyone. Networking and communication technology are playing a more important role than ever, as it is the only way for people in different places to communicate and stay connected.

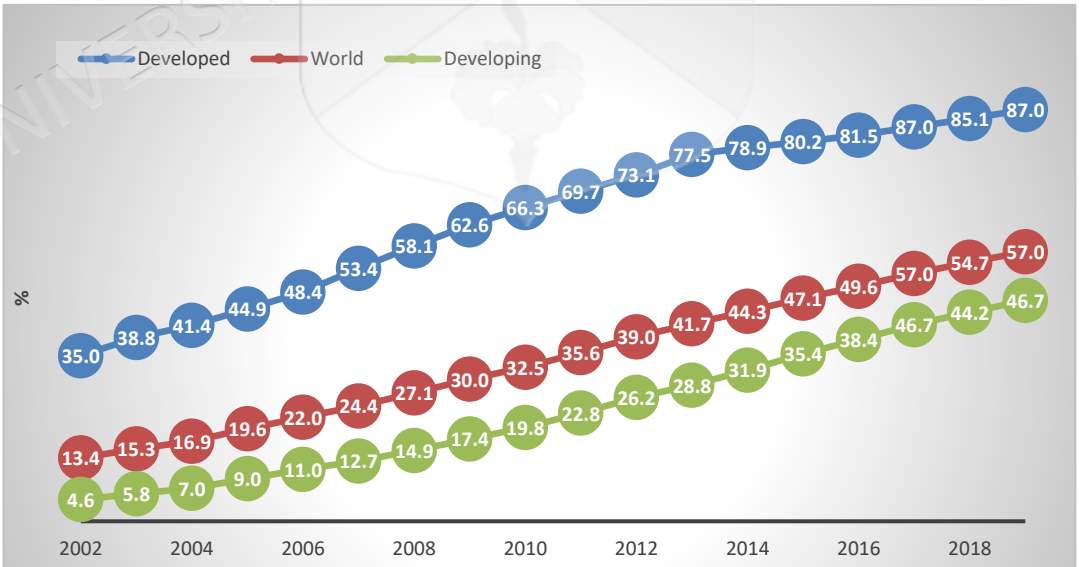
The World Health Organization (WHO) and partners have classified disasters into four main types: Natural, Biological, Technological and Societal (WHO et al., 2011). Not only do disasters cause serious damage to goods and habitation, but they also lead to the destruction of essential facilities such as communication, power supply, and transportation. A reliable and accessible communication system is crucial before, during, and after a disaster. However, the loss of power and insufficient fuel for backup generators have been identified as the main causes of telecommunication equipment being out of service (ITU-T Focus Group on Disaster Relief Systems Network Resilience and Recovery, 2013). In another report by Yash & Nainesh (2017), three reasons for communication system failures are the destruction of communication system components, damage to supporting infrastructure, and network congestion due to physical damage. In the event of a communication system breakdown during a disaster (e.g., flood, earthquake, and tornado), setting up a new network will take a significant amount of time and cost.

Therefore, preventative measures and a precautionary system must be available at all times. A disaster mitigation and preparedness framework has been developed by the government and researchers of a certain country as a preventive measure to manage emergency situations during disasters (State Emergency Management Committee-Western Australia, 2012). Based on the discussed disaster mitigation standards and reports, a temporary communication system must be made available in the event of an existing communication system breakdown. However, Gomes et al., (2016) and Khaled & Mcheick (2019) reported that forming a temporary communication system is challenging due to physical destruction, lack of compatibility between technologies,

and communication system congestion. Therefore, more efforts are needed to establish an alternative communication platform.

To enjoy the communication services at anytime and anywhere, there will be a cost. Depending on the needs, every mobile-broadband subscriber will be charged based on the data plan package chosen. As reported by the International Telecommunication Union (2019), active mobile-broadband subscriptions are increasing steadily every year. The Internet has become the largest communication platform for most mobile users. However, mobile Internet connectivity has not grown equally.

Referring to Figure 2.1, only 46.7% of the households in the developing country can enjoy Internet connectivity, compared to 87% of households in the developed country. To study the state of the global mobile Internet connectivity trend, the Global System for Mobile Communications Association (GSMA) has initiated the measurement of Mobile Connectivity Index (MCI), which covers the performance of mobile connectivity across 165 countries or 99% of the global population since the year 2014 (Global System for Mobile Communications (GSMA), 2019).




The developed/developing country classifications are based on the UN M49, see: http://www.itu.int/en/ITU-D/Statistics/Pages/definitions/regions.aspx.html Note: * Estimate
Source: ITU World Telecommunication /ICT Indicators database (2019)

Figure 2.1 Percentage of households with Internet access by level of development, 2002-2019

According to GSMA, until 2018, only 2.6 billion people in LMIC (Low- and Middle-Income Countries) were connected to the Internet, accounting for more than 40% of the LMIC population in the world. The four factors that affect the MCI are listed below:

1) Infrastructure: the availability of the network coverage.

2) Affordability: the ability people to sustain the service provider charges.

3) Consumer readiness: the awareness and the skills needed to use the technology.

4) Content and Services: the availability of the secure content and relevant services to the local population.

### 2.1.3 The Growing Trend of Mobile Users

Every year, mobile devices with improved capabilities and intelligence are introduced to the market. According to Cisco (2019), there will be 1.5 mobile devices per capita or 12.3 billion mobile-connected devices by 2022. The rapid growing number of mobile devices in the market is among the factors driving innovation in cellular communication technology. The 2G cellular technology has evolved to 5G with higher bandwidth, broader coverage, and ultra-low latency (Jiang & Liu, 2017). However, cellular communication technologies required a robust and stable backbone infra-structure with an intermediary agent at a high cost.

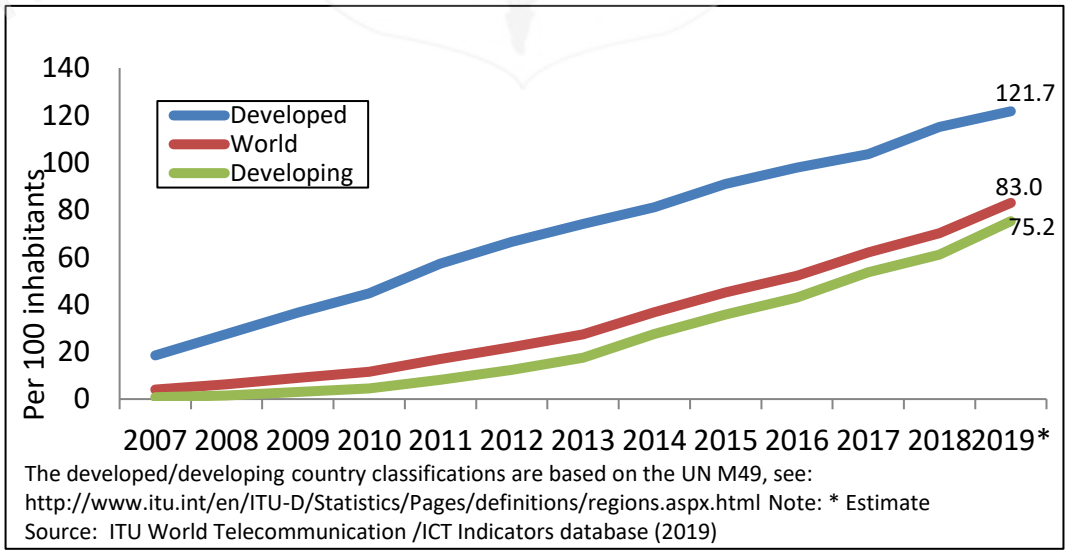



Figure 2.2 Active mobile-broadband subscriptions per 100 inhabitants, 2007-2019

Based on a consumer survey reported by Accenture(2012), “mobile Internet usage continues to be on a sharp upward trajectory”. Referring to Figure 2.2, the number of active mobile-broadband subscriptions is steadily increasing every year (International Telecommunication Union, 2019). The activities most commonly performed by mobile device owners include email, text messaging, blogging, searching for information on the Internet, logging into social networking sites, as well as watching videos or movies (ExactTarget, 2014; Osman et al., 2012).

Another survey was conducted on 3,469 random respondents in Malaysia (Malaysian Communications and Multimedia Commission, 2016). The results show that approximately 89.4% of smartphone users access the Internet on their devices. Furthermore, text communication and surfing social networking sites were the most common activities for Internet users. Mobile broadband subscribers in Malaysia increased from 64.3% in 2013 to 87.3% in 2015 (Malaysian Communications and Multimedia Commission, 2017).

Cisco (2019) has reported that the offload traffic generated by mobile devices connecting to fixed networks via Wi-Fi will surpass cellular traffic from mobile devices by 2022. Public Wi-Fi hotspots are expected to grow from 124 million in 2017 to 549 million by 2022, and Wi-Fi homespots (hotspots at home) are expected to increase from 115 million in 2017 to 532 million by 2022.

The above-mentioned statistical mobile usage phenomena can be summarized into three points. First, smartphones with more functionalities than ordinary cellphones have become essential devices, serving not only as communication tools but also for purposes such as social networking, education, e-commerce, and entertainment. Second, the Internet has become an essential network for most mobile users to conduct online activities. Third, the usage of wireless technology, especially Wi-Fi, has been on a growing trend as the primary medium for mobile communication networks.

However, there is no guarantee that the mobile network, which relies on static infrastructure as the backbone, will always be available due to unexpected and

challenging occasions, as elaborated in the previous section. Hence, an alternative network is required, especially during critical situations.

### 2.1.4 The Functional Model of the WCN

Due to the increasing usage trend of mobile devices such as smartphones, tablets, or phablets, researchers have been exploiting the networking functionalities of these devices to form alternative networks (Bellavista et al. 2010; Gardner-Stephen et al. 2012; Camps-Mur et al. 2013; Han et al. 2014; Liu et al. 2016). According to Bellavista (2014), achieving an alternative network with inter-network connectivity at all times is possible by maximizing the networking functionalities of underutilized mobile devices. The alternative network requires an innovative model to form the Wireless Community Network (WCN) so that mobile users can share information and resources in the ubiquitous environment. These resources come in various forms, such as data, files, internet access, storage, processing power, and other shareable resources. To establish the WCN, every mobile device in the network should be able to connect to at least two other devices either serially or in parallel. This function is known as a bridge or proxy.

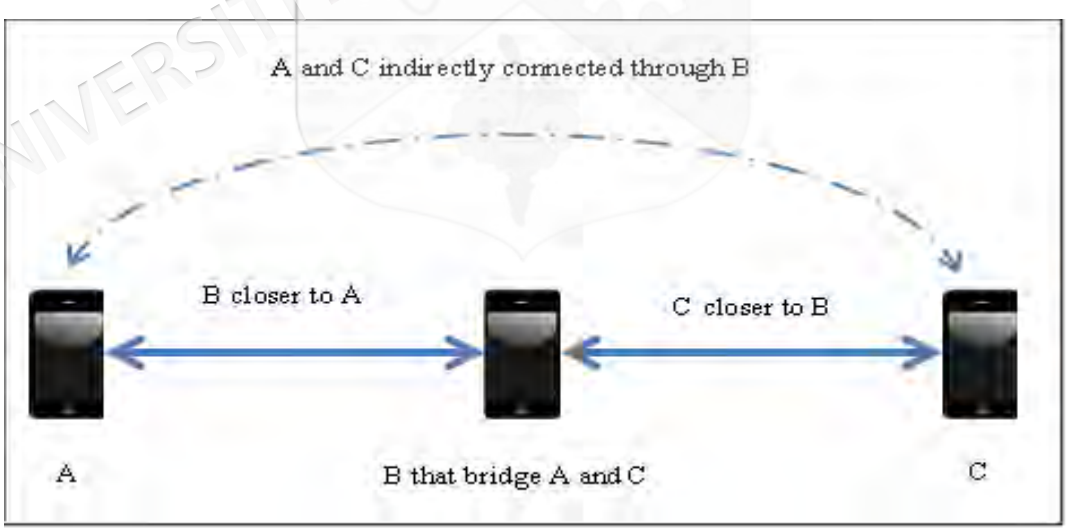



Figure 2.3 With bridging function, mobile device B will be the bridge of connectivity for mobile device A and mobile device C.

Figure 2.3 demonstrates a sample arrangement where a mobile device B is situated between mobile devices A and C. Since devices A and C are far away from

each other, their wireless signals cannot reach for direct data exchange. In this situation, device B, with the bridging function, plays the role of a proxy. Through device B, the data from device C will be transferred to device A and vice versa. With the bridging function, the devices in the WCN form a mesh network.

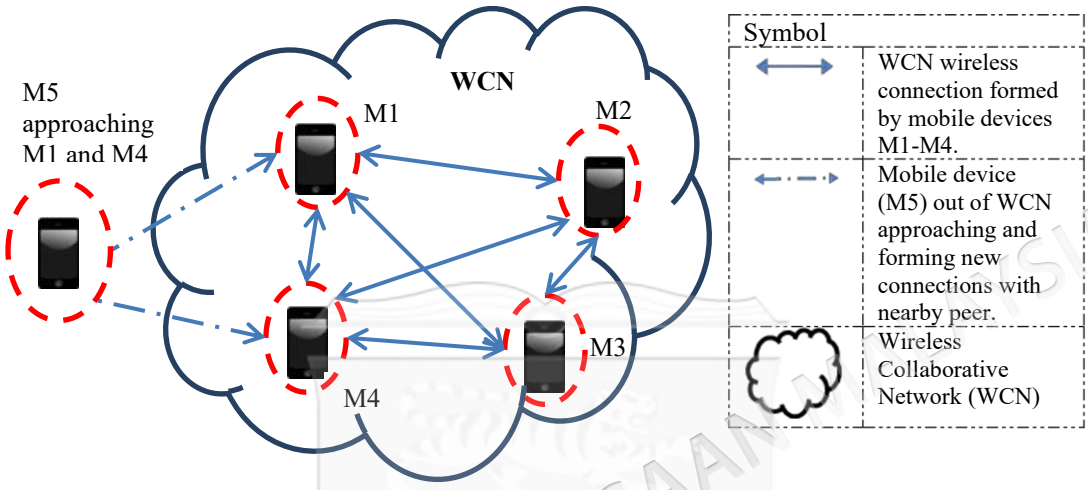



Figure 2.4 The WCN formed by smartphones (M1-M5) with bridging function.

Figure 2.4 depicts the mesh WCN formed by smartphones M1 to M5, all equipped with the bridging function. The smartphones M5, when approaching M1 and M4, become part of the WCN when the signal is strong enough.Due to end-user demand for resource sharing, the mobile operating system (OS) developer introduced the Wi-Fi tethering or Wi-Fi hotspot program. This program was created to enable a mobile device to act as a wireless tether or ad-hoc access point, allowing it to exchange data or share cellular Internet traffic with other mobile devices via the Wi-Fi interface. In the tethering process, one device takes on the role of the host, while other devices act as clients. The host serves as the centralized server, coordinating the data flow between the host-client and client-client. Referring to Figure 2.3, device B will serve as the host, while devices A and C will be the clients. With the implementation of the Wi-Fi tethering program, a small cluster of independent WCN can be formed by a group of mobile devices. However, a mobile device running the Wi-Fi tethering program (host) can only support connections with a limited number of other mobile devices (clients) at a time. Additionally, when the Wi-Fi interface of a device is in use, it cannot be used for another connection, which restricts the data transfer ability to more than one or two hops. Consequently, this limitation also restricts the range of

the communication network (Gardner-Stephen & Palaniswamy, 2011; Liu et al., 2016; Wang et al., 2016).

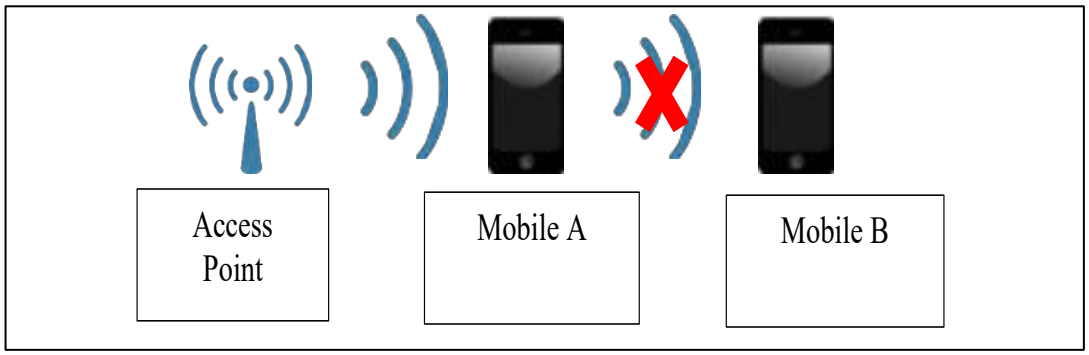



Figure 2.5 A Mobile Device Connected to an Access Point is Unable to be Connected to Mobile Device B through Wi-Fi.

Figure 2.5 demonstrates the scenario in which a mobile device A, connected to an access point (tethering device) through the Wi-Fi interface, will not be able to become a hotspot for other devices, such as mobile device B that is nearby device A (Wang et al., 2016). Due to this limitation, forming a WCN with Wi-Fi tethering is not possible by default.

Gardner-Stephen & Palaniswamy (2011) have tried to utilize the built-in tethering program on Android smartphones to form the WCN by modifying the system codes of rooted devices. Despite the suggested approach working, the solution is undesirable as it requires rooting the mobile devices. Rooting is a process of gaining complete control over the device's OS. Generally, rooting is not supported by the device manufacturer, and some rooted devices will automatically void the warranty provided by the manufacturer.

Without rooting the device, Raj et al. (2014) and Han et al. (2014) have proposed algorithm of role switching. By setting an interval time, a device's role will be switched interchangeably between tethering (as a host) and connecting (as a client) to the neighboring host. However, more energy and time are consumed during the role switching state. Furthermore, communication collisions might occur during role switching, as reported by Liu et al. (2016).

Besides the Wi-Fi Tethering program, the WFD protocol was introduced to allow direct connections between mobile devices over a Wi-Fi network for communication and data exchange operations. The protocol was formerly known as Wi-Fi P2P certified by Wi-Fi Alliance. WFD is a type of wireless mode that builds on top of the Wi-Fi infrastructure mode (Wi-Fi Alliance, 2016). However, unlike the common Wi-Fi infrastructure mode, WFD mode does not require a rigid Access Point (AP) fixed by the end user. When the protocol is activated, the participating devices are registered to a P2P group with a randomly generated group name. One of the devices will be appointed as the P2P group owner (P2P GO) role, which will act as the software Access Point (Soft AP). The remaining devices that participated in the P2P group will act as the P2P client. With minimal user action, the WFD protocol has simplified the data exchange process. However, the protocol only supports data exchange among the devices within the same P2P group. Furthermore, the process of data exchange between devices is coordinated by the P2P GO, which forms a star topology and not a full mesh topology. The mesh topology is required in the WCN so that the devices could interact with each other directly. These limitations have restricted the protocol from forming the WCN by default. Despite the limitations, it is noticeable that ongoing research works have adopted the technology to form a robust WCN ( Conti et al. 2013; Jung et al. 2014; Liu et al. 2016; Wang et al. 2016; Khan et al. 2017; Alami et al. 2017; C. Casetti et al. 2017; C. E. Casetti et al. 2018; Felice et al. n.d.; Links et al. 2019; Sunil et al. 2020). This shows that researchers strongly believe that the existing wireless interfaces available in a smartphone are still under-utilized.

## 2.2 WIRELESS TECHNOLOGY

In the context of communication networking, wireless technologies are used for data exchange, information sharing, and communication between two or more end devices over a distance wirelessly. The transmission of data between the end devices is achieved with an electromagnetic wave's transmission medium, such as radio waves, microwaves, and infrared radiation. Wireless technology can be found in most mobile devices, such as smartphones, laptops, tablets, smartwatches, wireless speakers, headphones, and many others. Examples of wireless technology include Wi-Fi,

Bluetooth, Zigbee, NFC, and RFID. The goal of this research is to propose an alternate communication model by exploiting the wireless technology available in smartphones. Therefore, the following section focuses on the main wireless technologies available in smartphones.

### 2.2.1 Bluetooth

This section briefly introduces Bluetooth technology (IEEE 802.15.1) and its implementations in the Android platform. Three protocol documents related to Bluetooth technology were studied to understand the underlying operations of the technology (Bluetooth Special Interest Group, 2014, 2021; Camp, 2000).

Bluetooth is a wireless communication technology that allows devices to exchange data over short distances. It is mainly used for short-range signal transmission in a small-scale network like a Personal Area Network (PAN). Bluetooth can be found in a wide range of devices, from personal devices like speakers, headphones, and digital watches to household appliances like TVs, radios, and air conditioning systems.

The first version of Bluetooth, Bluetooth 1.0, was publicly announced by the Bluetooth Special Interest Group (SIG) in 1999. At the point of this writing, the technology has evolved to the latest version 5.3 (Bluetooth Special Interest Group, 2021) . The signals of Bluetooth operate in the 2.4 GHz spectrum of the Industrial Scientific and Medical (ISM) band. Bluetooth uses Spread Spectrum modulation technique known as Frequency Hopping Spread Spectrum (FHSS), which continuously changes the frequency of the transmission in a random pattern. This technology is designed to reduce interference and improve communication reliability and quality. FHSS of Bluetooth divides the 2.4～2.485 GHz frequency band into 79 1 MHz bandwidth channels and hops 1,600 times per second. In addition to Spread Spectrum technology, Bluetooth also uses Adaptive Frequency Hopping (AFH), which allows Bluetooth devices to avoid frequency channels that are congested or have high levels of interference.

The working parameters for Bluetooth communication depend on the version of the protocol. This is because different versions emphasize different operating characteristics or requirements. As reported in Bluetooth Special Interest Group (2014), the data transfer rate for Bluetooth ranged from 721.2 kbps (Basic Rate) to 54 Mbps (Enhanced Data Rate high-speed operation). Similarly, the transmission distance of the Bluetooth protocol depends on the protocol's version, with a general working range falling between 10 meters to 240 meters. A basic Bluetooth network structure is known as a piconet. A piconet is formed with at least one master (central) node and one slave (peripheral) node. The maximum number of slave nodes per piconet is seven. The device that initiates the communication will be assigned the master role.

Based on the literature review, researchers exploited Bluetooth technology to form a MANET. Gardner-Stephen et al. (2012) and Wang et al. (2016) adopted a software-centric approach and developed an application that formed the MANET with a combination of Wi-Fi and Bluetooth technologies. Bellavista et al. (2010) and Dubois et al. (2015) developed middleware for heterogenous connections between Wi-Fi and Bluetooth technologies. It is noticed that Bluetooth is a promising transmission technology to form a WCN; however, the limited transmission speed and distance of Bluetooth, as compared to Wi-Fi, limit the efficiency of the operations in the WCN. Therefore, Bluetooth is not the primary consideration in this research.

The Android platform provides support for the Bluetooth network stack to manage data exchange with Bluetooth technology [1]. The Android Bluetooth stack is built on top of the Bluetooth Core Specification and implements the profiles and protocols defined by the Bluetooth Special Interest Group (SIG). It provides APIs for developers to create Bluetooth applications and enables features such as Bluetooth pairing, file transfers, and wireless audio. Programmatically, using the Bluetooth API includes the following steps:

---

[1] Bluetooth overview. (n.d.). Android Developers. https://developer.android.com/guide/topics/connectivity/bluetooth

1) Add the BLUETOOTH and BLUETOOTH_ADMIN permissions in the AndroidManifest file.

2) Get a reference to the BluetoothAdapter class's object.

3) If the Bluetooth interface is available and enabled, scan for available devices with Bluetooth enabled.

4) Manage the Bluetooth intent (eg. ACTION_FOUND) received by a BroadcastReceiver class's object registered.

5) Finally, connect to the targeted Bluetooth device (typically with socket created) for the remaining operation such as service discovery, audio streaming, or file transfer.

### 2.2.2 Near Field Communication (NFC)

Besides Bluetooth, another wireless technology that is becoming more common in smartphones today is the NFC protocol. NFC stands for Near Field Communication Technology. The technology provides secure and fast data exchange capability between two devices in proximity, typically just a few centimeters apart. NFC devices operate in the 13.56 MHz spectrum and offer various data transfer rates, depending on the version of NFC implemented. The initial versions of NFC, such as 1.0, 2.0, and 3.0, have data transfer rates of 106 kbps, 212 kbps, and 424 kbps, respectively (Sunny & Shobha, 2016). However, it is essential to note that with the advent of newer NFC versions, such as NFC 4.0 and NFC 5.0, the data transfer rates have substantially increased to reach up to 424 Mbps, making it significantly faster and more efficient for transferring larger files and data-intensive tasks.

Like Bluetooth, the NFC protocol also uses FHSS modulation technique to transmit data. The frequency of the NFC signal changes rapidly over time, spreading the data transmission over the 13.56 MHz range. NFC is a protocol that utilizes the mechanism of magnetic induction for data transfer between two devices. A source device generates a magnetic field by passing a small electric current through the NFC facility. The magnetic field is then transmitted to the target receiver. The target receiver senses the magnetic field and converts it back into electrical pulses to be

processed. When two NFC-enabled devices are brought within the supported proximity, their magnetic fields interact, allowing data to be transferred between the two devices.

When operating, the NFC device can be set in active or passive mode. Active devices generate magnetic waves and have their own power source. A passive device is powered by the magnetic field sent from another active device. Unlike Bluetooth, which supports multiple device communication in a network, the NFC protocol only supports point-to-point transactions between two devices at a time. The applications of NFC in smartphones have grown exponentially over the years. One of the most prevalent uses is mobile payments, where users can securely make contactless transactions by tapping their smartphones on compatible payment terminals. Additionally, NFC enables seamless data sharing between devices, allowing users to share files, contacts, and even URLs by simply touching two NFC-enabled devices together. Furthermore, NFC is commonly used for device pairing and configuration, making it convenient for users to connect their smartphones with other NFC-enabled gadgets, such as wireless speakers or headphones, without the need for complex setup processes (Motlagh, 2015).

In comparison to Bluetooth, NFC has distinct advantages. While Bluetooth is ideal for more extended range communication, typically up to 10 meters, NFC's limited range of a few centimeters offers enhanced security, as it requires devices to be in proximity for communication. This characteristic makes NFC an excellent choice for secure transactions and data sharing, where a higher level of physical proximity authentication is desired. NFC allows for quick and seamless communication between devices by simply bringing them close together. Furthermore, the low power consumption helps prolong the battery life of NFC-enabled devices. However, while newer versions of NFC (e.g., NFC 4.0 and 5.0) have improved data transfer rates, they still lag other wireless technologies like Wi-Fi and Bluetooth when it comes to transferring large files or handling data-intensive tasks.

Like Bluetooth protocol, the Android platform also provides support for the Android NFC stack for data exchange between devices in very close proximity[2]. The framework provides API for developers to produce NFC-based applications and services to interact with NFC tags and other NFC-enabled devices. The API covers the functionality to interact with higher-level NFC components, manage NFC transactions (NFC controller), parse NFC Data Exchange Format (NDEF) data, scan, and read NFC tags, and transfer NDEF messages. Programmatically, using the NFC API includes the following steps:

1) Add the NFC permissions in the AndroidManifest file.
2) Get a reference to the NfcAdapter class's object.
3) Check if the NFC interface is available and enabled.
4) Manage the NFC intent (eg. ACTION_NDEF_DISCOVERED) received by a BroadcastReceiver class's object registered.
5) Finally, read or write data to the target peripheral.

### 2.2.3 Wi-Fi

Wi-Fi is another common wireless technology found in all smartphones. Compared to Bluetooth and NFC, which allow communication over short distances, Wi-Fi supports longer range and higher signal transmission speeds (Sunny & Shobha, 2016; Vidakis et al., 2020). Wi-Fi is a standard term for the wireless technology managed by the Institute of Electrical and Electronics Engineers (IEEE). The standard code 802.11 was designated by IEEE in 1997 for the wireless architecture. The first version of the documented standard was labelled as 802.11-1997. The main goal of the 802.11 standard is to provide wireless connectivity for stationed and mobile stations within a compound area.

---

2 NFC basics. (n.d.). Android Developers. https://developer.android.com/guide/topics/connectivity/nfc/nfc

The Wi-Fi technology has been enhanced since the release of the first version. Initially, the first version of the 802.11 standard was limited to communication over the 2.4Ghz band at a maximum bandwidth of only 2 Mbps. Subsequent releases of the technology have been upgraded to support faster transfer rates or wider coverage ranges (Pahlavan & Krishnamurthy, 2021). Whenever a new version is released, it will be labeled with the code suffix "802.11." For example, as of the time of this writing, the current proposed version is 802.11ay-2021, which offers "Enhanced Throughput for Operation in License-exempt Bands above 45 GHz" and allows throughput ranges from 20 to 40 Gbps over up to 500 meters under optimal conditions (minimal interference). However, this protocol is not yet available for smartphones at this moment[3]. However, this protocol is not yet available for the smartphones at this moment.

**a. Main Protocols for Wi-Fi Technology**

Wi-Fi Technology is built on several sub-protocols and technologies that work together to form a wireless network. The important sub-protocols used in Wi-Fi include the Media Access Control (MAC) protocol, Physical Layer (PHY) protocol, Data Link Layer (DLL) protocol, Service Set Identifier (SSID) protocol, Internet Protocol (IP), Wi-Fi Protected Access (WPA&WPA2) protocols, and Advanced Encryption Standard (AES) protocol (IEEE Computer Society, 2016, 2020).

The MAC protocol is adopted to regulate the flow of data between end devices. Each network interface is assigned a unique MAC address, ensuring that the transmission of data packets reaches the targeted destination accordingly. The PHY protocol is the lowest layer of the 802.11 protocol stack. This protocol defines the physical characteristics of the networking interface, such as frequency, modulation, and transmission rate. For example, in the 802.11n standard, the physical characteristics (of the PHY layer) of the networking interface are fixed at 2.4 and 5 GHz bands, with speeds of up to 600 Mbps.

---

[3] IEEE GET ProgramTM. (n.d.). IEEE Xplore Browse Standards. https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68

The DLL protocol enables basic data transfer services between devices (known as stations – STA, according to the protocol's documentation). These transfer services include parsing the data into formalized frames and managing the transmission and reception of data frames between devices. This protocol also ensures data transmission reliability by detecting and correcting errors that may occur during transmission. The three basic frames established in the DLL protocol include management frames that control the basic operations of the Wi-Fi network (discovery, association, disassociation, and authentication), control frames that support the management of the wireless network (requesting the transmission of data or acknowledgment), and data frames that carry user data over the network.

The SSID protocol is used to identify a specific Wi-Fi network. It is a unique identifier broadcasted by access points so that interested clients can request a connection to the access point. SSID is often used in conjunction with security protocols like WPA, WPA2 for access control. Referring to the technical book by Parziale et al. (2006), the IP protocol establishes standardized guidelines for wireless devices to follow when transmitting data over the Internet. Both IPv4 and IPv6 are fundamental addressing protocols based on the IP protocol. These addressing protocols offer networking functionality such as addressing and routing, data transmission, end-to-end communication, and Internet connectivity. (IEEE Computer Society, 2016, 2020).

WPA, WPA2, and AES protocols provide a secure encryption mechanism for data transmitted over the Wi-Fi network, enhancing Wi-Fi security. According to Ramezanpour et al. (2023) the emergence of next-generation wireless networks, such as 5G and 6G, with a massive number of mobile and IoT devices has raised concerns about spectrum scarcity and quality of experience (QoE). The authors reported critical security vulnerabilities unique to 5G/WiFi coexistence, such as hidden node issues that can lead to service blocking, rogue base-station deployment, and eavesdropping. The potential security threats include spectrum hijack, service degradation, and pseudo man-in-the-middle attacks. Therefore, it is important to consider the vulnerabilities of Wi-Fi technology when designing the new model for WCN. One of the potential security measures is to implement encryption algorithms such as WPA.

WPA offers stronger security than WEP. At the time of writing, there are three versions of the WPA encryption mechanism available, namely WPA, WPA2, and WPA3, respectively. Due to the weaknesses of the previous versions, WPA3 was introduced (Alhamry & Alomary, 2022; Moissinac et al., 2021). However, support for WPA3 may vary depending on the age and type of Wi-Fi devices and routers being used. Some newer devices are WPA3-compatible, while others may only support WPA2.

**b. Operational of Wi-Fi Technology**

The Wi-Fi (802.11) technology works in several radio frequency (RF) bands, with the most widely used being 2.4 GHz and 5 GHz. Additionally, other bands are available, such as 60 GHz for directional multi-gigabit (DMG) networks. In recent years, the Wi-Fi Alliance has included the 6 GHz band as a standard for consumer products. However, it is still relatively new in the industry, and not many devices support the 6 GHz band yet (Wi-Fi Alliance, 2016).

For each frequency band, channels are divided to support the concurrent receiving and transmitting of data between devices. This channel division process is known as channelization. The channel width controls how broad the signal is for transmitting data and is measured in MHz units. For example, the 2.4 GHz band in the 802.11g protocol can be divided into 11 channels of 20 MHz width (Intel Corporation, n.d.). With different channel allocations, Wi-Fi devices can avoid interfering with each other and achieve the best data transmission rates. The exact number of channels and their frequency ranges can vary depending on the country or region, as regulatory agencies may assign different frequency ranges for Wi-Fi use.

Like other wireless technologies, Wi-Fi technology utilizes modulation techniques. The most used modulation technique in Wi-Fi is Orthogonal Frequency Division Multiple (OFDM). OFDM is used in the PHY layer and plays a crucial role in Wi-Fi technology. In this technique, data is divided into multiple parallel subcarriers (channels), enabling the transmission of large amounts of data over a wide frequency range, resulting in high data rates and robust performance (IEEE Computer Society, 2020).

A Wi-Fi network works in either infrastructure or ad-hoc mode. In infrastructure mode, an access point (AP) is needed as the communication center for all participating network devices. On the other hand, ad-hoc mode does not require an AP; the network devices communicate directly with each other. The infrastructure mode is known as the Basic Service Set (BSS), and the ad-hoc mode is known as the Independent Basic Service Set (IBSS)[4].

A typical use-case scenario of the Wi-Fi infrastructure mode would be a mobile device with the tethering function enabled, acting as an AP to share the Internet connection with another smartphone (Niranjan et al., 2015). In ad-hoc mode, a mobile device with ad-hoc mode enabled can connect directly to another wireless device over the Wi-Fi medium for data exchange. However, a Wi-Fi connection in ad hoc mode is platform dependent. For example, not every Android smartphone provides the functionality of ad-hoc mode for peer-to-peer communication (Lee & Sulaiman, 2019). Despite that, the Android system introduced WFD as an alternative for peer-to-peer communication.

There are several literature pieces reporting on the operational procedures of Wi-Fi technology. Some include technical reports on Wi-Fi (IEEE Computer Society, 2016, 2020), while others consist of review articles by researchers about the technology (Cao et al., 2022; Lee & Sulaiman, 2019). Based on the literature, whenever a device (client) or application needs to join a Wi-Fi network, the device must detect the availability of the network. The mechanism for detecting the nearby network is known as scanning. During the scanning of the nearby networks, the device will collect a list of available network identifiers. A network identifier is the name of a network defined and broadcasted by the master device (e.g., AP). The network identifier name is formally known as the Service Set Identifier (SSID).

---

[4] Wireless Hacks. (n.d.). O'Reilly Online Learning. https://www.oreilly.com/library/view/wireless-hacks/0596005598/ch01s13.html

As stated in IEEE Computer Society (2020), there are two types of scanning modes available in Wi-Fi, named active and passive scanning. During active scanning, the client device discovers the available SSID by broadcasting a probe request frame and continuously listening to the probe response frame returned by the master device. Probe request and response frames are types of management frames used for device or SSID discovery, which occurs before the connection and association process begins. On the other hand, during passive scanning, the client device is set in a state of listening for beacon frames periodically broadcasted by the master device. A beacon frame is a type of management frame that is transmitted by the master device to announce its presence and provide information about the network to the wireless devices in range. Probe request, probe response, and beacon frames contain basic information such as SSID, MAC address of the master device, supported data rates, and channel information.

In recent years, researchers have been using Wi-Fi's probe frames for various purposes, primarily related to network analysis, performance optimization, and security research. For instance, Rusca et al. (2023) focused their research on non-intrusive and independent datasets for user localization, introducing the concept of bounded locations to infer users' trajectories over time. The study provides insights into localization errors and proposes a new method for estimating user trajectories.

Hou et al. (2023) and Wan et al. (2023) utilized Wi-Fi probe data to study human flow. By capturing and analyzing the probe messages, the proposed model generated visualizations of the movement and patterns of pedestrians in an open area. Gebru et al., (2022) and Jundee et al. (2023) employed commercial sensors and devices like Raspberry PIs equipped with a Wi-Fi interface to capture Wi-Fi probe data.

Chang et al. (2023) and Jundee et al. (2023) demonstrated the potential of Wi-Fi probe data for analyzing mobility and travel behavior in different scenarios, such as campus networks and public transit. The literature has highlighted the increasing importance of Wi-Fi probe data in studying human behavior and movement patterns

in various settings, ranging from urban environments and shopping malls to public transit systems.

Additionally, the literature provides insights into the potential of the probe, or at least similar types of management frames, to play additional roles in data exchange. Hence, it is observed that the Wi-Fi protocol can be exploited to form a new communication model in WCN.

According to the Request for Comments documents RFC 2544 (S. Bradner & McQuaid, 1999) and RFC1242 (S. Bradner, 1991) by the Internet Engineering Task Force (IETF), the parameters for measuring the performance and characteristics of a general network include throughput, latency, frame loss rate, back-to-back frames, and system recovery. In addition to the measurement parameters listed in the RFCs, there are other characteristics relevant to a wireless network, such as goodput, jitter, and signal strength.

Signal strength is the measurement of the power of a wireless signal between two devices; the closer the devices are, the stronger the signal strength. Typically, the signal strength is represented as a negative value. The lower the negative value, the stronger the signal; the higher the negative value, the weaker the signal. For example, a signal strength of -30 dBm indicates the maximum signal strength, likely when the devices are positioned next to each other. On the other hand, -90 dBm is considered a weak signal reading.

Wi-Fi parameters play a crucial role in research related to wireless communication, networking, and various applications. These parameters provide valuable insights and data essential for understanding, optimizing, and improving Wi-Fi networks and related technologies. For example, Solodo & Malarić (2013) demonstrated measurements of Wi-Fi signal strength and its impact on download/upload speeds. The authors reported various factors influencing signal strength, including distance, interference from devices, and walls. Yasser et al., (2021) emphasized the importance of selecting the appropriate group owner (GO) to enhance the quality of service (QoS) and reliability of software over WFD technology. The

criteria for choosing the GO and backup devices are based on various parameters, such as battery power, received signal strength (RSSI), and signal-to-noise ratio (SNR), to optimize service performance. A summary of the definitions and units of measurement for each parameter is listed in Table 2.1.

Table 2.1 Definition for the measurement parameters in networking.

| Measurement parameters | Definition (based on RFC 1242 & RFC2544) | Unit |
|---|---|---|
| Throughput | "The maximum rate at which none of the offered frames are dropped by the device." (S. Bradner, 1991) | bits per second (bps)/ Bytes per second (Bp |
| Goodput | "The amount of data delivered per unit time to the application layer."(Mao et al., 2017) | bits per second (bps)/ Bytes per second (Bps) |
| Latency | "For store and forward devices: The time interval starting when the last bit of the input frame reaches the input port and ending when the first bit of the output frame is seen on the output port." (S. Bradner, 1991)<br>"For bit forwarding devices: The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the start of the first bit of the output frame is seen on the output port." (S. Bradner, 1991) | time (eg. Milliseconds) |
| Jitter | "The deviation in latency of a packet stream"(Schoonwinkel, 2016) | time (eg. Milliseconds) |
| Frame Loss Rate | "Percentage of frames that should have been forwarded by a network device under steady state (constant) load that were not forwarded due to lack of resources." (S. Bradner, 1991) | percentage |
| Back-to-back frames | "Fixed length frames presented at a rate such that there is the minimum legal separation for a given medium between frames over a short to medium period of time, starting from an idle state." (S. Bradner, 1991) | number of frames |
| System recovery | "The speed at which a device under test recovers from an overload condition."(S. Bradner & McQuaid, 1999) | time (eg. Milliseconds) |
| Signal strength | The measurement of the power of a wireless signal between two devices. | dBm / percentage |

As with Bluetooth and NFC, the Android platform also provides support for the Wi-Fi stack for data exchange over the wireless networ[5]. The framework provides an API for developers to perform wireless network operations such as configuration, scanning, discovery, and connection to a Wi-Fi network with or without Internet connectivity. Programmatically, using the Wi-Fi API includes some or all of the following steps:

1) Add the permissions (ACCESS_WIFI_STATE and CHANGE_WIFI_STATE) into the AndroidManifest file.
2) Get an instance of the WiFiManager class.
3) Check if the Wi-Fi interface is available and enabled.
4) Scan for available Wi-Fi networks.
5) Connect to a Wi-Fi network and then performing individual networking operation
6) Finally, disconnect from Wi-Fi network.

### 2.2.4 Comparison of the Wireless Technology

This section summarizes a review and comparison of the three wireless technologies discussed in the previous section. The comparison is based on several parameters, including operating frequency (GHz), transmission rate (Mbps or Kbps), and transmission distance. It is important to highlight that there are other wireless technologies. However, because the scope of this research is limited to smartphones, the comparison is confined to the three most common wireless technologies found in a smartphone. The summary of the comparison is depicted in Table 2.2. Among the three, NFC is designed for short-range communication between devices. It has the lowest transmission rate and the shortest transmission range. The operation of NFC only supports a maximum of two devices per session for data exchange between the two devices.

---

[5] Wi-Fi infrastructure overview. (n.d.). Android Developers. https://developer.android.com/guide/topics/connectivity/wifi-infrastructure

Table 2.2 Comparison of different wireless technologies in a smartphone.

| | Bluetooth | NFC | Wi-Fi |
|---|---|---|---|
| Operating Frequency | 2.4 GHz | 13.56 GHz | 2.4 GHz, 5GHz |
| Transmission rate (varies by version) | 732.2 Kbps ~ 50 Mbps | 106 Kbps, 212 Kbps, or 424 Kbps | 1~2400 Mbps |
| Transmission distance (varies by version) | <=10 meters | 0-20 centimeters | 20-200 meters |
| Supported nodes per host/network | 8 | 2 | 200 |

Bluetooth is designed to provide short-range data exchange and low-power consumption by the devices. It has a higher transmission rate and a longer transmission range than NFC. In a piconet, Bluetooth technology can support up to eight devices for data exchange or communication. It is usually used to form a personal area network where Bluetooth-enabled devices are located surrounding a central device or a person.

Wi-Fi is designed to provide a high-speed wireless connection over a wider area, such as a campus, home, or office. Wi-Fi has a range of 20–200 meters, although this can vary depending on the environment and the capability of the Wi-Fi module in the devices. Wi-Fi can support data transfer rates of up to 2400 Mbps, making it suitable for demanding applications such as high-definition video streaming. A Wi-Fi network can support a maximum of 200 devices. It is important to note that the working parameters of the technology are different depending on several factors, including the Wi-Fi protocol used and the capabilities of the devices.

Based on the comparison of the three wireless technologies, it is confirmed that Wi-Fi is the most suitable wireless protocol for this research, which aims to form a temporary communication network by utilizing the Wi-Fi interface of the mobile devices. This is because the technology provides the longest transmission range, the highest transmission rate, and the highest number of nodes per host or network. These properties are essential to provide the most efficient medium and optimize the store-and-forward mechanism in the development model.

## 2.3 WI-FI DIRECT (WFD)

As mentioned earlier, Wi-Fi technology supports two modes of operation: infrastructure mode and ad-hoc mode. The infrastructure mode allows multiple devices to connect to a host and form a communication network. In an infrastructure network, the host functions as the middleman that monitors the flow of traffic. On the other hand, the ad-hoc mode enables direct communication from device-to-device (D2D) without any infrastructure setting. Direct communication between devices in this mode is known as peer-to-peer architecture in networking. In this work, the device-to-device or peer-to-peer transmission mechanism is essential to realize the formation of the proposed model.

As explained in section 1.5.3, Android was chosen as the target platform for this research due to its popularity in the market and the availability of the open-source library and API of the wireless framework. Hence, this section provides an overview of the peer-to-peer connection protocol in the Android platform, which is WFD.


Figure 2.6 Different types of P2P topology formed by WFD standard.


Figure 2.7 Concurrent operation of a P2P device (middle).

WFD, formally known as Wi-Fi Peer-to-Peer and certified by the Wi-Fi Alliance, is a wireless mode that builds upon the Wi-Fi infrastructure mode (Wi-Fi Alliance, 2016). However, unlike the common infrastructure mode, the WFD mode does not require an Access Point (AP). The participating devices will negotiate to designate a device to take over the AP-like role. The device with the AP-like role is

referred to as the GO, and the devices connected to the GO are referred to as clients. The clients that connect to the GO could be legacy clients[6] or P2P clients based on the WFD standard. Figure 2.6 depicts the P2P topology formed between the GO and two clients. Concurrent operation, as shown in Figure 2.7 is possible if the GO supports multiple Wi-Fi interfaces, either physically or virtually. In the concurrent operation, the GO acts as the middle entity between two Wi-Fi groups. The first group is formed between the GO (middle device) and the AP, where the GO performs the role of a Wireless LAN Client, which connects to an AP (on the right). The second group is formed between the GO (middle device) and another client device (on the left), where the same GO performs the AP-like role and forms a connection with other clients. The concurrent ability of a P2P device allows connectivity expansion for resource sharing and transmission (Lee et al., 2018). Many research studies have utilized the WFD protocol in WCN. A detailed critical review of the WFD protocol is presented in section 2.5.

### 2.3.1 P2P Group Formation

The devices go through several stages when forming the P2P (Peer-to-Peer) group, depending on the pre-condition. The group forming stages typically include Discovery, GO negotiation, Wi-Fi Protected Setup (WPS) Provisioning, and Address Configuration. During the Group formation, the P2P devices communicate by sending network frames such as beacons, probe requests, and probe responses.

Figure 2.8 demonstrates a simplified process of forming a P2P group (Wi-Fi Alliance, 2016) Initially, the devices with the Wi-Fi P2P mode enabled enter the listen state. In the listen state, a device randomly selects one of the listen channels from the list of social channels. These social channels include channel 1, 6, or 11 for devices operating in the 2.4GHz band. The minimum period of a P2P device in the listen state

[6] A legacy client connects to the Wi-Fi Group using conventional 802.11 Wi-Fi standard, as if the client is connecting to an AP in Wi-Fi infrastructure mode.

is at least a contiguous period of 500ms every 5s, allowing other P2P devices to discover it.

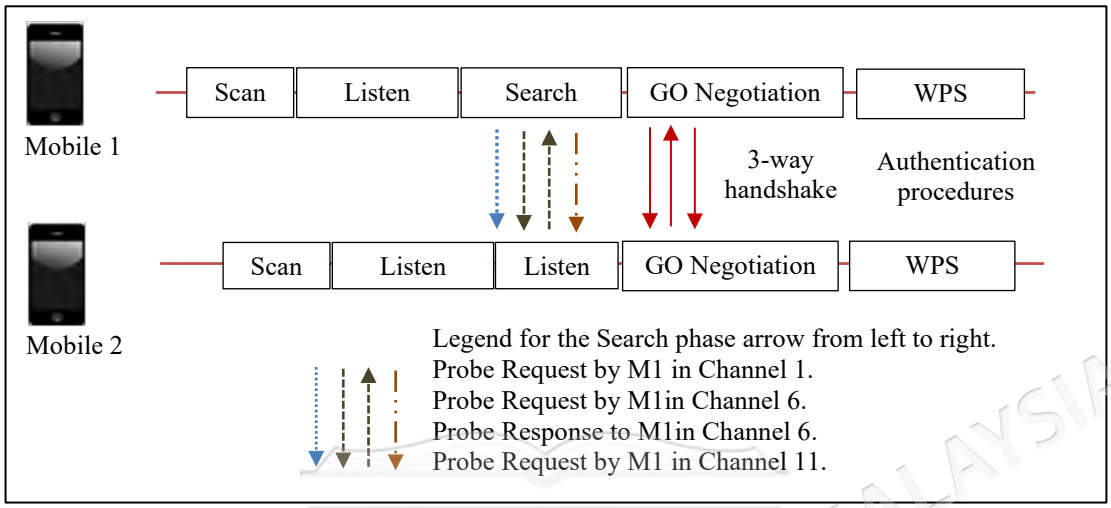



Figure 2.8 Sample P2P Group formation process.
Source: (Wi-Fi Alliance, 2016)

### 2.3.2 P2P Devices Discovery.

Khan et al. (2017) reported that P2P device discovery includes alternating between phases. During the listen state, the devices will alternate between scan and find phases. In the scan phase, the devices will scan all available channels to collect information about surrounding devices. The objective of the scan phase is to find a P2P or P2P group and fix the channel to establish a P2P group.

In the find phase, the device will alternate between the listen state and the search state at random intervals of N (an integer) every 100 TUs (Time Units) (Wi-Fi Alliance, 2016). In the find phase, the device will either wait (listen state) or send (search state) probe request or discovery beacon frames on each of the social channels. When the device discovery stage completes, the GO negotiation stage begins, followed by WPS provisioning.

### 2.3.3 GO Negotiation and WPS Provisioning.

Assume that it is the first time the P2P devices attempt to form the P2P group. The GO Negotiation stage begins after the Discovery stage has ended, and the neighboring

P2P devices have been identified. During this stage, the P2P devices will negotiate the GO role using a three-way handshake procedure. The procedure requires the transmission of GO request, response, and confirm frames (Wi-Fi Alliance, 2016).

After the GO role has been confirmed, the P2P devices will move on to the WPS Provisioning stage. WPS provisioning is compulsory in the WFD standard and normally requires minimal user intervention (e.g., pressing a button displayed on the screen). The WPS procedure utilizes WPA-2 security and uses the Advanced Encryption Standard (AES)-CCMP as the cipher and a randomly generated Pre-Shared Key (PSK) to form the authentication credential. The authenticated credential will be used by the P2P devices to skip the GO Negotiation stage and link back to the previously connected GO directly (Wi-Fi Alliance, 2016). WPS Provisioning is out of the scope of this work; therefore, no further explanation is provided in this writing.

### 2.3.4 Consumers' Products with WFD

WFD allows device-to-device communication over Wi-Fi without going through a central device, which is an AP. This technology enables Wi-Fi enabled devices to connect to each other directly, facilitating data, file, and media transfer. It has been made available in various consumer products, such as mobile phones, smart TVs, TV set-top boxes, gaming console devices, smart vehicles, and many more.

Since WFD is a standard managed by the Wi-Fi Alliance, manufacturers of consumer products with WFD technology must register their products with the Wi-Fi Alliance. As of early February 2023, the Wi-Fi Alliance has certified 29,409 products of different models with WFD technology[7]. This figure was collected from the Wi-Fi Alliance product finder website. The filtered and analyzed data are presented in Figure 2.9.

Referring to Figure 2.9, there are eight main categories of consumer products certified with WFD technology. Out of over 20,000 certified products, a total of

[7] Product Finder Results | Wi-Fi Alliance. (n.d.). https://www.wi-fi.org/product-finder-results?sort_by=certified

14,037 (approximately 47.7%) are mobile phone models equipped with WFD technology. Figure 2.10 depicts the mobile phones certified with WFD technology based on different operating systems. Out of the 14,037 certified phones of different models, 12,321 are Android-based, which accounts for around 88% of the devices certified (Wi-Fi Alliance, 2023).

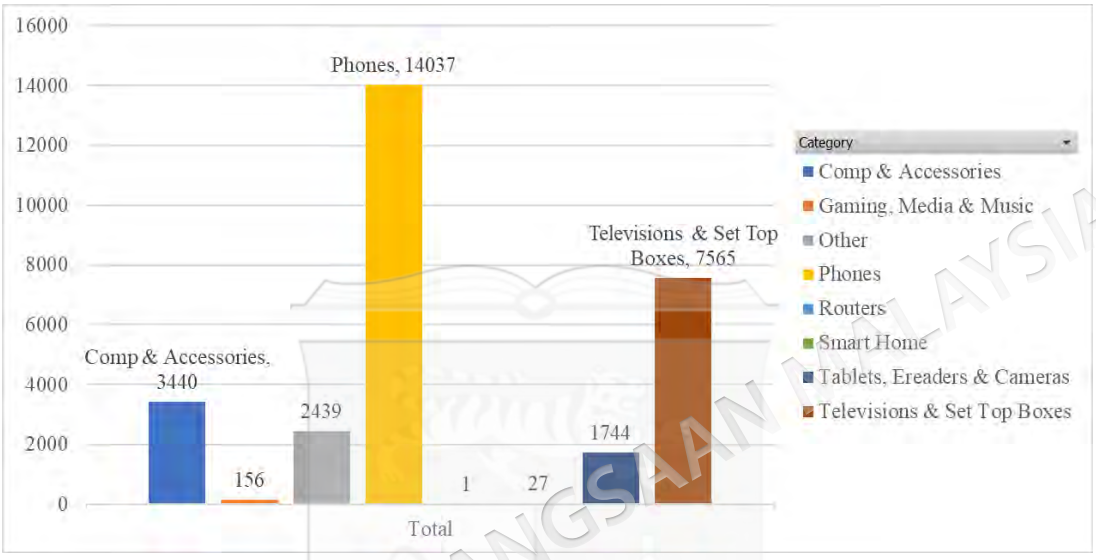



Figure 2.9 Consumer product certified with WFD technology.
Source: (Wi-Fi Alliance, 2023)

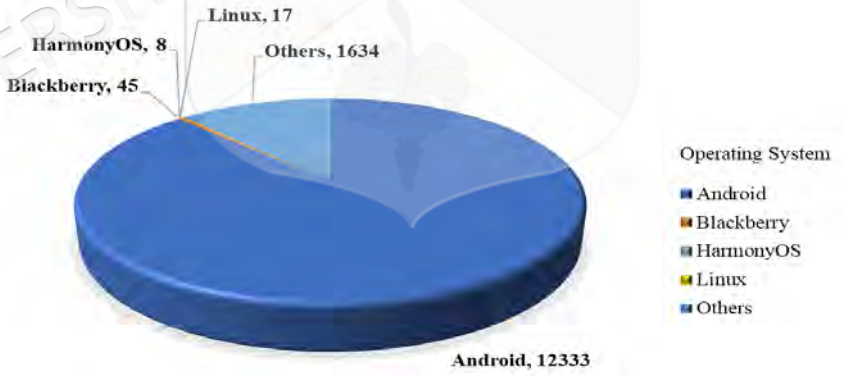



Figure 2.10 Phones certified with WFD technology, based on OS.
Source: (Wi-Fi Alliance, 2023)

In terms of Wi-Fi bands, the Wi-Fi Alliance has registered 14,073 mobile phones with various operating systems that operate in the 2.4 GHz band. In other words, 100% of the mobile phones from different models registered by the Wi-Fi

Alliance support the Wi-Fi 2.4 GHz frequency. Among these operating systems, 12,333 phone models running Android OS are registered to support 2.4 GHz.

Table 2.3 Comparison of mobile operating systems on different Wi-Fi bands.

| OS\Band | 2.4 GHz | 5 GHz | 6 GHz |
|---|---|---|---|
| Android | 12333 | 8161 | 66 |
| Blackberry | 45 | 34 | 0 |
| HarmonyOS | 8 | 8 | 0 |
| Linux | 17 | 7 | 0 |
| Others | 1634 | 994 | 0 |
| Grand Total | 14037 | 9204 | 66 |

Compared to the 2.4 GHz band, there are only 9,204 mobile phones with various operating systems that operate in the 5 GHz band. In other words, approximately 65.6% of the mobile phones from different models registered by the Wi-Fi Alliance support the Wi-Fi 5 GHz frequency. Finally, only 66, or less than 0.05%, of Android OS phone models are registered with the new 6 GHz band. The detailed breakdown in terms of mobile operating systems and the three frequency bands can be found in Table 2.3.

### 2.3.5 Advantages and Limitations of WFD

WFD enables direct communication between devices and is supported by most mobile devices with a Wi-Fi interface. Compared to other wireless technologies that support D2D communication, such as Bluetooth and NFC, WFD provides faster data transfer rates and longer-range capabilities. WFD is available on most Android devices, offering an open-source API for developers to access.

However, WFD works in GO-Client mode, meaning that one device can only act as either a (GO) or a client at a time. The WFD network can only function as a one-hop ad-hoc network, with the GO serving as the default gateway device. Although APIs are available to provide flexibility for developers to design their own group formation implementations, the protocol's specification provides limited information on dealing with larger groups (Wi-Fi Alliance, 2016). Consequently,

using the protocol without implementing additional procedures (e.g., group formation extension, multi-hop capability) restricts its ability to support extensive and continuous communication. To address these limitations, researchers have proposed a new framework, scheme, and model to enhance communication in the MANET or DTN research area.

## 2.4 PAST RESEARCH OF OPPNET

This section discusses past research by other researchers in the OppNet research area. As described in section 2.1.1, this research falls under OppNet research area, which is a sub-field of MANET. Many research works have been conducted to explore and design different approaches to form the OppNet. While proposals, designs, software, framework, architecture, or protocol have been developed to support network connectivity, some researchers have explored the possibility of using robots or invented devices to act as bridges or gateways for maintaining network connectivity.

### 2.4.1 OppNet by Hardware

This section discusses the work done on integrating networking hardware for wireless network connectivity. The hardware covers the invention of network devices, communication towers, and robots.

Stevens et al. (2014) designed a prototype of a portable telecommunication tower claimed to be low cost, portable, and openly available for replication. Adeyeye & Gardner-Stephen (2011) presented the communication infrastructure designed using the device they invented, known as mesh potato. Junio & Chavez (2018) introduced a framework for offline communication by using routers and a solar-powered server as the backbone of the communication framework. The router was used to form a local area network and act as a gateway for data dissemination between the mobile devices and the server. The server operated as the central point for managing user access to the framework and for message storage. A proprietary mobile application was built and installed on end-user devices to assist the users in communication operations.

Inspired by cross-field research, some researchers use robots to broaden and optimize wireless network connectivity. In the research project report, Reich et al. (2008) introduced the prototype of a mobile network named Spreadable Connected Autonomic Networks (SCAN), where any robot with the SCAN algorithm integrated would automatically move to maintain connectivity and, at the same time, function as a wireless access point. Liemhetcharat & Veloso (2012) deployed a team of mobile robots to automatically locate and then tether to an autonomous agent. The robots shared connectivity information using the developed distributed algorithm. Other researchers utilized robotic technology to enhance wireless network connectivity (Biswas & Veloso, 2010; Ocana et al., 2005; Sit et al., 2007). Although some of the reviewed works mainly focus on expanding the robot's wireless network connectivity for localization purposes instead of wireless network connectivity expansion, the concepts and ideas remain relevant.

Based on the literature, the considerations on the hardware integration to the wireless network connectivity basically include the total setup cost, radio interference, the scalability, and the geographical location of the setup base. Some researchers claimed that the prototype designed was not expensive and deployable in a real environment (Adeyeye & Gardner-Stephen, 2011; Liemhetcharat & Veloso, 2012; Stevens et al., 2014). Despite many successful cases from the literature, where the invented device proved to work in forming and extending the wireless network, it is believed that it is not the best solution after considering the cost, complication of setting up the hardware, acceptance of the end user and local authority in approving the hardware setup. Comparing with the common and easily available portable devices such as mobile phone or tablet which could provide similar functionality, the latter is much cheaper and already freely available in almost everywhere. Therefore, this research concentrated on utilizing the end user mobile devices. The technique and idea implemented such as the algorithm of scanning and discovery in the hardware invented base on the literature reported were considered when developing the new communication model.

### 2.4.2 OppNet by Software

This section discusses the research based on designing a software system for wireless network connectivity. The term 'software system' described here includes algorithms, applications, protocols, frameworks, and architectures. Many works have been done on creating continuous wireless network connectivity by incorporating software systems into mobile devices. Some researchers classified this research area as software-defined networking (Bellavista et al., 2010; Dubois et al., 2015; Gardner-Stephen et al., 2012; Liu et al., 2016; Raj et al., 2014; Wang et al., 2016).

Bellavista & Giannelli (2009) introduced the idea of implementing middleware in mobile devices to provide multi-hop multi-path Internet connectivity. They achieved this connectivity by exploiting the JSR-82 Bluetooth API and integrated the API into the proposed architecture known as MMHC. The new architecture has been tested and demonstrated success in forming multi-hop multi-path connectivity via the Bluetooth interface. However, the research only focuses on the Bluetooth interface, and not much information about the Wi-Fi interface could be retrieved. Furthermore, it was observed that the middleware works only partially on certain operating system platforms (e.g., Ms. Windows), which contradicts the main objective of supporting connectivity for heterogeneous multi-hop clients. This limitation could be due to restrictions imposed by networking hardware manufacturers.

Continuing with their previous work, Bellavista et al. (2010) developed an enhanced middleware that supports the Wi-Fi interface. The prototype RAMP has been tested and shown success in forming a spontaneous network connection to perform data transfer. However, it should be noted that the test case used for the work involved only up to 3 hops. To draw more solid and convincing findings, further testing with more hops and devices should be included.

Raj et al. (2014b) developed an architecture named E-DARWIN for the data collection and transmission process during a disaster. The architecture utilized the Wi-Fi Tethering function of mobile devices for data delivery across the ad-hoc network. By setting an interval time, a device's role will be switched interchangeably between tethering (as a host) and connecting (as a client) to neighbour hosts.

Another group of researchers introduced an approach for MANET communication through the WFD interface of mobile devices (Liu et al., 2016). They developed a proprietary routing protocol to support the role-switching mechanism. With the role-switching mechanism, the devices can autonomously forward data to the next hop via the Wi-Fi network. However, this role-switching state may consume more energy and time, and communication collisions might occur during the process (Liu et al., 2016).

Alternatively, researchers have developed heterogeneous connection frameworks by utilizing different wireless interfaces such as Wi-Fi and Bluetooth. These frameworks aim to support data delivery across the multi-hop heterogeneous network. For example, the Serval Project team has developed a well-established core component named Serval DNA (Gardner-Stephen et al., 2012). The Serval DNA core comprises the mesh datagram protocol (MDP), voice and audio carrier protocol (VoMP), file distribution system (Rhizome), and text messaging system (MeshMS). The Serval Mesh mobile application has been published in the Android market and proven to work effectively in many on-the-field experiments (Gardner-Stephen et al., 2013). The Serval mesh application utilizes Bluetooth and Wi-Fi in infrastructure and ad-hoc mode for content sharing in the mesh network topology. However, the phone needs to be rooted for the Wi-Fi ad-hoc mode to function. As far as what was identified, the application does not support WFD. In another work, Wang et al. (2016) developed a model named BWMesh that utilizes multiple wireless interfaces. BWMesh demonstrated the possibility of forming a multi-hop connection between devices by using Wi-Fi and Bluetooth interfaces interchangeably.

Bellavista et al. (2010) and Dubois et al. (2015) adapted the middleware concept and developed an application that works on a larger scaled, heterogeneous network for content sharing. However, due to the range and performance limitations of the Bluetooth interface, there might be some restrictions on the proposed framework. Furthermore, Maia et al. (2014) reported that the percentage of message loss was expected to increase when the number of hops increased. Therefore, the implementation of BT might introduce a reduction in overall performance.

Table 2.4 Summary of the past literatures.

| Articles | Approaches | Hardware/Software centric | Wireless interface | Limitations |
|---|---|---|---|---|
| (Raj et al., 2014) | Role switching technique (between host & client) | Software | Wi-Fi WFD | more energy and processing<br>higher network latency |
| (Liu et al., 2016) | Role switching technique (between host & client) | Software | Wi-Fi WFD | possible communication collision |
| (Bellavista et al., 2010) | Middleware for heterogenous connection | Software | Wi-Fi Bluetooth | limitations of Bluetooth interface compared with Wi-Fi: lower range, higher latency, slower data transfer rate. |
| (Gardner-Stephen et al., 2012) | Heterogeneous connection | Software | Wi-Fi Bluetooth | |
| (Dubois et al., 2015) | Middleware for heterogenous connection | Software | Wi-Fi Bluetooth | |
| (Wang et al., 2016) | Heterogeneous connection | Software | Wi-Fi Bluetooth | |
| (Adeyeye & Gardner-Stephen, 2011) | New hardware prototype – mesh potato+analog telephone adapter | Hardware & Software | Wi-Fi | Additional technical knowledge and set-up cost |
| (Stevens et al., 2014) | New hardware prototype – portable telecommunications tower | Hardware & Software | Wi-Fi | |
| (Junio & Chavez, 2018) | Existing router, solar-powered server + proprietary application | Hardware & Software | Wi-Fi | |

In addition to using mobile phones as the devices to form WCN, some researchers constructed prototypes to serve as gateways for WCN (Adeyeye & Gardner-Stephen, 2011; Stevens et al., 2014). It was reported that the implementation cost of the designed prototypes was less than that of a typical router and can be set up effortlessly in a real environment. However, setting up and operating the devices require some technical knowledge. In comparison, mobile phones are already available almost everywhere in the market and have the potential to provide similar functionality as a WCN gateway, without any additional hardware setup cost or technical knowledge needed. Table 2.4 summarizes the works done by other researchers and the limitations of their works.

### 2.4.3 Relevant Projects in the Market

In terms of related projects in the same research area, several researchers have designed and published innovative technologies. These technologies mainly target the formation of a temporary WCN for communication or data exchange without the need for an infrastructure network. Some of the famous projects include the Serval project, goTenna, Bridgefy, Open Mesh, and Open Garden

The Serval Project was initiated after the Haiti earthquake in 2010, led by the esteemed researcher Dr. Paul Gardner-Stephen. The team has published several inventions, the most popular of which is the ServalChat mobile application for messaging, voice calls, and file sharing using a smartphone. The application operates on a mesh architecture via Bluetooth or infrastructure mesh architecture via Wi-Fi. Additionally, the team has invented a portable router-like device named MeshPotato to form a mesh network. These products have been proven to work effectively, having been successfully deployed in several rural areas around the world (Adeyeye & Gardner-Stephen, 2011; Gardner-Stephen et al., 2012, 2013).

goTenna was founded by Daniela Perlein and Jorge Perdomo in 2012. The team provides a range of personal and mesh networking devices that allow users to communicate without relying on infrastructure-based network facilities such as cell towers, routers, or satellite connections. It operates in standalone mode via low-power, long-range RF technology to create a P2P network. The main products include goTenna MESH, goTenna PRO X, and goKit2 (goTenna, 2023).

goTenna MESH is a portable and lightweight mesh device that pairs with personal smartphones with goTenna's native application installed, enabling connected devices to send text and GPS locations. goTenna PRO X is a tactical-grade VHF/UHF radio device that pairs with smartphones or other devices with goTenna's native application installed. goKit2 is a complete set of mesh devices (PRO X+ tablet) to be used as an off-grid mobile command center during critical and emergency events, such as disasters. PRO X and goKit2 have been used by many licensed professional teams around the world to deliver critical information to the tactical edge or during search and rescue operations. The mesh devices support store-and-forward

mechanisms to relay messages and intelligently find the best path to ensure message delivery (goTenna, 2023).

Bridgefy was founded by Jorge Rios and Drew Harry in 2015. The team started the project with the idea of developing a mobile messaging application that works without the internet. The application then received overwhelming support from users around the world, and the project has been growing exponentially since then. The Bridgefy messaging app allows users to send text messages, images, audio, and location information even when they are not connected to the internet. The offline communication app also supports a delay tolerance mechanism, whereby a message could be delivered to a targeted device located far away from the source device through multi-hop message delivery by the mobile devices over the Bluetooth piconets. On top of that, the team provides the Bridgefy SDK, a development kit for developers to create apps that can work offline using Bluetooth mesh networking technology. The offline messaging app has been proven to work and is used by people around the world in critical situations such as natural disasters[8], civil unrest[9], or war[10].

The Open Mesh project started after the Egyptian government attempted to shut down the Internet in the country in January 2011 (Daniel Lyons, 2011) . The project was founded by Shervin Kordary Pishevar in 2011. The team distributed open-source mesh software through physical storage, such as optical discs, for communication devices, including smartphones, laptops, and devices with AP functionality. The open-source applications were used to establish a mesh network and allowed users to communicate without an Internet connection. However, as of the

[8] Mexico earthquake in 2017, source from: https://bridgefy.me/blog/mexico-city-uses-bridgefy-for-safety-after-earthquake/

[9] Koetsier, J. (2019, September 2). Hong Kong Protestors Using Mesh Messaging App China Can't Block: Usage Up 3685%. Forbes. https://www.forbes.com/sites/johnkoetsier/2019/09/02/hong-kong-protestors-using-mesh-messaging-app-china-cant-block-usage-up-3685/

[10] Koetsier, J. (2022, February 25). Ukrainians Prepping For Internet Loss By Getting Apps For Offline, Private, Mesh Communications. Forbes. https://www.forbes.com/sites/johnkoetsier/2022/02/25/ ukrainians-prepping-for-internet-loss-by-getting-apps- for-offline-private-mesh-communications/

time of this writing, the project appears to no longer exist, and thus not much information about the open-source software could be retrieved.

Open Garden was co-founded by Micha Boneliel in 2011. The project aimed to establish a decentralized and open-source platform that allowed wireless connectivity to users through a P2P network. The platform enabled users to communicate, exchange data, and even share Internet connections with other users who needed them. Open Garden used a combination of technologies, such as mesh networking and software-defined networking (SDN), to create a robust decentralized network (Olson, 2014).

FireChat was a proprietary mobile application developed by Open Garden. It allowed message delivery without using the normal Internet or mobile telecommunication network. As of the time of writing, the project appears to have been discontinued. However, the project had demonstrated its success based on the reported news on the massive number of users' downloads across different countries around the world (Olson, 2014).

A summary of all the projects discussed is demonstrated in **Error! Reference source not found.**. The objectives, features, and status of the projects have been listed. Based on the summary, it is noted that most of the projects began with the aim of developing an offline communication application. All the projects studied have adopted the store-carry-forward paradigm for multi-hop message delivery with the help of the nodes (mobile devices or proprietary devices built) as the gateway. The network formed is known as DTN, as the message delivery speed depends on the availability and density of the nodes available. All the projects reviewed utilized Bluetooth as the transmission medium. However, not all employed the Wi-Fi protocol. Bridgefy only works on Bluetooth technology, and goTenna works on Bluetooth and RF technology. Although most of the projects reviewed support Wi-Fi connectivity, as far as what was studied, only the FireChat project team has reported the implementation of WFD technology in the solution built. Therefore, it is assumed that WFD technology is underutilized, which led to the decision to adopt and focus on WFD technology as the transmission medium in the current work.

Table 2.5 Summary of the projects found related to current work.

| Project | Objectives | Features | Wireless Interface | Status |
|---|---|---|---|---|
| Serval Project | Disaster proof wireless networks rely on the connectivity of mobile devices. | Meshing protocol, Application to make call, messaging, file sharing | Bluetooth, Wi-Fi AP mode | Active https://www.servalproject.org/ |
| goTenna | Forming a robust mesh network for personal or professional use via the proprietary mesh devices | Location tracking, collaborative mapping, and messaging | Bluetooth, RF | https://gotenna.com/ |
| Bridgefy | Development SDK with Bluetooth mesh protocol and messaging app for messaging offline | SDK for meshing with Bluetooth, hope-to-hope data delivery, and an app for messaging | Bluetooth only | https://bridgefy.me/ |
| Open Mesh Project | Open-source tools and a routing protocol for ad-hoc mesh networks. | Meshing protocol API, Application to make call and messaging | Bluetooth, Wi-Fi | discontinued |
| Open Garden | Provide wireless connectivity via a P2P wireless network among mobile devices. | multi-hop wireless connectivity (master-client) | Bluetooth, Wi-Fi AP mode | discontinued |
| FireChat (Open Garden) | Allow message delivery without using the normal Internet or mobile network. Messaging | multi-hop messaging | Bluetooth, WFD | discontinued but apk still available on third-party online source |

## 2.5 LITERATURES FOCUSES ON WFD

Based on the preliminary investigation of the literature in the field, it has been found that WFD technology has the potential to establish a robust WCN. Therefore, a systematic scoping review was conducted to specifically identify relevant research in the field of P2P communication focusing on WFD in the last 10 years. The details about the process of the scoping review are reported in CHAPTER III. The remainder of this chapter will focus on the key literature that was screened and reviewed during the scoping review process.

### 2.5.1 Utilizing the WFD Protocol's Procedures to Form a Solution

Sha et al. (2016) created a mobile application that distributes tasks to stream a video to peers via a WFD network. The software works by assigning peers the task of downloading videos and then combining the downloaded segments to compile the complete video. When the service seeker device broadcasts a video chunk download service request using the service discovery (SD) protocol in WFD, nearby peers reply to the request and carry out the download task. Finally, the peers transfer the downloaded chunks to the device that requested the service, which acts as the GO based on the WFD protocol.

Sha et al. (2016) evaluated the mobile application created by modelling various scenarios to measure the parameters that guarantee smooth video playback. The measured parameters include the connection and disconnection period (in seconds), the buffered content's playback time (in seconds), the number of video chunks (n), the time for playing video (in seconds), and the extra buffered video content (in kB).

Likewise, Hu & Cao (2017) introduced a framework named QATO to offload network tasks to peers with better service quality. The network information of devices in proximity was exchanged by leveraging the SD protocol of WFD, and the data was embedded in the device's SD service instance. The peers with the SD protocol enabled searched and listened for the service instance and received the embedded data. The

peers then compared the local and neighbouring network information (proxy lists generated from SD) and made an offload decision, either to self-execute or offload network tasks. The uplink and downlink throughput of different carriers and the power consumption when using carrier connections were measured.

The results of the work proved that traffic offloading with WFD can save more energy and cause less delay compared to sending data via cellular interfaces. It was reported that Android introduced DNS-based SD since Android 4.1 (Jelly Bean), and the SD protocol took an average of 2 seconds to discover a peer. These procedures were chosen because the main aim of the works was to distribute tasks to nearby devices in the group, thus not requiring the multi-hop mechanism.

Links et al. (2019) and El Alami et al. (2017) have created a data exchange framework for a mobile device capable of transmitting data to an internet server even when there is no internet connection. The transmission without an internet connection was achieved through the service discovery (SD) request and response protocol of WFD. Before the SD protocol is activated, the HTTP request data is embedded in the device's SD service instance. Peers with the SD protocol enabled will search and listen for the service instance and receive the embedded HTTP request data. If the peer device has an internet connection, it will create an HTTP post request and deliver the HTTP request data to the target server. If the peer device does not have an internet connection, it will wait for n seconds to check again if an internet connection is available. If the internet connection is still not available, the peer will broadcast the HTTP data to the next peer. The same procedures are repeated until the HTTP data can be delivered to the destination server. The proposed protocol has been tested in a real-world environment with three mobile devices and also with the WiDiSi (Baresi et al., 2016) simulation tool. Some of the parameters measured include the time for service discovery (seconds), waiting time to re-broadcast data (seconds), propagation time (seconds), transmission time (seconds), the success rate, and the overhead of data delivery.

In another similar work, A A Shahin & Younis (2015) proposed a protocol based on the store-carry-forward algorithm to disseminate alert data in a critical

situation. When there is an emergency, the mobile device will activate the protocol and generate an alert message. The protocol will then embed the alert data (e.g., GPS location, event) in a newly generated SD service instance and disseminate the instance via the SD protocol. Initially, the time-to-live (TTL) of the service instance will be set to an integer value. The TTL value of every instance will decrease at a constant rate every few seconds. When the value reaches 0, the service instance will be removed from the library. When the nearby device receives the SD service instance from other devices, the embedded alert data will be extracted and stored in the local library. The stored alert data will be forwarded to other nearby devices, and the process will repeat until all nearby devices receive a copy of the alert data.

Other than utilizing the SD protocol, some researchers proposed altering the source code of the SD protocol to embed additional data into the standard protocol's frames. Pan & Wang (2019) proposed a data exchange scheme by modifying the source code of the device discovery procedure of the WFD protocol. The device name field of the probe request and response packets has been modified to carry the message to be transmitted between devices. A simulator developed with Python was used to test the proposed scheme. The result shows that the latency of message delivery has been reduced compared with the implementation of the original WFD protocol for data exchange.

In another similar working concept, Almeida et al. (2020) compared the performance of WFD in the real vehicular environment with the simulation model in the OMNet++ simulator. The parameters measured include communication range, packet delivery rate, and packet inter-reception time. The experiment results showed that the packet delivery rate starts to drop at 100 meters in the VANET setting and at 120 meters in the simulator. Beginning at an average distance of 150 meters, the delivery rate continues to drop. The authors also reported that long connection establishment times limit the QoS (quality of service) in data transmission, especially in VANET and in an environment with obstacles. Hence, the authors proposed the "beacon stuffing" method of embedding additional data into the device name fields (32 bytes) in the beacon frame to carry additional information. The study emphasized the difficulty of using WFD in mobility scenarios due to the lengthy connection

establishment time. Therefore, a schema with a faster connection time is needed to ensure the nodes can communicate in a short period of time.

The proposed techniques have been proven to improve the performance of P2P data exchange. Moreover, the idea of utilizing or modifying the standard protocol of WFD technology as the transport method for data exchange is innovative. This introduces a novelty in the research field by providing additional options for transferring data over the Wi-Fi protocol. However, implementing the proposed scheme would require root access or recompilation of the Android system source code, making it impractical in real-world situations. Furthermore, none of the reviewed proposed models could guarantee the delivery of source data to the destination. In other words, the proposed protocol only supports unidirectional data propagation.

Based on this key literature, a potential research gap has been identified in the area, which is to improve the proposed model to support bidirectional data exchange procedures to form ad-hoc communication networks. Additionally, the proposed model should maintain the originality of the WFD protocol and not require rooting or recompiling the system source code.

### 2.5.2 Applying the Concept of Multi-layer Operations and Abstraction

Li et al. (2020) developed an intergroup and intragroup communication model by utilizing WFD at the application layer. The proposed model adopted the role-switching mechanism, which is similar to findings reported in other literature (Liu et al., 2016; Raj et al., 2014). The authors proposed the handover procedure for GO and Group Member (GM) roles in a WFD group, using a fuzzy-logic-based normalized quantitative decision algorithm. The performance parameters measured include throughput (Mbps), goodput (Mbps), packet loss rate (%), and handover duration (seconds).

The test results showed that the model could achieve a throughput of 31.7 Mbps for intragroup communication and a goodput of 4.76 Mbps for intergroup communication. The high throughput of intragroup communication was due to data

exchange happening within the WFD group (intragroup) in close proximity, functioning similarly to normal infrastructure Wi-Fi transmission. The lower transmission rate (goodput) occurred during data exchange between WFD groups (intergroup). In intergroup communication, the handover model introduced required the processes of disconnection, discovery, and formation, taking an average of 7.8 seconds.

Apart from the results, the literature revealed some important key points closely related to the current work. First, the operational mechanism of WFD was highlighted, indicating that only the GO can obtain GM's information and capture events regarding GMs joining or leaving the group. This is crucial for designing a mechanism for WFD group management. Second, the authors reported that throughput measurement evaluates the performance of layer 2 and layer 3 protocols, while goodput measurement assesses the performance from layer 4 to layer 7 of networking standard protocols. This clarifies the appropriate measurement parameters for reporting. These key points were considered during the development of the new model and the performance measurement process.

Maia et al. (2014) developed a framework with carry-and-forward routing algorithms for developers. This framework implemented an abstraction technique that combined and abstracted the management of the underlying BT and Wi-Fi operations. The framework was divided into three layers: the lowest connection-aware layer handled the implementation of wireless connections, such as BT, Wi-Fi, and WFD. The connection-aware layer managed operations like device discovery, connection and disconnection, and unicast and broadcast message delivery. The network layer, situated at the second level, was designed to manage message routing using the AODV and flooding algorithms. The application layer, the highest layer, provided interfaces for developers to implement the framework. This approach relieved developers from focusing on connection and routing operations, allowing them to concentrate solely on data or message exchange within their applications.

To provide a proof-of-concept, the framework was implemented in a mobile application built on the Android platform. Seven Android-based smartphones were

utilized to test the performance. The average Round Trip Time (RTT) for message delivery between two hops using the AODV algorithm was recorded at 250 ms. Message loss percentages were tested using the flooding algorithm: approximately 0% for 3 hops, 5% for 4 hops, and 10% for 5 hops. The increase in message loss percentage with the number of hops was attributed to instability issues in the BT API of the Android platform. However, no data was available for the RTT of message delivery with the flooding algorithm implementation, and the data loss ratio of the framework using AODV algorithms was not recorded. This absence of data makes it difficult to fully evaluate the feasibility of the framework. Despite these limitations, the work demonstrated the proof-of-concept for the abstraction approach and contributed valuable information to the DTN research domain.

In another similar work, Khawaja et al. (2020) introduced the ad-hoc collaboration space framework, which offers an API abstraction layer atop the WFD protocol. This API allows developers to integrate a data communication model into their mobile applications. The framework was tested with a drawing application as a collaborative tool, synchronizing drawing sessions between smartphones. Parameters measured included SD and group formation times, state update and synchronization times, and memory overhead generated. Multiple smartphones were employed for experimental testing, forming various test case sets. Promising synchronization durations were demonstrated for operations performed after a WFD group was formed. Reported synchronization times were generally under 100 ms for different device and operation combinations, attributed to devices operating in Wi-Fi mode with speeds of 1~2400 Mbps (as illustrated in Table 2.2), based on available Wi-Fi versions.

Notably, the synchronization model of the framework is confined to intragroup connections and does not support data transmission in multi-group, multi-hop configurations. However, the experimental testing methodology showcased systematic testing in diverse settings. This systematic approach is crucial to assess the framework's adaptability in different conditions. Hence, this well-structured experimental testing methodology was considered during the evaluation and testing phase of this research.

Furthermore, the abstraction technique employed in the framework, as well as the development model applied by researchers in the literature (Khawaja et al., 2020; Li et al., 2020; Maia et al., 2014) has also been adopted in this research. This is because the abstraction technique has been proven to increase reusability, simplifies implementation, and reduces the complexity of the lower layer classes.

### 2.5.3 Applying the Concept of Middleware or Mobile Agent

Arnaboldi et al. (2017) developed middleware to compute the best Wireless Fidelity (Wi-Fi) Direct (WFD) group configuration for enhancing multi-hop communication data dissemination performance. The exchange of group configuration data (SSID, battery status, and computational resources) between nearby devices is facilitated using the SD protocol. Based on the exchanged information, each device computes and identifies a potential node between groups to act as the gateway for data transmission. Subsequently, a new connection is established with the selected node through the WFD protocol, enabling multi-hop data dissemination across groups.

For evaluation, the baseline WFD protocol was used as the benchmark, and the ONE simulator was employed (Keränen et al., 2009). Three test cases were formulated, representing low, medium, and high mobile node mobility scenarios, with varying numbers of nodes and movement speeds. The evaluation metrics included the frequency of message exchanges between nodes, the percentage of successful message dissemination, and the rate of battery consumption by the nodes.

The results demonstrated that the middleware significantly improved network connectivity and message dissemination in low and medium mobility scenarios, but its performance was comparable to the baseline protocol in high mobility scenarios. However, it's worth noting that the evaluation was conducted solely in a simulated environment, lacking experiments on physical devices, which raises questions about the middleware's real-world performance guarantee. Keränen et al., (2009) have incorporated a middleware concept in their model, which shares similarities with the agent-based mobile application proposed in this research, designed to possess data dissemination capabilities.

### 2.5.4 Focusing on GO Selection

Meng-Shiuan Pan & Lin (2018) proposed a model for reducing GO load in WFD intragroup communication. The model worked by dividing the main WFD group into several small groups and then using the store-carry-forward protocol to disseminate data in multi-hop mode. The authors proposed using the SD protocol in each peer to broadcast data for group formation and repair procedures. Once the peers received the necessary data, the roles of the GO and client devices would be switched to relay the data across the formed small groups. For evaluation, the measured parameters included network formation time, data dissemination time, network repair time, and SD discovery time. The model was tested in an event-based Wi-Fi P2P simulator in various test cases.

During the simulation, it was found that the average network formation time increased as the number of devices increased. However, when the maximum number of devices in a group was set to a higher value (e.g., greater than 5), the average network formation time was successfully reduced. The ring-based data dissemination scheme outperformed the tree-based data dissemination scheme in terms of network formation time, data dissemination time, and instant repair time. Additionally, the authors conducted an SD discovery time experiment. In this experiment, more than 8 devices were set up as service provider devices, providing 1 to 4 different kinds of services. Then, a device was designated as the service discovery device, triggering the SD process. The SD process was executed 20 times, and the SD times were recorded. It was found that SD times were not consistent, with the longest duration being 15.2 seconds and an average of 4 seconds.

Based on the results, the authors reported that the proposed schemes can effectively balance the GO energy consumption and reduce data dissemination time. However, the energy consumption of the peer devices logically increased due to the role-switching mechanisms imposed on the peers. Furthermore, the proposed model's evaluation was based on simulation, with no physical device implementation. The SD discovery time experimental approach is useful for evaluating SD discovery performance and has thus been adopted in this research (Pan & Lin, 2018).

Ahmed A Shahin & Younis (2015) introduced a protocol for multi-group communication. In the introduced protocol, the WFD's SD (Service Discovery) protocol was used to exchange rank values calculated based on battery information. Devices with higher ranks became GOs, and the GOs appointed proxy members to act as intermediaries for forwarding messages between groups.

The authors reported that a device is only allowed to connect to one WFD group at a time. Moreover, the IP addresses assigned to a WFD group fall within the 192.168.49.x/24 range, with the GO's IP address fixed at 192.168.49.1. Consequently, communication between WFD groups is not possible by default. To overcome this limitation, the authors modified the platform's source code to allow devices to create different subnets.

A simple chat application was installed on three devices to test the protocol. However, the experiment conducted was merely a proof of concept involving only five devices, and no performance evaluation was considered in the reported work. Furthermore, modifying the source code is not practical in the real world, as rooting the device is required to achieve such modifications (Shahin & Younis, 2015).

Yasser et al. (2021) introduced a model named ERRM to choose the GO and backup device based on software service and parameters such as battery power, RSSI, and SNR of the devices in a WFD group. The backup and GO devices were selected at the beginning stage when the WFD group was formed. The selection was based on the intent value calculated by each device. Zhang et al. (2014) also introduced the intent value calculation, focusing their work on GO selection during the WFD group formation phase. The intent value is calculated based on parameters like RSSI, SNR, and the remaining battery of the device. This value is then exchanged by injecting it into the probe request frame of the peer discovery phase.

When a GO leaves, the backup device will quickly form a new WFD group to replace the GO. The ERRM model was tested in three experimental settings with four mobile devices. For each experiment setting, the same procedures were executed three times. The procedures began with the formation of the WFD group, and then the GO

was removed from the group to force the execution of the ERRM model. The formation and reformation times were recorded for comparison. Although it was reported that the group reformation time with ERRM was reduced by 80%, the result might differ. This is because the GO and the backup device were chosen at the beginning of group formation, and the mobility of the mobile devices and the resources of the selected backup device might have changed over time. Additionally, there might be another device with a higher intent value that joins the group later. Therefore, the state is dynamic, and the backup device to replace the GO might not be suitable when the replacement happens.

Despite the limitations of the proposed model, this well-written literature discusses the limitations of WFD and the GO selection procedures. Part of the experiment and evaluation methodology has been adopted for this work.

### 2.5.5 Implementing the Heterogenous or Hybrid Model

Pyattaev et al. (2016), Zuo et al. (2015), and Djajadi & Putra (2014) have employed the hybrid concept (internet and local network) in the model developed to ensure data delivery to its destination. Zuo et al. (2015) built an interactive chat system that allowed communication between mobile devices. When the peers are in proximity, the application will activate the WFD group-forming functionality and then allow communication between the devices that joined the WFD group. When devices are out of range, the interaction session will be switched to online mode (over the internet) for continuous communication.

Djajadi & Putra (2014) developed an alert system that allowed alert messages to be sent to nearby devices in case of an emergency. When the peers are in proximity, the WFD protocol will be used as the transportation for alert message broadcasting among peers. When the devices are far from each other, the alert message will be delivered to the online server using the phone SMS service. The authors have demonstrated the workings of the mobile application by showing multiple screenshots of the interface as a proof of concept.

Pyattaev et al. (2016) combined cellular technology with the WFD technology to enhance the overall performance and speed of data exchange between mobile devices. The cellular network was used to deliver all the peers' information to an online server. The server utilized this information to appoint the best GO and to assist in the P2P device discovery and connection establishment stages. When data exchange between devices was required, the server identified the devices in close proximity and sent the command to form a P2P connection. This way, the device discovery and GO negotiation were skipped, improving the P2P connection formation time. The data exchange load was offloaded to the peers in proximity, and the data exchange happened over the Wi-Fi protocol. In this circumstance, the results showed that the introduced model, which utilized the WFD protocol as the data exchange channel, outperformed the original cell uplink. However, the model required root access to the mobile device to modify the WFD operations, which might not be practical in the real world.

Holzer et al. (2016) proposed a heterogeneous network for communication. The authors suggested combining BT and WFD technology to form a WCN. The proposal was realized through the development of middleware. In the model, the WFD protocol was used to create the intragroup communication network, while the BT standard connection protocol was utilized for intergroup communication.

The proposed communication network was used to compare the performance of three routing algorithms: gradient-routing (GR), flooding, and counter-based scheme (CBS). The evaluation recorded the delivery rate (%) and message load generated during communication. To test the model, the middleware was installed on 10 smartphones, and the experiment was conducted in a room. Two sets of experiments were carried out.

In the first setting, the CBS and flooding algorithms in the broadcasting scenario were compared. In this scenario, a device broadcasted a message to all peers in the network (intergroup and intragroup) when the heterogeneous network was formed. The delivery rate for both flooding and the CBS algorithm was approximately 90%. However, in terms of message load, the CBS routing algorithm generated only

40% of the message loads in the network, compared to 100% generated by the flooding algorithm. This discrepancy occurred because the CBS algorithm provided a controlled scheme to broadcast messages with conditions, while the flooding algorithm did not provide any control schemes and always broadcasted messages received.

In the second setting, the GR and CBS schemes were compared in a unicast scenario. In this scenario, upon formation of the heterogeneous network, all devices in the network sent a message to a targeted device. The delivery rate for both the GR and CBS schemes was estimated to be around 97%. However, the GR scheme added 40–50% more message loads to the network compared to the CBS scheme. This difference was because in the GR algorithm, every device forms a routing table, and the routing table is referred to as a direction to transmit the message to its targeted destination.

Based on the review, it was noted that the BT protocol limited the distance in intergroup communication. The authors did not explain the mechanism to maintain interconnectivity between peers. Furthermore, disconnections are not detected or repaired automatically in the BT protocol by default. Therefore, the proposed model may not work in situations where the devices are not static and occasionally move. Although the GR scheme ensured a better data delivery rate, the table maintenance cost grows exponentially as the number of peers grows. Hence, the scheme might not be suitable in a WCN with a large group of mobile devices. Nevertheless, the systematic experimental method implemented and parameters for evaluation measured in the literature can be adapted for current work.

## 2.6 SUMMARY

In this chapter, the details about the literature and technical documents was reviewed and reported. The first section describes the common wireless technology used in mobile devices. The second part presents details about the WFD protocol (Wi-Fi Alliance, 2016). The third part reviews relevant literature and projects related to our

research. Finally, the literature focused on the WFD protocol was screened and reviewed based on the scoping review technique.

The findings revealed that researchers have proposed numerous innovations and contributions in the OppNet research domain. Some focused on developing software to form the WCN, while others introduced additional hardware for a robust WCN. In the scoping review of the WFD protocol, researchers exploited it to introduce new ideas to the research domain. Some modified and recompiled the protocol's source code or altered the frame structure used by the protocol. Others incorporated a multi-layer structure in their frameworks or developed middleware to enable multi-hop data delivery. Additionally, researchers are working on GO selection to reduce group formation time. Some suggest using multiple wireless interfaces on devices to create a heterogeneous network and improve P2P connections, reducing data dissemination time in multi-hop or store-carry-forward schemes.

Based on the literature reviewed, a research gap was identified, which is to introduce a communication model that utilizes the WFD to support bidirectional data exchange procedures for forming ad-hoc communication networks. The proposed model should maintain the originality of the WFD protocol without requiring rooting or recompiling the system source code, thus avoiding changes to low-level source code. The middleware and abstraction techniques were adapted based on the reviewed from other research works.

The literature review also contributed to identifying evaluation parameters, measurement criteria, and systematic experimental testing methodologies for this work. The next chapter explains the methodology constructed to achieve the three objectives listed in section 1.4.

# CHAPTER III

# METHODOLOGY

## 3.1 INTRODUCTION

This chapter discusses the detailed procedures and research design for the overall research methodology. The chapter is divided into four main parts. The first part provides an overview of the research methodology. The second part explains the methods and procedures used for the preliminary analysis. The third part describes the methods and procedures used for model development. Finally, the last part explains the deployment, measurement, and evaluation conducted on the developed prototype.

## 3.2 RESEARCH METHODOLOGY

This section outlines the overall methodology for the research. Figure 3.1 illustrates the correspondence between the objectives (as listed in section 1.4) and the planned methods. The execution of individual methods ensures the attainment of all three objectives, ultimately accomplishing the main goal of this study. To begin, a preliminary analysis, scoping review, and comparative study were conducted to fulfil the first objective. Subsequently, the prototyping development methodology was adopted to create the model for the second objective. Finally, manual installation and software testing methods were employed to realize the third objective.

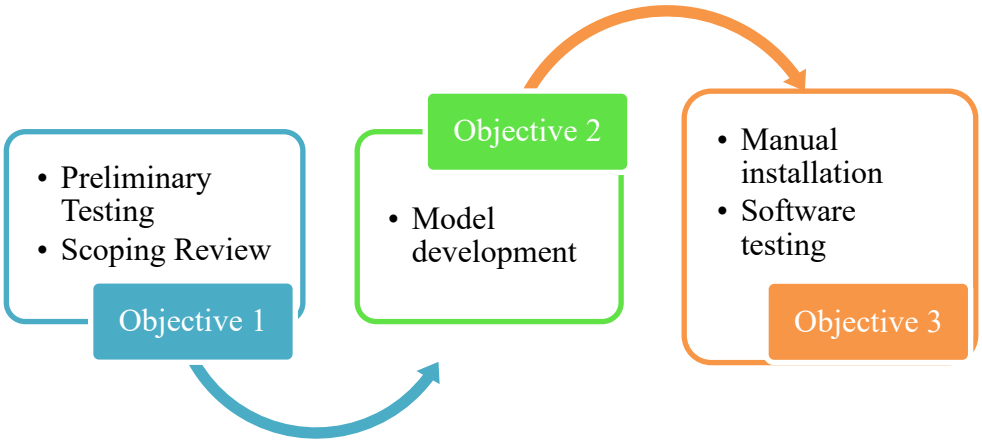



Figure 3.1 Mapping between objectives and methodologies.

## 3.3 PRELIMINARY ANALYSIS

The first objective of this research is to identify constraints, limitations, and potential solutions for WCN formation by end-user devices. Constraints and challenges were identified based on two main factors: existing technology and previous literature. To assess the current state of the art, two straightforward test cases were designed using Android-based smartphones. Furthermore, a scoping review was undertaken to identify and review pertinent literature. The activities involved during the scoping review are explained in section 3.3.2. The results and findings of the scoping review are available in section 4.2, CHAPTER IV.

### 3.3.1 Preliminary Testing

Preliminary testing marked the initial stage in which potential or existing solutions for forming a WCN were identified. This stage is crucial, as it involved studying solutions proposed by other researchers and subsequently evaluating the pros and cons of these solutions. Alongside the solutions proposed by other researchers, the preliminary testing stage also provided a technical overview of wireless technology in mobile devices.

For the preliminary testing, two test cases were developed. The first test case aimed to assess the feasibility of utilizing BT and Wi-Fi interfaces on smartphones to create a WCN. The second test case focused on comparing applications available in the marketplace. Table 3.1 presents the specifications of the smartphones utilized in

the preliminary testing. These smartphones were all based on the Android operating system (albeit different versions) and were equipped with Wi-Fi and BT interfaces. The NFC protocol was excluded from consideration due to its suitability solely for data exchange between two devices, rather than for establishing a mesh network, which is required for a WCN.

Table 3.1 The specification of the smartphones for the experiment

| Brand and model | Samsung Galaxy J1 mini prime/SM-J106B | Asus Zenfone GO/X013D | Lenovo A5000 | Samsung Galaxy Mini / GT5570 |
|---|---|---|---|---|
| Android version | 6.0.1 | 5.1.1 | 4.4.2 | 2.3.6 |
| Year manufactured | 2017 | 2016 | 2015 | 2011 |
| RAM | 1GB | 2GB | 1GB | 280Mb |
| Wi-Fi | 802.11 b/g/n | 802.11b/g/n | 802.11 b/g/n | 802.11 b/g/n |
| Wifi Direct | Yes | Yes | Yes | No |
| BT & Profile | Version 4.0 A2DP, AVRCP, DI, HFP, HID, HOGP, HSP, MAP, OPP, PAN, PBAP | Version 4.0 A2DP, EDR | Version 4.0 A2DP | Version2.1 A2DP |
| NFC | No | No | No | No |

**a. Test Case 1 – Experiment on Existing Wireless Interfaces**

The first test case was designed to assess the feasibility of establishing a Wireless WCN using BT and Wi-Fi interfaces on a smartphone. A sample file transfer process was executed to simulate a basic operation within the WCN. The progression of the file transfer was observed and presented in Table 3.2. The aim was to verify whether the file could be transmitted from a source device to all other devices through an ad-hoc network formed using the existing available wireless interfaces. All mobile devices are equipped with at least two or three types of radios (e.g., multi-band cellular, Bluetooth, and Wi-Fi) that can be utilized for communication with other devices.

In the preliminary testing phase, the higher-level communication platform of the operating system was explored to determine the potential for integrating the

devices into the WCN with minimal coding or modification to the existing device system. Given that the multi-band cellular interface has been hardcoded by the manufacturer, making modifications to the cellular interface was ruled out due to the complexity of accessing and altering the kernel codes. As a result, the available options were limited to the Bluetooth and Wi-Fi interfaces.

**i. Test Case Setting and Operations**

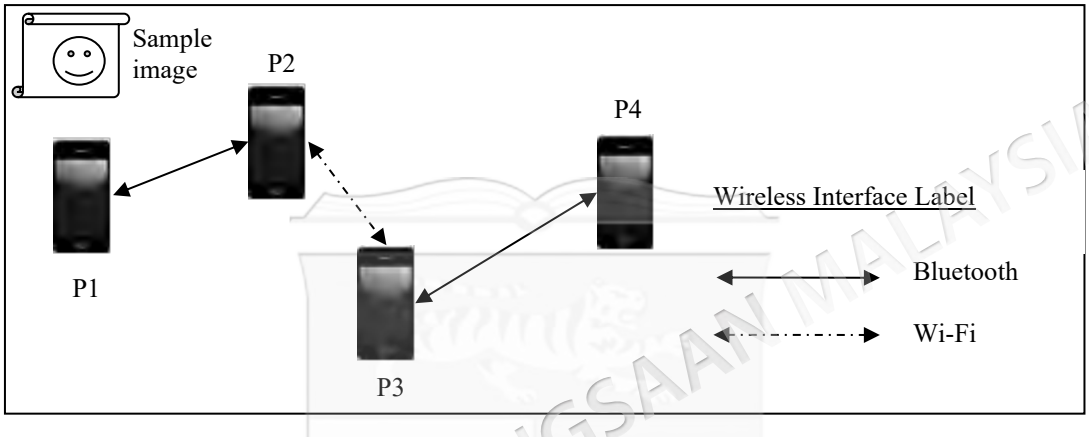



Figure 3.2 The setting for the preliminary testing

A random image size of 2 MB was used as a resource shared by all devices. Before initiating transmission, the wireless interfaces of all devices were enabled. The devices were arranged in a fixed configuration as depicted in Figure 3.2. P1 and P2; P3 and P4 were interconnected via the Bluetooth interface, and P2 and P3 were interconnected via a Wi-Fi interface. For Bluetooth connections, the file-sharing operation was performed manually using the default Bluetooth file-sharing feature in the Android system. For Wi-Fi connections, the application SHAREit from Google Play was installed on every device for file-sharing procedures.

**ii. Observation and Finding**

The findings and observations were tabulated in Table 3.2 for better visualization and interpretation. Based on the conducted tests, it was confirmed that forming a partial WCN with a resource-sharing facility is possible by utilizing two distinct wireless interfaces on a mobile device. This is feasible because a device can simultaneously connect to two other devices using the two different wireless interfaces available by

default on the phone. However, achieving a complete WCN topology in the form of a mesh network is not possible with the default wireless interface of mobile devices. In terms of performance, the inconsistency of data transfer speed and connectivity range poses a major limitation. The Wi-Fi interface supports a longer-range connection and a higher data transfer rate compared to Bluetooth. Consequently, files can be sent farther and faster in the WCN via a Wi-Fi interface. Therefore, the Wi-Fi interface is the superior choice and has been selected as the focal point of the wireless interface in this work.

Table 3.2 The findings of the test case 1 experiment

| Observation | Premise | Factors |
|---|---|---|
| Device P3 is situated near Device P1, but due to the limited connection interface of P3, which is connected to both P2 and P4, the file must pass through P2. | This highlights the necessity for a technique that leverages the wireless interface to facilitate multiple connections. | Interface limitation |
| When one of the paired devices moved beyond the range of the connection, the file-sending session ended, and file-sending failed. | This shows that a technique is needed for the device to stay connected to the neighbouring node whenever possible. However, maintaining connectivity between devices is challenging as the devices' movement or direction is uncontrollable. | Network connectivity preservation |
| The connection time with BT was shorter and more consistent. A Wi-Fi connection took longer and showed inconsistent timing (with up to a 5-second variance) when connecting to a neighbouring device. | This indicates that the device needs a rapid technique to execute the connection operation as soon as it is required. | Device's interface response time<br>- How quickly the Wi-Fi interface could respond. For instance, scanning and connecting to other devices, so that the user experience will not be affected by the latency. |

The goal of this research is to introduce a model for forming a WCN within a MANET. As a result, the model should support an infrastructure-less or ad-hoc communication structure. As elaborated in section 2.2.4 the Wi-Fi protocol has been selected as the central wireless technology due to its remarkable capabilities in comparison to other options. It has been documented that the Wi-Fi protocol supports two types of ad-hoc networking operations (IEEE Computer Society, 2016). One is the native ad-hoc mode, and the other is the WFD protocol, which inherits the

functionalities of the conventional Wi-Fi protocol. Alami et al. (2017) & Liu et al. (2016) indicated that implementing Wi-Fi Ad-hoc mode in Android devices requires access to the kernel codes, making it impossible to apply the mode at the application layer with the Android SDK. Consequently, developing a mobile application that utilizes the Wi-Fi Ad-hoc mode demands root access and recompilation of the entire system for each target device. This approach is impractical when aiming to introduce a framework that utilizes the ad-hoc mode and establishes it as a standard for all mobile devices.

Furthermore, the Wi-Fi ad-hoc mode does not support long-range communication, thereby limiting the scalability of the formed network (Wi-Fi Alliance, 2016). Additionally, Rodrigues et al. (2016) reported that device-to-device communication using P2P protocols such as WFD can reduce traffic by 65% compared to conventional communications over the Internet or access points. Based on all of these observations, WFD has been chosen as the focal protocol in this work.

**b. Test Case 2 - Experiment on Existing Applications**

As part of the preliminary testing, an experiment on relevant applications in the mobile market was conducted to understand the existing inventions or innovations. The main purpose of this experiment was to test P2P-based communication applications in the mobile market. The components observed by each application include the wireless interfaces and protocols supported by the application, the data exchange or communication operations provided, and the additional functions of the application. Whenever possible, the strengths and limitations of each application were identified. The test was conducted by adopting the black-box testing approach. Black-box testing is a technique that focuses on the behaviour of the software being tested. By assuming the software is a black box, the tester does not need to know the details of the implementation or the internal structure of the software. The inputs are provided to the software, and the outputs or behaviours generated are observed and recorded (Lonetti & Marchetti, 2018). Based on the initial selection criteria, six applications were chosen and then tested. The detailed test case settings and operations are

presented in the next section, and the results of the testing have been summarized in a table.

**i. Test Case Settings and Operations**

To select the application for evaluation, the selection criteria include:

1) The application supports offline ad-hoc communication or data exchange.

2) The application has been downloaded at least 500 times from the marketplace.

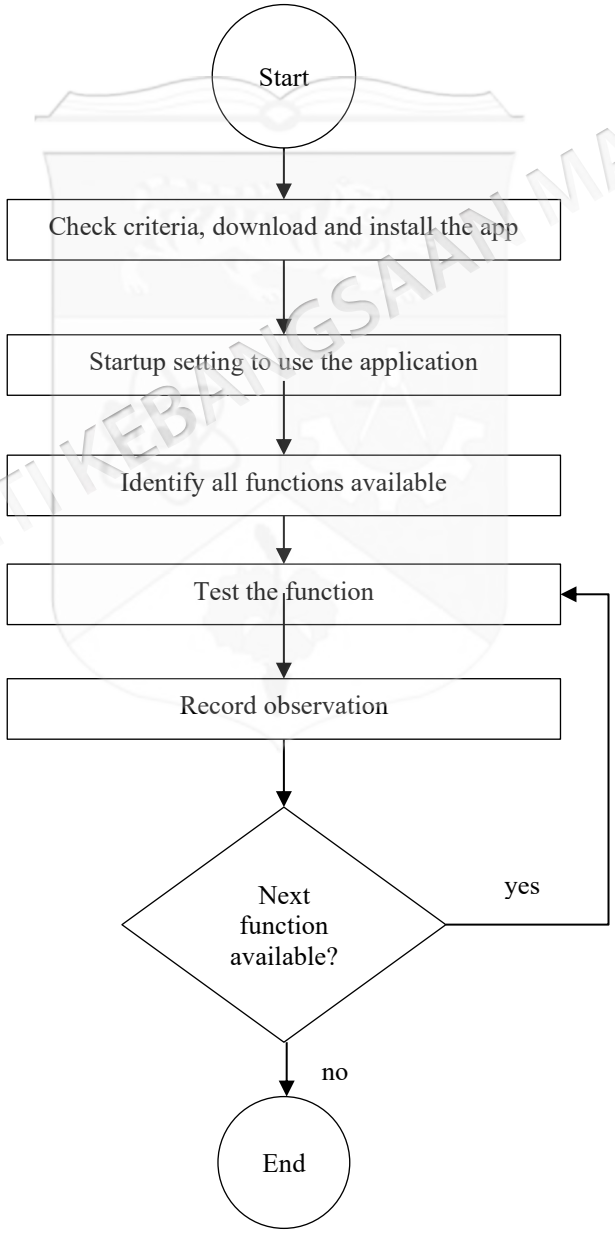



Figure 3.3 Testing procedures on targeted applications.

A standard procedure was established to test the functionality of each application. The procedures are depicted in Figure 3.3. An application was installed on all devices after selection according to the criteria. The introduction page of the application was explored to comprehend its functionality and operations. Post-installation, start-up procedures, including account creation, basic settings, and application configuration, were performed as per the application's requirements. Subsequently, all offline communication functions of each application were tested, and the results were documented.

**ii. Observation and Finding**

The findings and observations are shown in Table 3.3. Based on the observations, it was noted that most of the tested applications utilized the BT and WFD protocols for offline communication. It was noticed that, except for FireChat, all applications support only the WLAN or WFD protocol. Applications working on the WLAN protocol require a router as a centralized device to manage the infrastructure network. The WFD mode did not work when the device was in Wi-Fi mode. This confirms the operational model of the Wi-Fi interface as reported by Wang et al. (2016), since the WFD protocol is a subset of the Wi-Fi 802.11 protocol. Unless a device has two physical Wi-Fi ports, it can only operate in Wi-Fi or WFD mode by default.

Table 3.3 Comparison results recorded from testing and observations

| | BT | Wlan | WiFi Direct | Messaging | Voice Call | File Sharing | Extra feature |
|---|---|---|---|---|---|---|---|
| Serval mesh | ✓ | ✓ | × | ✓ | ✓ | ✓ | Make a call, Maps |
| FireChat | ✓ | ✓ | ✓ | ✓ | × | × | Sending photo |
| WiFON | × | × | ✓ | ✓ | × | ✓ | TicTacToe Game |
| Briar | ✓ | ✓ | × | ✓ | × | × | Private chat group<br>Forum<br>Blog |
| OfflineChat | ✓ | × | ✓ | ✓ | × | × | Using GPS to identify the location on the map |
| Near Peer | × | × | ✓ | ✓ | × | × | send canvas drawing |

The Serval application was tested and worked in BT and WLAN modes. The tested functions include messaging, voice calls, and file sharing. Additionally, the

application provides an application-push-over function, allowing the transfer of the application's setup file from one device to another via BT or WLAN protocol. The FireChat application utilized BT, WLAN, and WFD modes. The application supported a messaging tool with a photo transfer feature. It was observed that the battery levels of the devices dropped at a faster rate compared to other tested applications. WiFON worked only in WFD mode for offline communication. The features supported by this application included messaging and file sharing. Alongside the offline mode, the application also supported online communication over the Internet. It was noticed that when the application was in online mode, P2P operations such as detecting nearby peers were disabled. This indicated that the application only worked in either WLAN mode with an Internet connection or WFD for P2P communication in offline mode. Additionally, it was observed that the P2P communication followed the standard procedures, which include peer scanning and discovery, GO election, and then P2P connection, as reported in the Wi-Fi P2P technical document (Wi-Fi Alliance, 2016). Therefore, the communication procedure with P2P must be set up manually, requiring additional time for the procedures. The Briar application supports messaging, blogging, and forum activities. It operates in BT or WLAN mode, with or without an Internet connection. The group-forming procedure requires peers to use a BT connection and camera to scan the QR code for the first pairing process. The application worked smoothly and remained stable in BT mode; however, when operating in WLAN mode, the peer searching process took a longer time, as did the duration for a message to be delivered over the Internet. OfflineChat provided messaging functionality over a BT or WFD connection. Besides messaging, the application also offered GPS location detection and displayed the device's location on an embedded map within the application. The application only supported one-to-one P2P communication over the BT or WFD protocol. The NeerPeer application only worked in WFD offline communication mode and did not support communication over BT or WLAN. The P2P communication followed the standard group formation procedure reported in the Wi-Fi P2P protocol document. The application supported direct communication between two devices or a group of devices. A device had to be the master for a group, and intragroup communication was not possible in this application. Therefore, the application only supported communication in proximity.

In conclusion, it was noticed that most of the P2P communication that works on the BT protocol is stable and consistent. However, it is undeniable that the constraints of BT (explained in section 2.2.1) have limited the operation and capacity of P2P communication. Although the WFD protocol purportedly provides better capabilities than BT, the findings from the preliminary testing showed that none of the experimental findings reported a stable implementation of the protocol to form a robust ad-hoc mesh communication network.

Furthermore, all the tested applications have adopted the original WFD P2P communication procedures, and the group-forming procedures required the manual intervention of the user to establish the connection for the first time. Moreover, these procedures consumed additional time. Therefore, the decision was made to exploit the protocol to create a model that supports communication in a WCN. The proposed model should be straightforward, minimize the group-forming procedures, and reduce user intervention in the group-forming process.

Based on the detailed investigation of the protocol document, it was found that the WFD protocol offers an optional procedure known as the Network Service Discovery (NSD) protocol. Generally, this protocol provides a framework that allows a host or server device to register services to announce the available communication services. This approach will result in a more effective and target-oriented communication system, enabling client devices' applications to directly identify and communicate with the relevant source providing the required service. The detailed operations of the protocol have been investigated and reported in section 3.3.3di.

To comprehend the extent to which the WFD and SD protocols have been exploited in the formation of WCN, a scoping review was conducted. The following section explains the process by which the scoping review was conducted.

### 3.3.2 Scoping Review

In general, there are several types of literature research methods to discover research related to a chosen topic. Some of the common techniques include traditional literature reviews, scoping reviews, and systematic reviews. The purpose of a scoping

review is to provide a thorough and systematic overview of the existing literature on a particular research topic. The review technique was introduced by Arksey and O'Malley in 2005 through an article published (Arksey & O'Malley, 2005). In the article, Arksey & O'Malley (2005) proposed the first scoping review framework, comprising six procedures. The framework was then enhanced by Levac and colleagues in 2010 (Levac et al., 2010). The enhanced version of the scoping review framework includes a more comprehensive process for each procedure. In 2015, the working group of the JBI produced a detailed manual for conducting scoping reviews. The manual has been updated, and the original framework has been revised to encompass nine procedures (Aromataris, 2021). The approach introduced by JBI aligns with the PRISMA-SCR checklist. Therefore, the checklist has been incorporated into this research to serve as a standard reporting guideline for a scoping review. It is important to note that only a portion of the checklist items was introduced, as the checklist covers a wide array of components for a more comprehensive literature review, which may not be necessary for the current stage. The primary goal of the scoping reviews conducted in this research was to study an overview of the literature, focusing on WFD and SD protocols under the MANET domain. The review also aimed to investigate the solutions developed by researchers in the relevant field and the measurements involved in evaluating the performance of these solutions. The details of the research questions are explained in the following section.



**a. Priori protocol of the scoping review**

The PRISMA extension for scoping reviews (PRISMA-ScR) was published in 2018 (Tricco et al., 2018). As explained by Tricco et al. (2018), scoping reviews can be conducted to "examine the extent, range, and nature of the evidence on a topic or question". Figure 3.4 lists the procedure for the conducted scoping reviews, adapted from the JBI manual (Aromataris, 2021) and PRISMA-ScR checklist[11].

11 PRISMA Extension for Scoping Reviews (PRISMA-ScR): Checklist and Explanation | EQUATOR Network. (n.d.). https://www.equator-network.org/reporting-guidelines/prisma-scr/

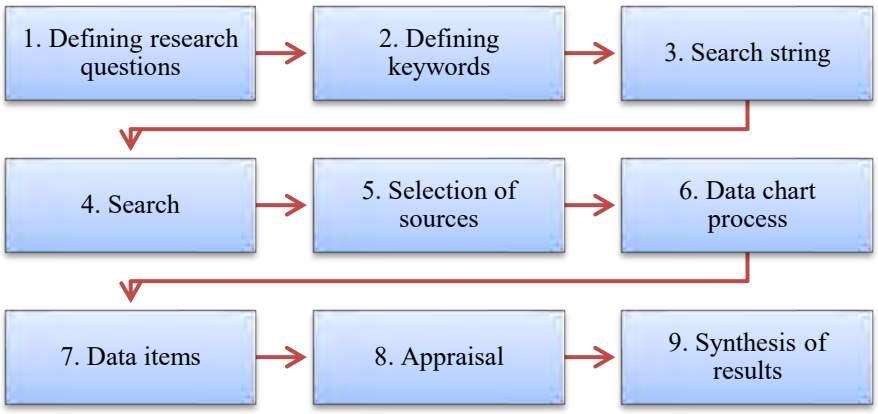



Figure 3.4 Process of conducting scoping reviews.

Source: Aromataris (2021)

1) Defining research questions

RQ1: How are the WFD and SD protocols of Android phones utilized in the research area?

- inclusion criteria, concept: Utilization of WFD protocol

RQ2: What are the technique and parameters to measure the performance of the solutions?

inclusion criteria, concept: Performance measurement techniques

RQ3: What performance metrics are being measured?

inclusion criteria, concept: performance metrics

2) Defining keywords and context.

Keywords: D2D communications; peer-to-peer network; MANET, Wi-Fi Direct; service discovery, performance, measurement, Android

Context: Android platform, Wi-Fi Direct Protocol, from 2009 to 2022 (this is because, according to the WiFi P2P Technical Specification, the first version was released on 2009-12-09, and the scoping review was conducted in the third quarter of 2022)

3) Search string

The relevant keywords are D2D communications; peer-to-peer network; MANET, Wi-Fi Direct; service discovery, performance, measurement, and Android.

Filter the keywords:

The relationship between the keywords listed is illustrated in Figure 3.5. The full term for D2D is “device-to-device”; it is similar to the term P2P. Some researchers use P2P and D2D interchangeably. D2D, or peer-to-peer networking is a sub-domain of MANET. Using the terms D2D or P2P automatically covers the MANET domain. Therefore, MANET can be excluded from the keyword list. The focus of the scoping reviews is to explore relevant research that utilized WFD and SD protocol. The WFD protocol is a type of P2P network in the Android OS. Hence, the keywords “P2P network” and Android can be omitted. For performance measurement, there is no one standard term to be used, and usually, the measurement methods will be reported to evaluate the proposed solution. Therefore, the keywords "performance" and "measurement" were not included in the search string. Lastly, the final keywords related to the aim of the scoping reviews were, "Wi-Fi Direct", and "Service Discovery.".

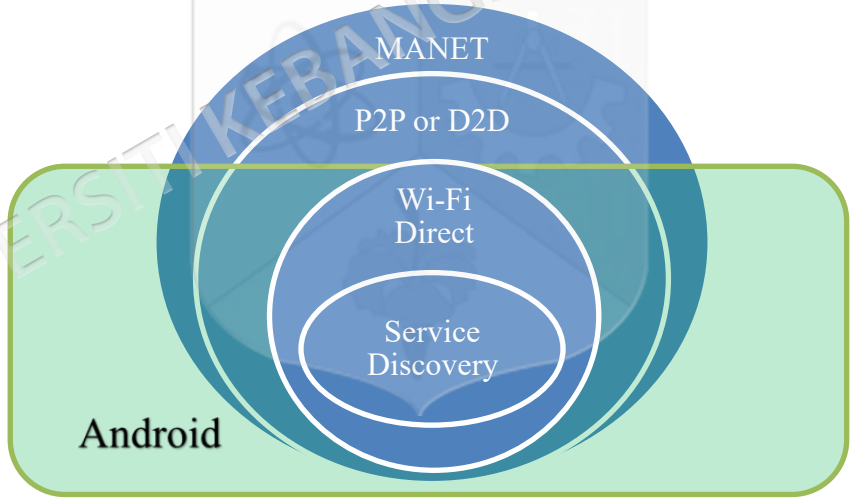



Figure 3.5 The relationship between the keywords identified for scoping reviews.

4) Search

The search was conducted on three well-established information technology databases. IEEE Xplore, ACM digital, and Scopus. The detailed search strings for each database are reported in the next chapter. The search results’ bib text was downloaded and imported into Mendeley for the filtering and screening processes.

5) Selection of sources

a) An article was included if it met the following criteria:

- The article reported on the application of peer-to-peer service discovery.
- The article evaluated the performance of the peer-to-peer discovery or service discovery protocol.
- The article was available in English.

b) An article was excluded if it met the following criteria:

- The article did not implement Wi-Fi P2P
- The article did not focus on the Wi-Fi protocol for P2P network formation (e.g., Bluetooth, NFC, ZigBee)
- The full text of the article was not available (e.g., conference abstracts)

6) Data charting process

The data charting process was conducted independently. The collected literature data was filtered, tabulated, organized, and presented in table form.

7) Data items, appraisal, and synthesis

The data items are presented in Table 4.2 of CHAPTER IV. The critical appraisal and synthesis of relevant sources are presented in section 2.5 of CHAPTER II.

### 3.3.3 Possible solutions

#### a. Socket programming

Socket programming is a programming feature that enables two programs to communicate bidirectionally over a network. The program requires an IP address and port number to establish a socket connection for communication. In socket-based communication, the client device and the server establish a connection by creating and employing sockets, using the configured IP address and port. The client process connects to the server by creating a socket and specifying the server's IP address and port number. The server process awaits incoming connection requests from clients by creating a socket and binding it to a specific port number on the server's machine.

Once the connection is established, both devices can initiate the data exchange process through the socket. The client transmits data to the server by writing it to the socket, and the server receives the data by reading it from the socket. Similarly, the server sends data to the client by writing it to the socket, and the client receives the data by reading it from the socket (*Socket Programming*, n.d.).

Numerous applications in the Android market utilize socket programming, providing client-server communication functionality. Socket programming is employed to facilitate communication between the proxy server and the client. A proxy server acts as a bridge between two devices to facilitate data exchange when direct communication between the devices is not possible. The socket programming paradigm is used to implement the functionality of the proxy server application, making it one of the viable tools for bridging the communication gap. Consequently, it holds potential for implementation as a solution for forming WCN. As part of preliminary testing, a proxy server tool named "Servers Ultimate"[12] was installed and evaluated. The detailed findings were presented in a seminar (Lee et al., 2018). The operational procedures for the host and client devices are outlined in Table 3.4. The left column illustrates the procedures for the host device, while the right column outlines the procedures for the client device. Once the host enables the proxy sharing function, the two devices initiate the request exchange process.

Table 3.4 The operating procedures between the client and a host acting as a proxy server

| Procedures for the host device | Procedures for the client device |
|---|---|
| 1. Connect the proxy device to a network with an Internet connection through infrastructure mode. | 1. Connect to the proxy device through ad-hoc mode. |
| 2. Form a new network and set the SSID and passkey. | 2. Set the proxy address and port |
| 3. Enable or disable a proxy for Internet connection | |

Based on the experiment on the proxy process, it was found that there were some limitations to the application. Firstly, the end user must have some networking

[12] Servers Ultimate - Apps on Google Play. (n.d.). https://play.google.com/store/apps/details?id=com.icecoldapps.serversultimate

background to fully understand how the application works, and this is not always feasible. Secondly, there is a restriction on the mobile device's Wi-Fi interface's nature, which constrains the bridge's functionality. Thirdly, the proxy server only works with the IP protocol in a specific subnet, and the scalability of the network is limited. Therefore, an attempt was made to exploit the proxy server and socket programming concepts. Building on the client-server concept, a server or host is a device that provides the service, and the client device connects to the server for the needed service. The bridge device needs to function as both host and client simultaneously. The concept of bridging is illustrated in Figure 3.6. The middle device running the prototype will connect to an Internet network through infrastructure mode while also connecting to another network through ad-hoc mode. The main operations of the middle device were summarized in a list of operations as follows:

1. Connects to infrastructure mode
2. Connects to ad-hoc mode
3. Parse requests
4. Send requests
5. Forward results

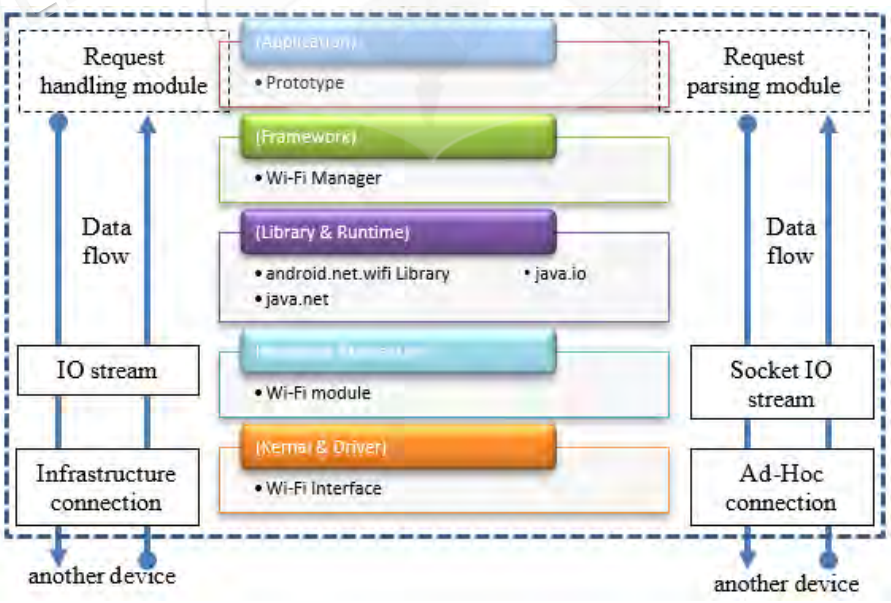



Figure 3.6 The bridging model of the device in the middle

A prototype named "Proxy Net Share (Beta)" was built using the Android Studio IDE. Figure 3.7 represents a screenshot of the prototype, with the function of each button labelled. In the prototype, the infrastructure network and ad-hoc connection configuration procedures were combined, and the proxy setting procedure was done with just one button click. This is achievable, as 192.168.49.1 is the fixed GO IP address. Therefore, it was reasonable to fix the proxy IP, and the user of the client device did not need to perform the complicated proxy setting steps. By default, the client device that wants to connect to a proxy server must go through five configuration steps. Figure 3.8 shows the reduced number of steps from five to three with the implementation of the prototype.

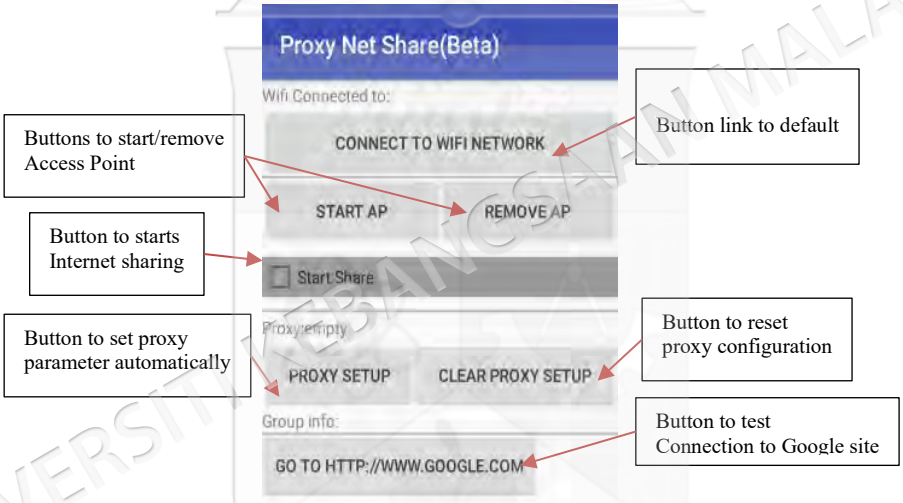



Figure 3.7 Screenshot of the prototype developed for preliminary testing

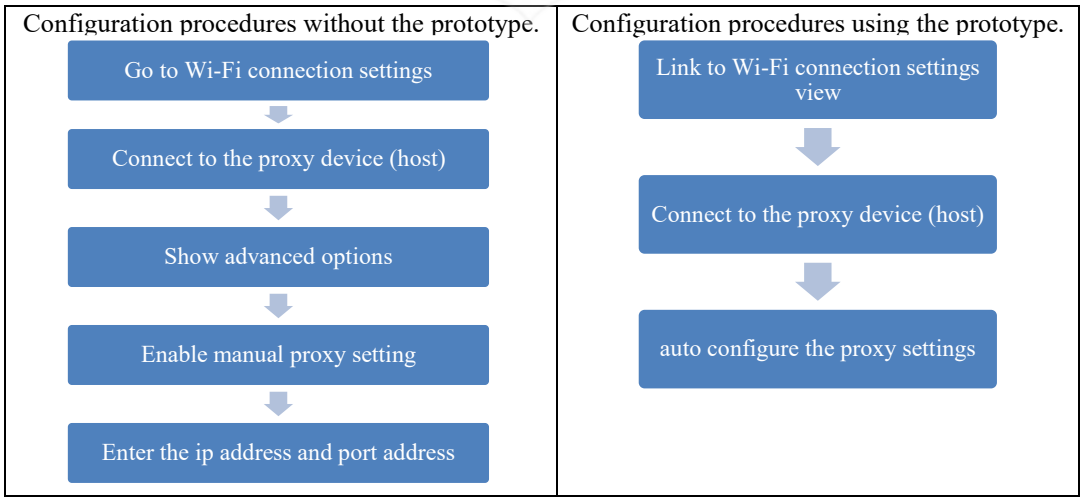
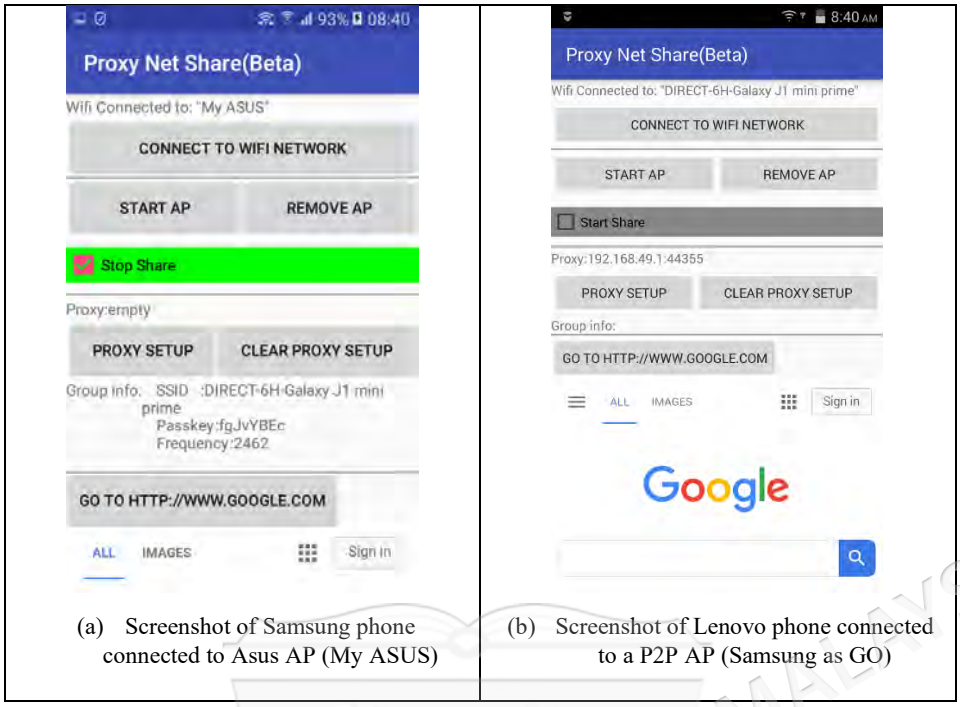



Figure 3.8 Comparison of the client device configuration steps without (left) and with (right) the prototype.

A simple experiment with test cases was conducted to assess the developed prototype. For the experimental setup, three mobile phones with different versions of Android were employed to ensure diversity. The brands and Android versions of the three devices are Samsung (V6.0.1), Asus (V5.1.1), and Lenovo (V4.4.2). The first phone (Lenovo) is assumed to be in a situation where no cellular network connection is available, and it is distant from any open access router with an Internet connection. However, the Lenovo phone is within the Wi-Fi signal range of the second phone (Samsung), which itself is connected to an Internet gateway. This Internet gateway is the third phone (Asus), linked to the cellular network with an Internet connection. The Asus phone shares the Internet connection by employing the tethering feature of the Android system. The primary objective of this test case scenario is to provide an Internet connection to the Lenovo phone. The Samsung phone serves as the bridge, connecting the Lenovo phone to the Internet gateway. Figure 3.9 illustrates the arrangement of the overall test case. Figure 3.10 demonstrates the screenshot depicting the status of the ASUS phone with the hotspot enabled and a client phone (Samsung) connected.


Figure 3.9 The arrangement of the overall test cases scenario


Figure 3.10 Screenshot of the prototype running on Asus phone with hotspot enabled




(a) Screenshot of Samsung phone connected to Asus AP (My ASUS)

(b) Screenshot of Lenovo phone connected to a P2P AP (Samsung as GO)

Figure 3.11 Screenshot of the prototype running on the proxy device (a) and a client device (b) as the result of the test case

Figure 3.11 (a) illustrates the resulting screenshot captured from the Samsung phone, which acts as the intermediary proxy device running the prototype app. Detailed explanations are provided below:

1) The Wi-Fi connection status indicates connection to the "My ASUS" hotspot.

2) The Access Point (AP) function has been initiated with the generated and presented P2P group information.

3) The Internet sharing function has been activated.

4) The Internet connection to the Google website is active.

Figure 3.11 (b) displays the result screenshot captured from the Lenovo phone, functioning as the client device running the prototype app. The detailed explanations are as follows:

1) The Wi-Fi connection status shows connection to the temporary P2P group generated by the Samsung phone.

2) The Wi-Fi proxy is configured to host 192.168.49.1 and port 44355.

3) The Internet connection to the Google website has been successfully established.

The results show that the third phone (Lenovo) successfully connected to the Internet through the second phone (Samsung). The steps to enable the Internet sharing functionality to have been simplified in the prototype. The prototype has successfully diverted Internet traffic from a tethered phone gateway to the destination, which is one hop away. This shows that socket programming could be a possible scheme for data exchange between devices in WCN. However, the data exchange scheme with socket programming requires configurations on the client and server sides, along with proxy settings. The coding implementation of a proxy might be different on different phones, and hence the proposed solution will only work on certain phone models. Furthermore, the IP address of the tethering device is fixed at 192.168.49.0.

Almost all Android-based mobile phones can be used as an access point (AP) with the built-in tethering function if the tethering function has not been disabled by the phone maker. The phone acting as an AP can then be connected by other devices through a standard Wi-Fi connection to share an Internet connection. Theoretically, an Android phone with a tethering feature should be able to perform as a bridge by default. However, it was found that once the tethering function is activated, the Wi-Fi connection to which the phone is connected through Wi-Fi infrastructure mode will be disabled. This limitation has also been reported by (Wang et al., 2016).

Therefore, it was concluded that a phone connected to an infrastructure mode Wi-Fi network is unable to use the tethering feature. Hence, the solution is limited only to intragroup communication. As a result, a robust solution is required to form a WCN that covers as many phones as possible, and it should not be restricted to working only within one network group.

**b. Role switching**

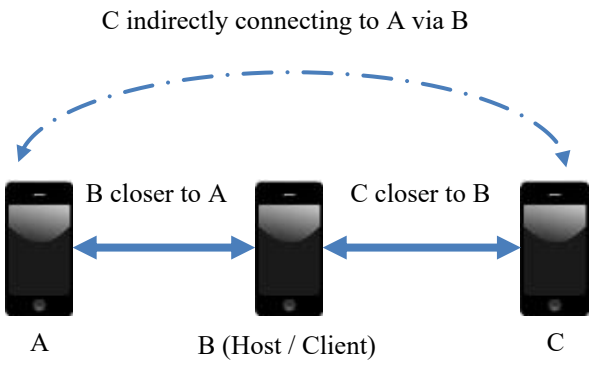



Figure 3.12 Role switching at the middle device

A fully functional WCN should support P2P data exchange in mesh form. The devices in the WCN should be able to communicate with more than one device, either simultaneously or within short intervals of time. Figure 3.12 illustrates an ad-hoc scenario in which three devices are part of the collaborative network. At all times, the devices that have chosen to be part of the collaborative network should be able to communicate with each other. In terms of distance, A is closer to B than C, with C being far away from A but able to connect to B. In this context, B serves as the bridge between A and C. A acts as the host for B, and B acts as the host for C. Due to the constraint that B can function as either a host or a client at a given time, the concept of switching modes can be applied to B. B will alternately function as a client for A and a host for C on a suitable time interval, effectively acting as the bridge between A and C.

By default, the mobile device operates in client mode, seeking out the service provider (server). Android also includes an additional built-in feature allowing the mobile device to become a service provider (WFD GO, tethering) and serve as a host (server) to provide data exchange services to multiple client devices. However, these built-in functions are limited to intragroup data exchange and support only a restricted number of client devices. Given the constraints imposed on mobile devices, some researchers have proposed implementing role-switching mechanisms to overcome these limitations and establish multi-hop multi-group communication. Role switching involves adjusting the role of a device to function as either a receiver that receives

data from other devices or a sender that forwards data to other devices. This approach expands the range of targeted communication devices and the achievable communication distances.

For example, certain researchers concentrate on achieving seamless group reformation in the P2P network. The objective of these studies is to introduce a method for GO role transfer (Chaki et al., 2015; Kalbarczyk & Julien, 2018; Yasser et al., 2021). While the proposed solution successfully maintained the connectivity of D2D communication, it required additional processing effort from the mobile device due to calculations and analyses involved in selecting potential or backup peers as the GO, based on factors such as peers' network quality or other device performance capabilities, like energy level (Liu et al., 2016; Pan & Lin, 2018; Raj et al., 2014; Yi et al., 2020). Consequently, the decision was made not to adopt a mode-switching scheme in this research.

**c. Virtualization**

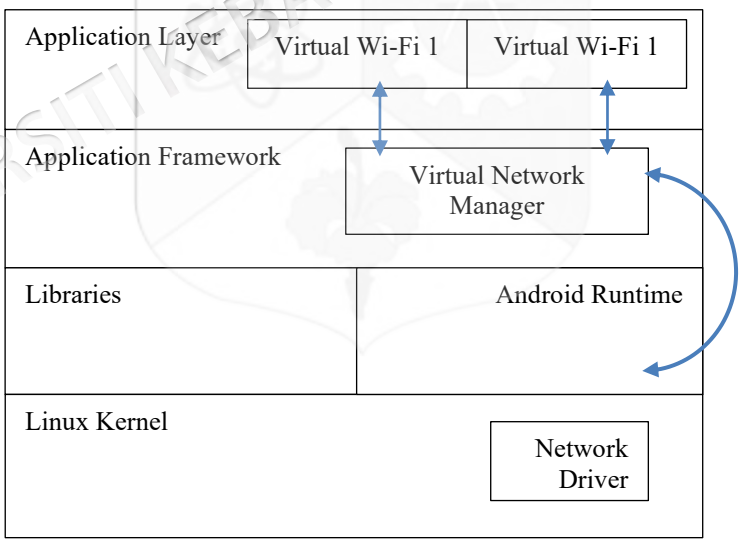



Figure 3.13 Virtualization of Wi-Fi network interfaces at application layer

The other possible way to solve the problem is to apply the virtualization concept. Since 2004, Ranveer & Paramvir have proposed MultiNet, which is an architecture that virtualizes a WiFi wireless card, enabling the computer to connect to different wireless networks simultaneously (Chandra et al., 2004). MultiNet was developed and tested specifically for the Windows platform and personal computers. Figure 3.13

represents the idea of network interface virtualization based on the Android platform. A virtual network manager will be integrated into the application framework layer of Android as the manager to provide abstractions of the network interface. The manager, situated between the application and kernel layers, can virtualize the Wi-Fi interface before it reaches the application layer, creating the impression that mobile devices have two Wi-Fi interfaces that can simultaneously connect to multiple networks.

Referring to Figure 3.12, with the virtualization feature implemented in device B, device C will be able to reach device A, thus forming the collaborative network. Although virtualization in the networking field is not a new research area, it is still innovative in terms of its specific implementation for compact mobile devices like smartphones. Several software packages with similar functionality to MultiNet are available, such as Connectify, OSToto, Baidu WiFi, Thinix, and Virtual Router, to name a few. To the best of the author's knowledge, all identified software is applicable to computers only. Limited processing power or memory resources on a compact mobile device, like a smartphone, could be the primary reason why virtualization is not being introduced.

Furthermore, introducing virtualization to the higher layer mostly necessitates modification of the lower layer source code in the system stack, requiring root access. Rooting is the process of gaining complete control over the device’s OS. Generally, device manufacturers do not support rooting, and rooting some devices will automatically void the manufacturer's warranty. Hence, a virtualization approach that requires rooting is not a feasible solution. The solution should avoid the need to modify or recompile the existing source code, enabling implementation on most devices in the market.

**d. Utilized existing protocol**

According to the literature review, some researchers have been exploiting existing wireless protocols to form WCN networks. The formation of WCN was realized through either implementing the API that is provided to mobile application developers

or modifying the underlying layer source code, which requires recompilation of the system source code. For example, M.-S. Pan & Wang (2019) and Yasser et al. (2021) have exploited the device discovery procedure of the WFD protocol as the carrier of the encoded data. The data was encoded into the probe request and response frames exchanged during the device discovery stage. In addition, some researchers also utilized the SD procedure in the WFD protocol to carry additional information along with the service instance created during the SD procedure (Aoude et al., 2018; Arnaboldi et al., 2017; Camps-Mur et al., 2013; Holzer et al., 2016; Hu & Cao, 2017; Junio & Chavez, 2018; Khawaja et al., 2020; Links et al., 2019; Pan & Lin, 2018; Sha et al., 2016; Shahin & Younis, 2015, 2015). The data carried along with the service instance was either the message to be delivered or additional data required for WCN establishment.

According to (Li et al., 2021), sending data via Wi-Fi Hotspot or WFD consumes a significant amount of energy. When the phone is in the idle stage, the average current is 123.47 mA. However, the average current consumption increased by 300.91% when the phone was sending data at 5 GHz and by 131% at 2.4 GHz. In the scan state of WFD, the phone consumes 103.85% and 100.81% of the current when operating on 5 GHz and 2.4 GHz frequencies, respectively. This indicates that the phone in the WFD state before connection on 2.4 GHz consumes the least current. Therefore, the solution should primarily focus on the WFD scan state, considering energy efficiency.

Ahmed A Shahin & Younis (2015) reported that the selection of the Group Owner (GO) will impact the lifecycle of an application. Hence, limitations such as random GO selection, lack of delegation, negotiation phase limitations, and absence of seamless reformation will influence the application's lifetime. Furthermore, Zhang et al. (2014) highlighted issues in the default implementation of WFD, including inaccurately reported RSSI values, a fixed intent value, and an unknown listen period for the GO. The authors also pointed out flaws in the default group formation scheme in the WFD protocol. As a result, the selected GO might not be the optimal candidate for efficiently disseminating messages along the optimal route. The increased connection duration on the GO has extended the time required for group formation

and joining. This was corroborated by Camps-Mur et al. (2013), who found that the total connection delay for WFD took a minimum of 4 to 10 seconds. Due to this, group formation and the GO selection scheme were avoided in this research, and the focus was directed toward the stage preceding GO negotiations.

### i. P2P discovery

The fundamental operations of a P2P connection with WFD are explained in section 2.3. Prior to establishing the connection with WFD, the P2P discovery procedure is executed. As depicted in Figure 3.14, the steps preceding GO selection encompass device discovery and service discovery. Device discovery is a mandatory step for devices to identify nearby peers. However, the service discovery procedure is optional. This section exclusively focuses on the detailed stages before GO selection or group formation, as outlined in the technical documentation of the P2P protocol (IEEE Computer Society, 2016; Wi-Fi Alliance, 2014, 2016). An example of the P2P discovery procedure illustrated between two devices can be found in Appendix A.

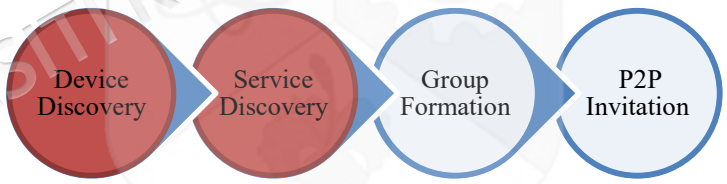



Figure 3.14 The process before group formation in a P2P connection with WFD

**Device discovery procedures**

The device discovery procedure employs probe request and response frames to exchange device information. Figure 3.15 illustrates the main device discovery procedures. The discovery procedure consists of two major phases: "scan" and "find". The scan phase allows devices to identify the optimal channel to form a P2P group. This is realized by scanning all supported channels on a device to collect information about the peers or networks. The find phase is implemented to ensure that devices executing device discovery procedures come to the same channel so that the

communication session can begin. During the find phase, devices will alternate between "listen" and "search" states after an interval time noted as 100 TU, where TU (time unit) is a random integer between 1 and 3 or fixed by the device developer. A device in the listen state listens to probe request frames from other devices, while a device in the search state sends one or more probe request frames on the social channels (1, 6, 11). The overall duration of each phase is implementation dependent. A sample illustration of the device discovery procedure between two devices can be viewed in Appendix A.

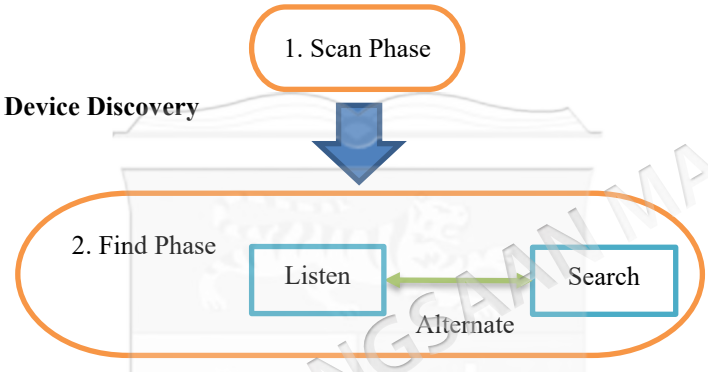



Figure 3.15 Stages in device discovery procedure

The format of a probe request frame is illustrated in Table 3.5. The probe request frames sent by a device in search state include two information elements (IE) named Wi-Fi Simple Configuration (WSC) IE and P2P IE. Table 3.6 lists the attributes of WSC IE for the probe request and response frames, and Table 3.7 lists the attributes of the P2P IE for both frames.

Table 3.5 Probe Request frame format

| Order | Information Element | Note |
|---|---|---|
| | WSC IE | The WSC IE shall be present in the frames transmitted by a P2P Device |
| Last | P2P IE | The P2P IE shall be present in the frames transmitted by a P2P Device |

Source: (Wi-Fi Alliance, 2016)

Table 3.6 Attributes of WSC IE

| ID (Type) | Notes |
|---|---|
| 0x1021 | Manufacturer |
| 0x1023 | Model Name |
| 0x1024 | Model Number |
| 0x102C | Out-of-Band Device Password |
| 0x103C | RF Bands |
| 0x1042 | Serial Number |
| 0x1047 | UUID-E |
| 0x1049 | Wi-Fi Alliance Vendor Extension(includes Version2 sub element) |

Source: (Wi-Fi Alliance, 2016)

Table 3.7 Attributes of P2P IE

| Attributes of P2P IE for Probe Request frame | | Attributes of P2P IE for Probe Response frame | |
|---|---|---|---|
| **ID** | **Attributes** | **ID** | **Attributes** |
| 2 | P2P Capability | 2 | P2P Capability |
| 3 | P2P Device ID | 8 | Extended Listen Timing |
| 6 | Listen Channel | 12 | Notice of Absence |
| 8 | Extended Listen Timing | 13 | P2P Device Info |
| 13 | P2P Device Info | 14 | P2P Group Info |
| 17 | Operating Channel | 25 | Advertised Service Info |
| 21 | Service Hash | | |

Source: (Wi-Fi Alliance, 2016)



Based on the results of the scan and find phase, a device will either join or not join a P2P group. A device will join another group based on the discovered GO, either using the WSC data obtained or connecting directly to the provisioned GO. If the device is not joining another P2P group, it can perform any one of the three operations:

1. The device can send a request to another P2P device to join a P2P group.
2. The device can send a request to another P2P device to activate the persistent P2P group.
3. The device initiates the GO negotiation procedure to form a new P2P group.

If none of the three operations is executed, the device can choose to initiate the service discovery procedure.

**Service discovery (SD) procedures**

The SD procedure is an optional process that can be executed after the discovery (D) procedure is complete. It is extensible and works with higher-layer service advertisement procedures, such as Bonjour[13] and UPnP protocol[14]. Through the SD procedure, a P2P device can identify the list of services, information about single or multiple services, and updates on services offered by the device. The service provider (e.g., host device) utilizes SD procedures to advertise the service offered in the P2P network. Additionally, the SD procedure enables devices to advertise additional information alongside the advertised service. The client or target device with similar interests discovers the service and then utilizes the collected information for subsequent operations. An illustrative example of the SD procedure between two devices can be found in Appendix A.

For internal data exchange operations, the SD procedure utilizes the Generic Advertisement Service (GAS) protocol, as defined in the IEEE 802.11 protocol document (IEEE Computer Society, 2016). By employing the GAS protocol, public action frames serve as the transport for SD procedures, involving one or multiple GAS initial request and response action frame exchanges. When a device initiates the SD, the SD request frame, which initiates GAS initial request frames is prepared and sent from the P2P device. The targeted P2P device, upon detecting and interpreting the request, responds with the SD response frame, driving GAS initial response frames. As outlined in the protocol document, the GAS initial response frame used by the SD

[13] *Bonjour*. (n.d.). Apple Developer Documentation. https://developer.apple.com/documentation/foundation/bonjour

[14] *OCF - UPnP Standards & Architecture*. (2020, April 24). Open Connectivity Foundation (OCF). https://openconnectivity.org/developer/specifications/upnp-resources/upnp/

response frame accommodates vendor-specific content with distinct fields. Consequently, the SD response frame might include additional Service Response Type Length Value (TLV) information. Should the SD response frame exceed the GAS initial response packet size, the GAS fragmentation procedure is triggered. This procedure employs a GAS comeback request and response frame to identify and regenerate the fragmented frames. A detailed breakdown of the GAS initial request frame is available in Appendix B, while the detailed breakdown of the GAS initial response frame can be found in Appendix C. An analysis of GAS public action frames revealed that the protocol doesn't impose size restrictions on the frame, given that the last attribute of the frame contains vendor-specific content. Within this attribute, the last TLV is denoted as response data, and its size varies. Nevertheless, it was presumed that a standard or recommended size existed for protocol implementation. Thus, the protocol document RFC 6763, Domain Name Based (DNS) SD, was examined to ascertain the recommended size for the service instance.

**ii. Domain Name Based (DNS) Service Discovery**

In terms of implementation, the DNS SD protocol provides two objects to represent a service instance to be published by the host device: DNS Service (DNS SRV) and DNS Text (DNS TXT) records. The SRV record provides information about the target host and port, while the TXT record offers additional details about the service instance in a key-value pair structure.

The structure of an SRV record = <Instance>.<Service>.<Domain>

- The <Instance> portion is stored as a DNS label and limited to 63 octets, with a maximum of 15 Unicode characters or 21 Kanji characters.
- The <Service> portion specifies what the service does and consists of a pair of DNS labels, hence being limited to 126 octets.
- The <Domain> portion specifies the DNS subdomain and is limited to 255 octets.

> Based on the data retrieved, the total size of the SRV record should be: 63 + (2*63) + 255 = 444 octets

The TXT record provides the facility to add additional information about the service instance. Every DNS SRV record will be paired with a DNS TXT record, even if there is no additional data to store. The general format for a DNS TXT record is a key-value pair. The length of the key should be fewer than nine characters, and the key value is not case-sensitive. The value can contain opaque binary data, US-ASCII, or UTF-8 characters. Based on RFC 6763, the total size of a typical DNS TXT record should be 200 bytes or less to ensure packet transfer efficiency. Although S. Cheshire & Krochmal (2013:11) reported

> Txt record can be up to 65535 bytes long, when using Multicast DNS [RFC6762] the maximum packet size is 9000 bytes, including the IP header, UDP header, and DNS message header, which imposes an upper limit on the size of TXT records of about 8900 bytes.

The authors also explained that

> In practice the maximum sensible size of a DNS-SD TXT record is smaller even than this, typically at most a few hundred bytes.
>
> Source: S. Cheshire & Krochmal (2013)

To further investigate the size limit of each DNS TXT record, the technical documents RFC 1034[15] - Domain Names Concepts and Facilities, and RFC1035[16] - Implementation and Specification were studied. In RFC 1034, Mockapetris (1987) proposed that "the format of each string within the record is a single length byte,

---

[15] Mockapetris, P. (1987a, November 1). RFC 1034: Domain names - concepts and facilities. https://www.rfc-editor.org/rfc/rfc1034

[16] Mockapetris, P. (1987, November 1). RFC 1035: Domain names - implementation and specification. https://www.rfc-editor.org/rfc/rfc1035

followed by 0-255 bytes of text data." Therefore, it is confirmed that each individual record in the DNS-SD TXT record group should be less than 256 bytes long. This is a very important piece of information, as the model developed utilizes the DNS-SD TXT record as the carrier of the data to be exchanged between nodes.

After a detailed review of the protocol document, it was observed that the SD procedure provides the functionality to facilitate the data exchange between devices, and the Android platform has readily provided the API[17] for the SD procedures. Therefore, it was decided that the model would be developed to exploit the functionality of the SD procedures to realize the P2P data exchange operations in WCN. The details about the implementation of the SD procedure in terms of coding are discussed in the following section.

### 3.4 MODEL DEVELOPMENT METHOD(OBJ2)

The second objective involves the development of a prototype that will be tested on the Android platform. Hence, the prototyping and development approach were adopted. The prototyping approach is a model in which iterations of the build, test, and rework phases are conducted until an acceptable prototype is achieved. The prototyping phases implemented in this research are depicted in Figure 3.16.

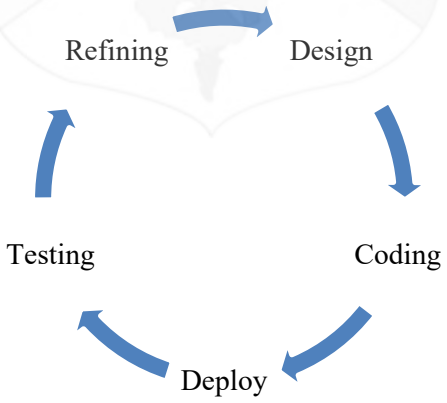



Figure 3.16 The iterative process of the prototyping model implemented

[17] *Use Wi-Fi Direct (P2P) for service discovery*. (n.d.). Android Developers. https://developer.android.com/training/connect-devices-wirelessly/nsd-wifi-direct

The development of the prototype had gone through six major iterations in total. Each iteration includes new features or modifications to the prototype to form the final model for P2P communication. The features added or modified in each iteration are listed in Table 3.8. The Android Studio IDE was used for the prototype's development. Debugging tools were utilized during the development process to assist in troubleshooting errors in the code. During the development process, a quick unit test was conducted to ensure the procedures operate as expected. At the end of each iterative cycle, an integration test was conducted to check the overall operations of the model.

Table 3.8 The feature introduced, and outcome generated from each iteration for prototype development

| Features added or modified in each iteration | Outcome |
|---|---|
| Basic P2P connection | A P2P connection was established between two devices. |
| Implement service discovery | Device A was able to publish a service, and Device B was able to discover the service |
| Modify service discovery protocol | Messages were encoded into the SRV instance, and peer devices were able to receive the messages. |
| Introduce multi-hop data exchange | The store-carry-forward mechanism was implemented. |
| Implement multi-threading | The operations of delivery and discovery have been separated into multiple threads. |
| Implement abstraction | The main operations were grouped and formed a level of abstraction for code reusability. |

### 3.4.1 Data Exchange Model

Lee & Sulaiman (2019) reported that connectivity preservation is one of the challenges in forming a robust WCN. Therefore, a mobile device in a WCN should be equipped with bridging capability, as discussed in section 2.1.4. Intragroup communication is possible with the default P2P communication framework on mobile devices. However, intergroup communication is not possible on an off-the-shelf device due to the lack of routing ability by default. Furthermore, multi-hop

communication requires an addressing and group management scheme, enabling each device to identify and communicate with the other end devices.

In WFD, only the GO obtains information about the GMs and their states. When a GM is leaving or a new member is joining the group, only the GO will be aware. In other words, other GMs will not be aware of the presence of additional members in the group (Li et al., 2020). It was noted that Android provides the API for the SD procedure to facilitate data exchange between devices even before a full connection is established. Therefore, the data exchange model was developed by leveraging the SD protocol and utilizing the API provided by Android. Additionally, the concepts of abstraction and middleware were adopted and identified as agents in this research.

**a. Conceptual architecture view**

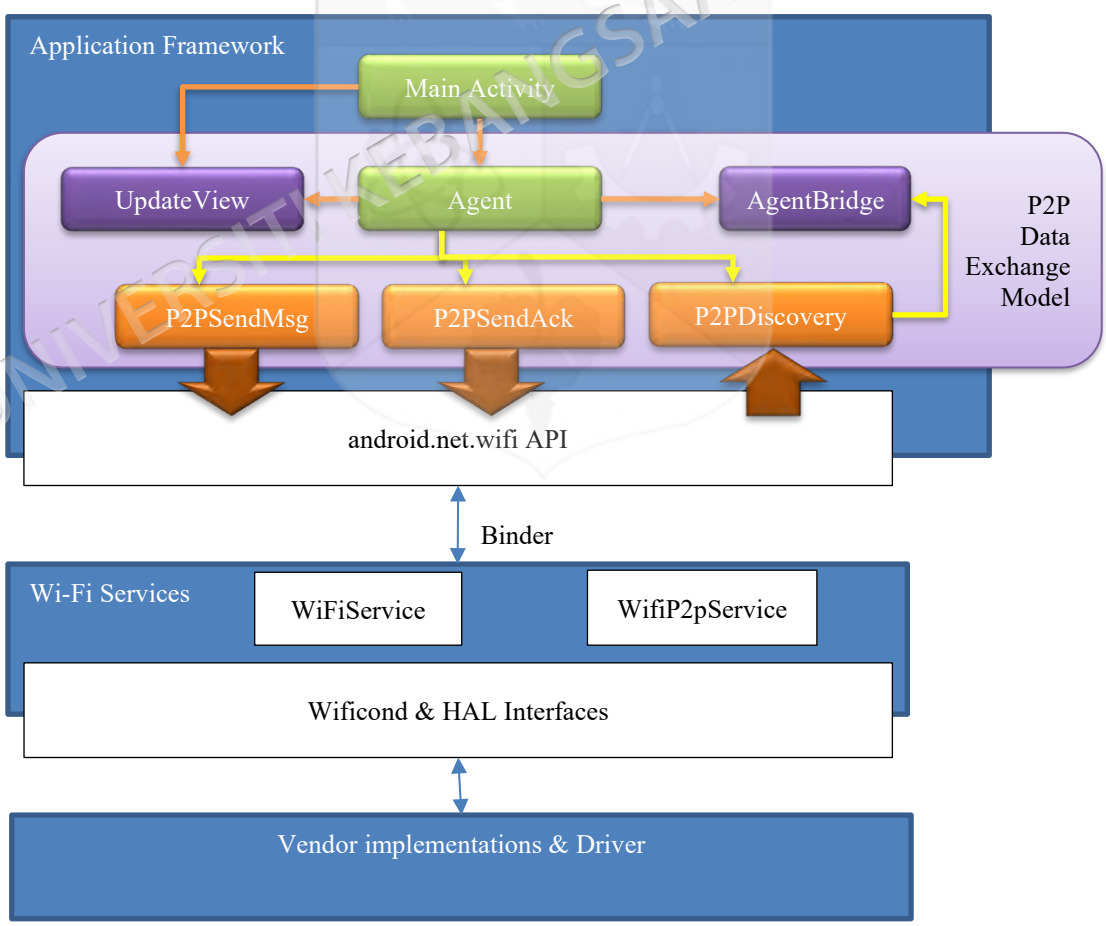



Figure 3.17 Conceptual architecture for the model developed

Figure 3.17 shows the conceptual view of the P2P data exchange model that was developed on top of the Android Wi-Fi architecture. The execution of the model was developed at the application framework layer and utilizes the API provided by the android.net.wifi package to link to the underlying layer of the Wi-Fi framework. When the object from the application framework calls the methods of the API, the binder that is located between the application framework and the Wi-Fi services layer triggers the Wi-Fi services through the Binder IPC mechanism. Next, the Wi-Fi services send the relevant signal for the final physical hardware process execution through a binder or using either the Android Interface Definition Language (AIDL) or HAL Interface Definition Language (HIDL). In the model developed, the abstraction concept was applied, where the operation of P2P data exchange was embedded in the agent object. The detailed working of the model is explained in the following section, where the class diagram of the model is presented.

**b. Class diagram**

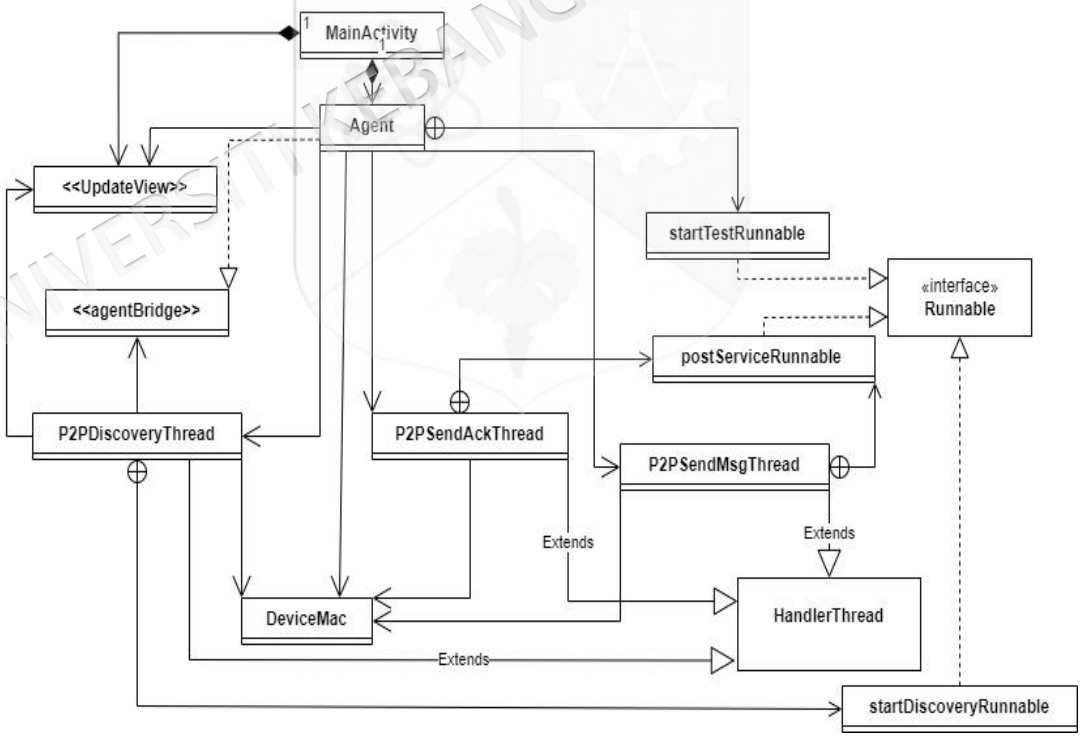



Figure 3.18 Class diagram of the model developed

Figure 3.18 depicts the class diagram of the new P2P communication model mapped to the conceptual architecture. Six main classes are created, including the

MainActivity class for a standard Android application, and two interfaces for data passage from the lower layer to the upper layer of the model. The middleware concept is visualized through the Agent class implementation. The underlying operations of P2P communication with the WFD Framework have been abstracted by the Agent class. Thus, the MainActivity only needs to manage UI object events generated from the end user's actions, while P2P data exchange operations are managed by the Agent object.

When data needs to be delivered, the Agent interacts with the P2PSendMsgThread and P2PSendAckThread. Upon data reception, the P2PDiscoveryThread handles the message, passing the data to the Agent through the AgentBridge interface and to the MainActivity through the UpdateView interface. The following sections explain the functions of each introduced class in the developed model


Figure 3.19 MainActivity class for the UI operations

Figure 3.19 depicts the attributes and overridden methods of the MainActivity class. MainActivity is the standard class that all Android applications must inherit. It

provides the standard methods to be overridden and serves as a means to standardize the execution of the internal methods.

For the prototype developed, only the onCreate() and onDestroy() methods are overridden. The onCreate() method groups and initiates all the UI components, such as Button, EditText, TextView, ScrollView, and ProgressBar, as depicted in Appendix D. The listeners are created at the local scope of onCreate() to handle the events triggered by the UI components.

As illustrated on the class diagram in Figure 3.18, the classes that are connected to MainActivity include the Agent and UpdateView classes. Therefore, in the MainActivity class, the objects myAgent for the agent and mainUpdateView for UpdateView are created.

Once the user has entered the message into the EditText (input_message) component and pressed the send Button, the postMessage() method of myAgent will be invoked. When there are UI interface update activities, the methods of the UpdateView object will be triggered, and the UI components will then be updated accordingly. The code snippet for the listener method is illustrated in Appendix D.1.

Figure 3.20 depicts the attributes and methods of the Agent class. The Agent class stands as the core forming the data exchange model, encompassing the most attributes and methods for coordinating data exchange operations. Within the multithreading paradigm, the Agent class serves as one of the threads that extends the HandlerThread[18]. The HandlerThread finds its foundation in the built-in Android class, providing the mechanism for queuing and passing messages. These messages comprise a series of runnable tasks directed to the queue through a handler. The messages loaded into the message queue execute sequentially, guided by the looper, within the thread. Upon the creation of an Agent object, the onLooperPrepared() method triggers, initializing the handler object for the Agent. The Agent class extends

[18] HandlerThread |.(n.d.). Android Developers. https://developer.android.com/reference/android/os/HandlerThread

the HandlerThread primarily to oversee experimental testing procedures. These procedures encompass methods such as startTest(), stopTest(), runTest(), and getNextMessage(), as depicted in Appendix D.2


Figure 3.20 Agent class that coordinates the P2P operations

The pDiscovery, pSend, and pAck are the key objects created in the constructor of the Agent class. The methods startDiscovery() and stopDiscovery() were created to command the pDiscovery thread to initiate message discovery procedures. When a message is sent to the Agent through the postMessage() method,

the pSend object passes the message to the P2PSendMsgThread class for delivery. Incoming messages to the Agent originate from the agentBridge interface implementation. These incoming messages can either be of type 'message' from the BCmessageReceived() method or acknowledgement messages from the ackmessageReceived() method. Upon receiving a message from the BCmessageReceived() method, the postAck() method is called to dispatch an acknowledgement message through the pAck thread. In the event of an acknowledgement message received from the ackmessageReceived() method, the clearService() method is invoked to clear the service instance created by the pSend thread. When the need arises to reset all services created by the pSend thread, the clearAllServices() method is employed. Appendix D.3 provides code snippets illustrating the postMessage() and postAck() functions, which trigger the message posting procedures for the message delivery threads, namely P2PSendMsgThread and P2PSendAckThread. Finally, the cleanup() method is invoked, performing necessary clean-up operations when the application is closed or terminated by the Android system.


Figure 3.21 P2PSendMsgThread class that manage the data delivery operations

Figure 3.21 illustrates the attributes and methods of the P2PSendMsgThread class. The primary function of this class is to interact with the Wi-Fi API for message delivery operations. The message delivery operation is achieved by utilizing the API from the WFD framework. The central method in the P2PSendMsgThread is the postMessage() method. Referring to Appendix D.4a.), when the postMessage() method is invoked and received, the Message object will be passed to the postServiceRunnable class, which implements the Runnable interface for message delivery procedures. The instance of the postServiceRunnable class created will then be dispatched to the message queue of the P2PSendMsgThread class via the handler object for execution.

The novelty of the P2P data exchange model lies in the mechanism by which the message to be delivered is encoded into the service info object of the WFD protocol. This allows the message to be published along with the services advertised by the device. When a peer device detects the published services, it decodes them and retrieves the original message. As explained in 3.3.3d.ii, each record's maximum size in DNS-SD TXT should be less than 256 bytes. The control of the message size limit has been set in the listener for the 'send' button object in the MainActivity class. The message encoding process occurs in the 'run()' method of the 'postServiceRunnable' object. The messages are encoded in a HashMap class object, providing the required key-value pair structure for the service information records. Referring to Appendix D.4b.), the record object from the HashMap class is created, and the key-value pair data, including the "message," is included in the record using the 'put()' method. Subsequently, a new instance of the 'WifiP2pDnsSdServiceInfo' object is created using the 'newInstance()' method. Note that the record object is included as a parameter of the 'newInstance()' method.

Following the creation of the service info object, the service info will be added to the Wi-Fi P2P framework via the 'addLocalService()' method of the 'WifiP2pManager' class, as depicted in Appendix D.4c.). The 'WifiP2pManager' class also provides the 'removeLocalService()' method to remove the service instance from the framework, as listed in Appendix D.4d.).

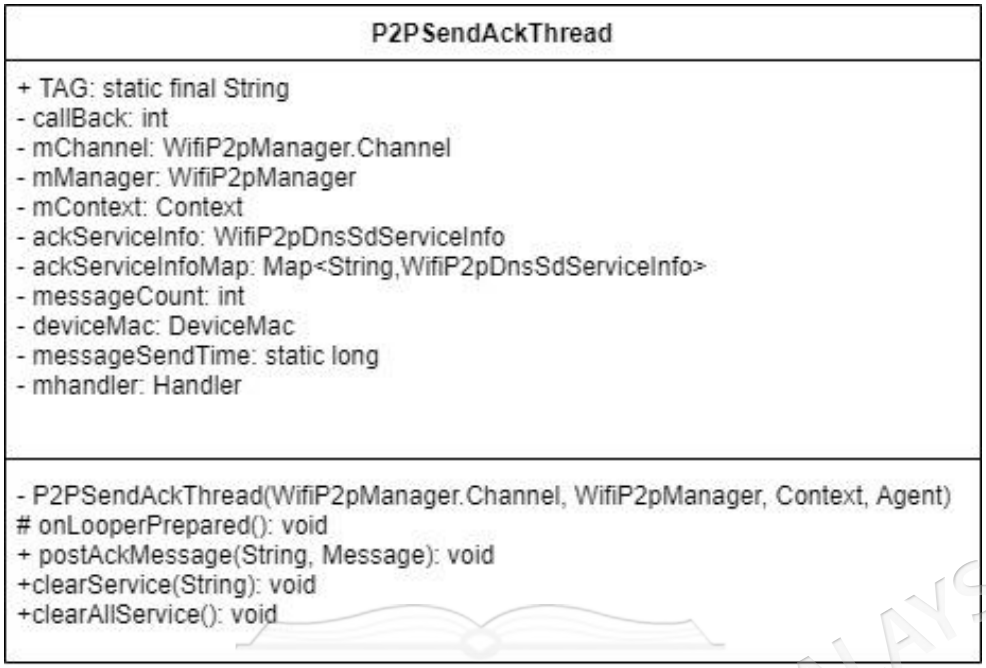



Figure 3.22 P2PSendAckThread class that manage the acknowledgement message delivery operations

Figure 3.22 depicts the attributes and methods of the P2PSendAckThread class. The primary function of this class is to interact with the Wi-Fi API for acknowledgement (ack) message delivery operations. The ack message is a message type introduced in the model. The ack message is automatically triggered by the agent upon receiving and decoding a message sent by a peer. This ensures that the sender receives an ack message as a notification to remove the service info instance created within the framework. The structure of the ack message record is illustrated in Appendix D.5a.). Since the ack message serves as an acknowledgement, there are only three pairs of records in the HashMap. As elaborated in section 2.5.1the identified research gap is the need to introduce a bidirectional data exchange procedure. This research gap is addressed through the implementation of the ACK message delivery. Similar to the P2PSendThread, the service instance created and registered with the Wi-Fi P2P framework should be removed to prevent overloaded services within the framework. Therefore, the proposed solution involves clearing the service at intervals. Referring to the code snippet in Appendix D.5b.), the postDelayed() method of the Handler class is utilized with the second parameter named callBack. The callBack variable stores an integer value calculated based on the average message delivery time between two peers, as determined from the functional testing procedure.


Figure 3.23 P2PDiscoveryThread class that manage the data discovery operations.

Figure 3.23 depicts the attributes and methods of the P2PDiscoveryThread class. The main function of this class is to discover and gather messages, then pass them to the Agent through the agentBridge interface. The model starts and stops the discovery process by calling the startDiscovery() and stopDiscovery() methods in the Agent, as illustrated in Appendix D.6a.). Apart from agentBridge, the P2PDiscoveryThread object also possesses access to the UpdateView interface to update the ProgressBar component in the MainActivity class. When the Wi-Fi service is not ready, the status will be sent from the Wi-Fi framework to the listener in discoverService(). Then, the P2PDiscoveryThread will update the ProgressBar component via UpdateView. Appendix D.6b.) depicts the code snippet in the discoverService() method that calls WifiP2pManager.discoverServices() to start the discovery process. The P2PDiscoveryThread receives the event update from the Wi-Fi framework and passes the update to UpdateView.

Arnaboldi et al. (2017) reported that the SD procedure runs every 2 minutes when the WFD connection is activated. In terms of the API, WifiP2pManager.discoverServices() is a one-off procedure. Therefore, a callback mechanism is needed for the model to continuously execute the discovery process. The proposed solution utilizes the message queue and looper components of the Thread class to repeat the discovery procedure. The WifiP2pManager.discoverServices() procedures are included in the discovery() method, as shown in Appendix D.6b.). Referring to Appendix D.6c.), the discovery() method is included in the discoveryRunnable runnable class so that it will be executed by the P2PDiscoveryThread. The callback mechanism is realized through the Handler's postDelayed() method with the callBackTime interval. When the runnable task posted in the message queue is executed by the P2PDiscoveryThread, the discovery() method will be invoked. Upon the successful execution of the WifiP2pManager.discoverServices(), the discoveryRunnable will be posted to the message queue again via the handler.

The P2PDiscoveryThread implements the WifiP2pManager.DnsSdServiceResponseListener and WifiP2pManager.DnsSdTxtRecordListener classes. However, only WifiP2pManager.DnsSdTxtRecordListener is utilized because the encoded data is placed in the DNS TXT object. Appendix D.6d.) depicts the code snippet where the listener is initiated. Appendix D.6e.) demonstrates the code snippet that manages the encoded message belonging to a sender message category. In this case, the message will be forwarded to the Agent for UI updates, broadcast, and triggering the P2PSendAckThread to send an acknowledgment message. If the same message has been encoded before, the broadcast and UI update operations will not be executed, and only an acknowledgment message will be sent. Appendix D.6f.) demonstrates the code snippet that manages the broadcast message category. If the broadcast message received was originally sent by the device itself, the clearService() function of the agentBridge will be invoked to remove the service instance from the Wi-Fi framework. On the other hand, if the received broadcast message is new, it will be sent to the agentBridge for broadcasting. Simultaneously, an acknowledgment message will be sent to the original sender to indicate that the broadcast message has

been received. Lastly, if the broadcast message has already been received, an acknowledgment message will be sent to the original sender to signal the cessation of broadcasting the message. Appendix D.6g.) demonstrates the code snippet that handles the acknowledgment message category. The initial step is to ensure that the ACK message is intended for the device by checking the MAC address, which has been encoded in the record list. If the MAC address matches that of the device, the message will be passed to the agentBridge to retrieve the UID and then remove the service instance from the Wi-Fi framework. Acknowledgment messages that have been received before or do not belong to the current device will be ignored.

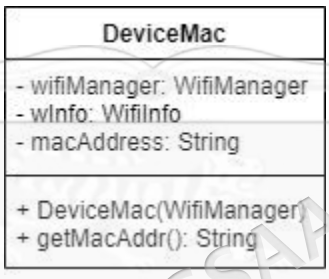



Figure 3.24 DeviceMac class to manage devie MAC address retrieval operations

Figure 3.24 Figure 3.24 depicts the attributes and methods of the DeviceMac class. The main function of this class is to retrieve the MAC address of the device that invoked the getMacAddr() method.


Figure 3.25 agentBridge interface for the implementing class to revert data back to Agent object

Figure 3.25 depicts the attributes and methods of the agentBridge interface. The primary function of this interface is to enable the P2PDiscoveryThread object to convey the status of the discovered message to the Agent class. The messageReceived() method oversees the procedures to execute when a new message is received. The ackmessageReceived() method manages the procedures to be executed when an acknowledgment (ack) message is received. The

BCmessageReceived() method handles the procedures to be executed upon receiving a broadcast (BC) message. Lastly, the clearService() method notifies the Agent class to dispatch a command to the P2PSendMsgThread, thereby removing a service instance based on the recorded UID.

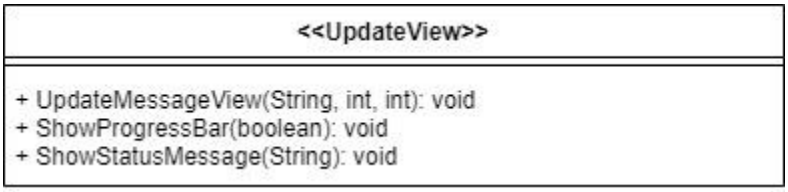



Figure 3.26 UpdateView inferface for the implementing class to update objects on the UI

Figure 3.26 depicts the attributes and methods of the UpdateView interface. The main function of this interface is to allow the Agent and P2PDiscoveryThread to invoke the methods that will update the UI component managed by MainActivity. The UpdateMessageView() method manages the procedures to display messages on the message box. The ShowProgressBar() method manages the procedures to update the ProgressBar component on the UI, representing the progress of message discovery. The ShowStatusMessage() method manages the procedures to display the status message of the Wi-Fi connection.

It is crucial that every message has a unique ID (UID) so that the recipient device can identify the source device and also filter and delete messages that have already been received. Additionally, when a message is disseminated in a WCN, the UID allows the original sender to independently drop the message. The UID consists of two parts, separated by an '@' symbol. For example, 48:88:ca:08:d2:4c@1671892756089. The first part is the MAC address of the sender, in this case, 48:88:ca:08:d2:4c. The second part of the UID is the timestamp when the message is generated; in this case, 1671892756089, based on the sample UID. The UID is encoded as one of the records in the SRV TXT, with the key fixed as 'UID'.

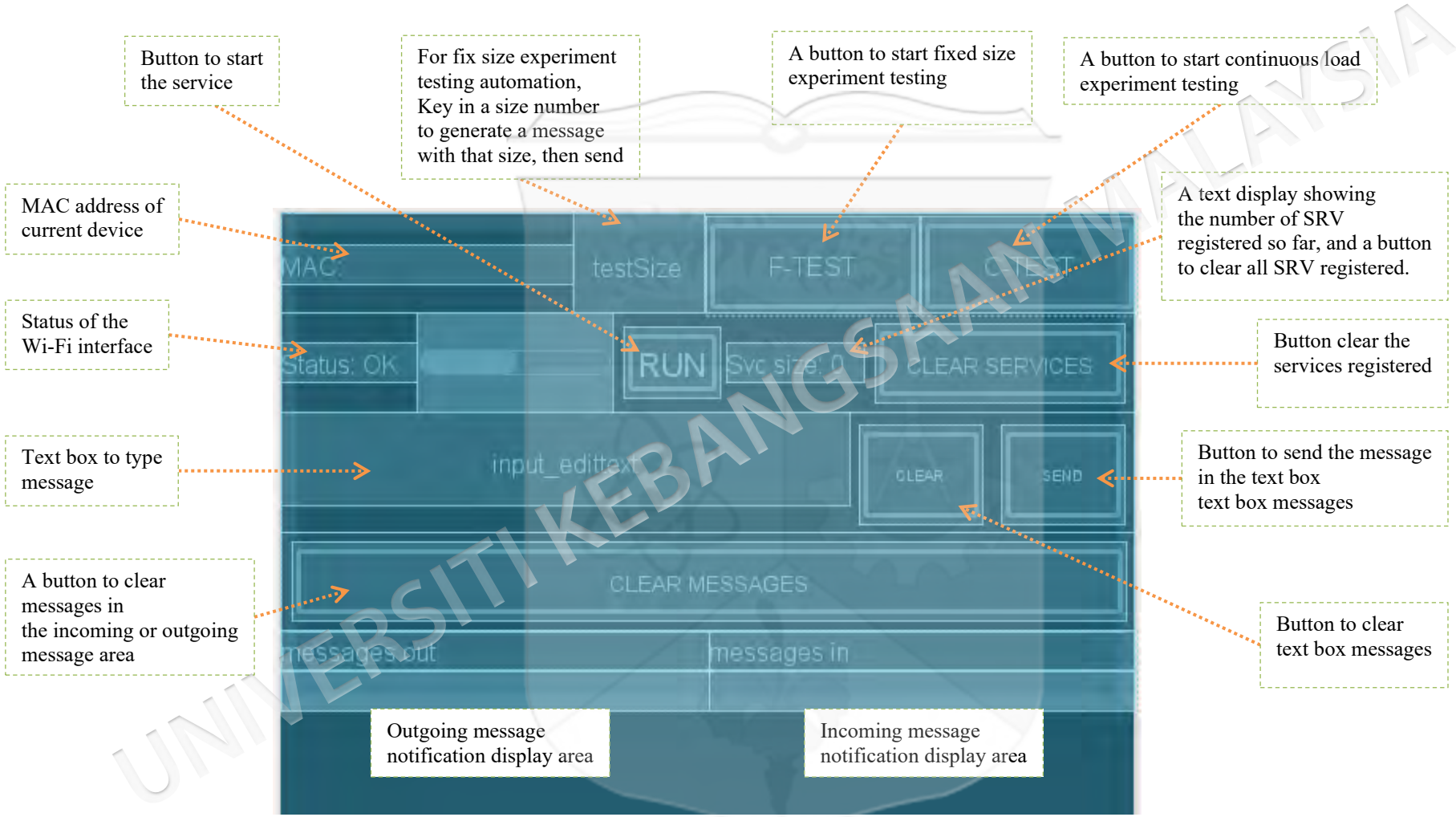



Figure 3.27 The user interface for the final prototype with labels

**c. User interface of the final prototype**

Figure 3.27 depicts the wireframe of the final prototype designed, featuring labels describing the objects on the interface. To utilize the prototype, ensure that the Wi-Fi interface on each device is activated and toggle the 'RUN' button. The straightforward steps for sending a message to a peer are outlined below:

1) Ensure that the Wi-Fi interface is enabled.

2) Once the status turned “OK”, click the “RUN” button.

3) Enter the message in the text box.

4) Click “Send”

The sent message will be shown at the bottom left, under the label "message out," while the received message will be displayed at the bottom right, under the label "message in." The buttons labeled as "F-TEST," "C-TEST," and "CLEAR SERVICES" are intended for experimental testing purposes, as elaborated in 3.5.2c.

## 3.5 DEPLOY, MEASURE, AND EVALUATE(OBJ3)

This section explains the methods for completing the third objective, which is to measure the performance of the developed model. The main goal of the evaluation is to provide proof of concept for the model. The methods for performance measurements were adapted from the literature gathered in the scoping review presented in section 4.2, as well as from the performance measurement for network interconnect devices benchmarking document RFC2544[19]. Specifically, the test cases were planned based on the literature, and the measurement parameters were identified from the literature and confirmed according to the benchmarking document.

[19] Bradner, S. (1999, March 2). RFC 2544: Benchmarking Methodology for Network Interconnect Devices. https://www.rfc-editor.org/rfc/rfc2544

### 3.5.1 Testing

The testing process was conducted throughout the development of the model, following the adoption of the prototyping development model. Unit testing and integration testing were carried out at the conclusion of each prototyping cycle. Unit testing aimed to assess the functionality of all functions and modules. This was achieved by deploying the prototype to an actual phone and evaluating the functionality of each UI component. Integration testing, on the other hand, focused on validating interactions between modules. Following the completion of each prototyping cycle, the prototype was deployed on a minimum of two smartphones. Data exchange procedures were then executed ten times to ensure the functionality of the model.

Finally, integration testing was performed on the ultimate prototype. The subsequent section provides an explanation of the testbed and experiment setup.

### 3.5.2 Testbed & experiment setup

There were three smartphones and three laptops used for the experiment. The final prototype was installed on all three smartphones. The device's date and time were manually synchronized in the device settings before conducting the test. During the experimental testing, the smartphones were connected to the laptop with the Android Studio IDE running. The Android Studio's Logcat tool was used to record the log generated by the smartphone during testing. The log data was imported into the Microsoft Excel spreadsheet for tabulation and analysis.

#### a. Device specification

Table 3.9 lists the specifications of the smartphones used. There are two Lenovo smartphones and one Asus smartphone. All smartphones support Wi-Fi 802.11b/g/n protocol and are equipped with BT version 4.0. Table 3.10 lists the specifications of the laptops used. There are two HP laptops and one Lenovo laptop. All laptops are equipped with a built-in Wi-Fi card and have at least 8GB of RAM.

Table 3.9 Specification of the smartphones used in the experiment testing

| Brand and model | Asus Zenfone GO/X013D | Lenovo A5000 | Lenovo Vibe C A2020a40 |
|---|---|---|---|
| Android version | 5.1.1 | 4.4.2 | 5.1.1 |
| Year manufactured | 2016 | 2015 | 2016 |
| RAM | 2GB | 1GB | 1GB |
| Wi-Fi | 802.11b/g/n | 802.11 b/g/n | 802.11 b/g/n |
| Wifi Direct | Yes | Yes | Not enabled |
| BT & Profile | Version 4.0 A2DP, EDR | Version 4.0 A2DP | Version 4.0 A2DP |
| NFC | No | No | No |
| Photo of the device |  |  |  |

Table 3.10 Specification of the laptops used in the experiment testing.

| Brand and model | HP Pavilion dv4 1038TX | HP Pavilion 14 V228TX | Lenovo ThinkBook 14 G2 ITL |
|---|---|---|---|
| RAM | 8GB | 16 GB | 8GB |
| Processor | Intel® Core™ 2 Duo P7350 @ 2.00GHz | Intel® Core™ i5-5200U CPU @ 2.20GHz | 11th Gen Intel® Core ™ i5-1135G7@2.4GHz |
| Wi-Fi adapter | Intel® Wi-Fi Link 5100 AGN | Broadcom BCM43142 802.11 bgn | Intel® Wi-Fi 6 AX201 |
| Photo of the device |  |  |  |

**b. Parameters for the performance measurement**

The measured parameters were based on the scoping review literature and the benchmarking document RFC2544. The complete list of parameters relevant to the DTN P2P network's measurement has been provided in CHAPTER II, Table 2.1. However, not all measurement parameters were related to the performance measurement in this context. The pertinent parameters include goodput (bps), latency (ms), also known as delivery delay, and frame loss rate (%), also known as delivery ratio. As reported by Li et al. (2020), the goodput measurement is used to evaluate the performance from layer 4 to layer 7 of networking standard protocols. Therefore, the goodput of each test case will be measured using the formula:

$$goodput = \frac{message\ load(bits)}{transmission\ time(s)} \quad \ldots(3.1)$$

where the transmission time is:

$$transmission\ time = destination\ time(ms) - source\ time(ms) \quad \ldots(3.2)$$

and the average transmission time:

$$average\ transmission\ time = \frac{\sum_{i=1}^{n} transmission\ time\ set\ n}{n} \quad \ldots(3.3)$$

According to S. Bradner & McQuaid (1999), latency is defined as timestamp B minus timestamp A. Therefore, it is assumed that latency is equal to the transmission time parameter as utilized in the article by Lee & Sulaiman (2016). Finally, the parameter frame loss rate is calculated based on the RFC2544 benchmarking document, which is:

$$frame\ loss\ rate = \frac{(input\ count - output\ count) * 100}{input\ count} \quad \ldots(3.4)$$

Based on the benchmarking document (RFC2544), the test for calculating latency must be performed at least 20 times, and the final value reported should be the average of the values recorded. Therefore, the execution for each set of test cases was carried out at least 20 times to calculate the average of each set.

**c. Test cases**

The main objective of the test cases was to evaluate the performance of the proposed model. Two test case sets were set up for testing. The first set of test cases focused on the performance of data exchange between two smartphones. The second set of test cases focused on the performance of the multi-hop or storing and forwarding mechanism introduced.

**i. Test case 1- Data exchange**

For this test case, the main goal was to measure the performance of the data exchange procedure between two devices: the Lenovo A2020 and the Lenovo A5000. The devices were separated by 1 meter. A series of various messages were exchanged between the devices, and the timestamps were recorded using the Android Studio logcat tool. The message load represents the total message length or size in bytes. The messages eventually stored in the SD-TXT record are ASCII-based characters. Thus, a character is equivalent to one byte, and a message length of 60 is equal to 60 bytes. As discussed in 3.3.3d.ii. the optimal size of a record with key-value pairs should be less than 256. The string "message" was employed as the key; therefore, the largest tested message load must be under 248. The first test case consists of two sets of test cases. The first set was designed to test data exchange operations by sending a fixed-size message load. The second set was intended to test data exchange operations by sending an increasing message load at a fixed rate.

- Constant message load testing

Based on the SRV TXT record size limits of 248, a test case group was created with four different message loads (60, 120, 180, and 240). For each set of test cases, the fixed-size message was sent from device A 105 times, followed by

sending the same fixed-size message from device B 105 times. Each test recorded in the log includes the date, timestamp, tag, test number, UID message, and message length.

- Constant increment message load testing

For the constant increment message load testing, the message was sent by a device with a constant increment size of 10. For example, the first message to be loaded was 10, followed by 20, 30, and so on, up to 240. The test case was executed ten times to collect the log record with the same logging information as described above.

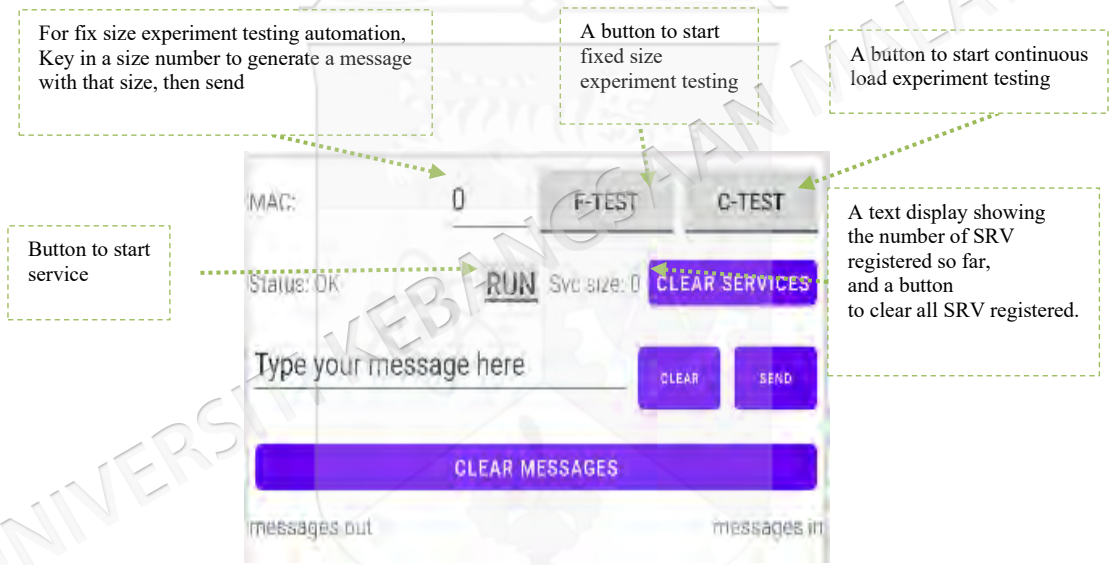



Figure 3.28 The UI components created for experiment testing

Three main functions were designed and implemented to automate the experiment testing process. The source code of the underlying operations of the UI components is listed in Appendix D.2. Referring to

Figure 3.28, the "F-TEST" toggle button was programmed to test the constant message load testing procedures. Upon pressing the button, the startTest() method was invoked. The load value entered in the left text box was read and used as an argument for the getNextMessage() method to generate the test message once. The generated

message load consisted of a series of alphabets from A to Z, with the series restarting from A when it reached the last alphabet, Z. Utilizing the postDelayed() method implementation, the program attempted to send the fixed message every 5 seconds. The subsequent message was sent after the acknowledgment message of the previous test message was returned. This process continued until the "F-TEST" button was toggled off. The "C-TEST" button was programmed to test the constant increment message load testing procedures. Like the "F-TEST", the "C-TEST" button triggered message posting every 5 seconds. During each posting iteration, the getNextMessage() method increased the message load by ten, progressing through a series of alphabets from A to Z.

Table 3.11 details the test case design for constant message load testing. A total of four test sets were conducted. In set 1, the message load remained fixed at 60, and the message was sent 105 times from device A2020 to device A5000, followed by 105 times from device A5000 to device A2020. For sets 2 through 4, the same steps were repeated with constant message loads of 120, 180, and 240, respectively.

Table 3.11 The test case design of the fixed size message load testing

| | Set 1 | | Set 2 | | Set 3 | | Set 4 | |
|---|---|---|---|---|---|---|---|---|
| | x = message load size; i= number of iteration for each test | | | | | | | |
| Devices | x=60; 1<=i<=105 | | x=120; 1<=i<=105 | | x=180; 1<=i<=105 | | x=240; 1<=i<=105 | |
| A2020 | sender | receiver | sender | receiver | sender | receiver | sender | receiver |
| A5000 | receiver | sender | receiver | sender | receiver | sender | receiver | sender |

Table 3.12 The test case design of the constant increment message load testing

| | | **Set 1 to Set 10** |
|---|---|---|
| | | for each set, message load size (x) increased by 10; (x=x+10 & x<=240) |
| Devices | A2020 | Send |
| | A5000 | Receive |

Table 3.12 lists the test case design for constant-increment message load testing. The test was conducted ten times, and for each set of tests, the message load started at 10 and increased by 10 until reaching 240.

For all the test data collected, outlier data was removed, considering possible interference from the surrounding wireless equipment or unexpected behaviour of the Wi-Fi interface during the test. The outliers were values that displayed extreme readings, such as excessively long delivery times or significantly larger variances in delivery time compared to the average.

**ii. Test case 2- Data forwarding**

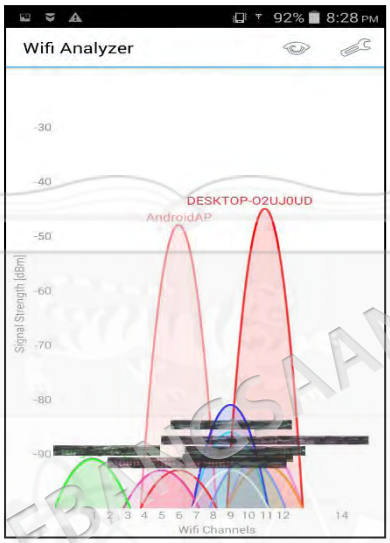



Figure 3.29 The Wifi Analyzer used to analyze the Wi-Fi signal

For this test case, the main goal was to measure the performance of the data forwarding procedure between three devices: the Lenovo A2020, the Lenovo A5000, and the ASUS X013D. The first and last devices were separated at a distance where both devices would not be able to detect each other's Wi-Fi signal, so the data could be forwarded by the device in the middle. Since the Wi-Fi transmission distance is device-dependent and influenced by the environment, there is no single recommended or fixed distance that could guarantee the devices would not detect each other's signals. Therefore, two preliminary experiments were conducted to identify the suitable positions of the devices for the second test case. The application named "Wi-Fi Analyzer," as shown in Figure 3.29 was installed on the smartphone as a tool to measure the Wi-Fi signal strength between devices.

The first experiment was conducted in an open area to test how far the wireless signal could be transmitted. It was found that the Wi-Fi signal transmission distance

exceeded 30 meters in an open area. The second experiment was conducted indoors, within a building, to test how far the wireless signal could be transmitted. It was found that the propagation distance of the Wi-Fi signal was significantly reduced to below 15 meters within a building. This limitation in transmission distance was believed to be due to obstacles such as walls, furniture, and electrical devices that affect the propagation of the Wi-Fi signal.

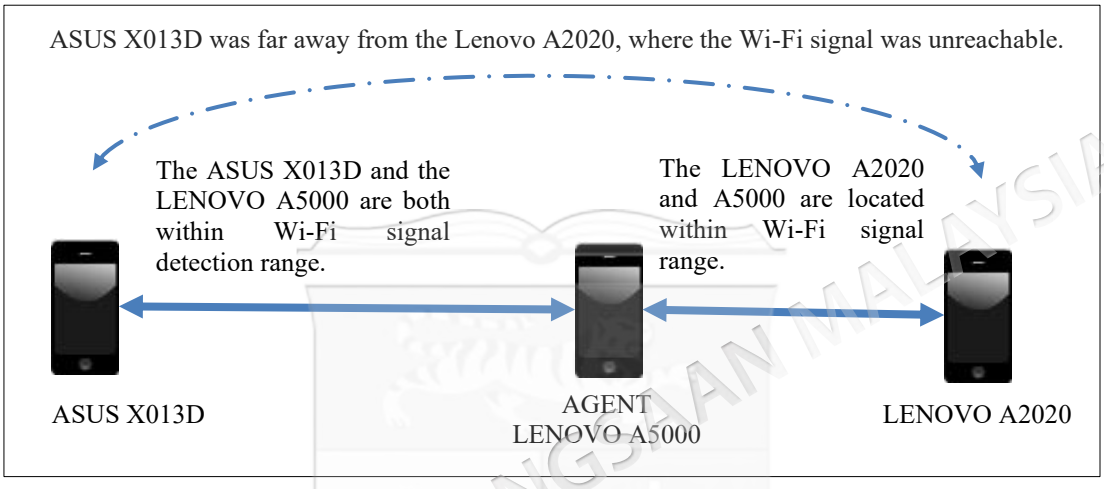



Figure 3.30 The arrangement of smartphones for the data forwarding test case

It was decided that the second set of test cases would be conducted indoors. This is because the chances of end users being in an open space all the time are lower than indoors. Furthermore, the main purpose of the experiment was to provide a proof of concept to demonstrate the operational feasibility of the developed model. Therefore, a comparison between indoor and outdoor settings was not included as one of the objectives, and it is sufficient to conduct the experiment in an environment closer to reality. However, conducting the experiment in an outdoor setting can be considered in the future for comparative analysis. Figure 3.30 illustrates the position of each smartphone for the second test case. The device ASUS X013D functioned as the sender device and broadcasted the messages to nearby peers. The device LENOVO A5000, located within the communication range of the ASUS X013D, received the messages and acted as the agent executing the forwarding operation. The device LENOVO A2020, which was out of communication range with LENOVO A5000, could only receive messages from LENOVO A5000.

Table 3.13 The test case design of the fixed size message load testing with three devices

| | Set 1 | Set 2 | Set 3 | Set 4 |
|---|---|---|---|---|
| | x = message load size; i=iteration for each test | | | |
| Devices | x=60; 1<=i<=120 | x=120; 1<=i<=120 | x=180; 1<=i<=120 | x=240; 1<=i<=120 |
| X013D | sender | sender | sender | sender |
| A5000 | forwarder | forwarder | forwarder | forwarder |
| A2020 | receiver | receiver | receiver | receiver |

Table 3.13 lists the test case design for message forwarding testing. A total of four test sets were conducted. For Set 1, the message load was fixed at 60, and the message was sent 120 times from ASUS X013D. For Sets 2 to 4, the same steps were repeated with message loads of 120, 180, and 240, respectively.

## 3.6 SUMMARY

In this chapter, the step-by-step methods for achieving the research objective were explained. For the first objective, a preliminary analysis was conducted. The preliminary analysis was divided into two stages. The first stage tested the wireless interfaces available on a smartphone and the relevant applications in the marketplace. This was done to understand the current state of the art in the research domain. The second stage of the preliminary analysis elaborated on the systematic scoping review procedures conducted to identify the relevant literature, research methodology, and measurement parameters in the research domain.

For the second objective, the model development procedures were carried out. The adopted software development model was explained. The conceptual architecture view, class diagram, and user interface of the final prototype were presented. For the third objective, the deployment, measurement, and evaluation procedures were executed. The elaboration covered the testbed, experiment setup, measured parameters, and test cases for the testing procedures.

In the next chapter, the detailed results and findings generated from all the research procedures will be elaborated upon.

# CHAPTER IV

# RESULT AND DISCUSSION

## 4.1 INTRODUCTION

This chapter presents the outcomes and results generated from the research methodology planned. The chapter is divided into three main parts. The first part presents the second outcome of the preliminary analysis, completing the first objective. The second part presents the outcome of the second objective, which is the prototype with the integrated data exchange model. The third part presents the results from the conducted experiments, completing the third objective.

## 4.2 RESULT FROM PRELIMINARY ANALYSIS - SCOPING REVIEW

The preliminary analysis was conducted to achieve the first objective. The observations and findings of the first part of the preliminary analysis, i.e., the preliminary testing, are reported in section 3.3.1. The second part of the preliminary analysis is the scoping review, conducted to comprehend the extent to which the WFD and SD protocols have been exploited in the formation of WCN. The a priori protocol of the scoping review is reported in 3.3.2a. This section presents the results and findings of the scoping review.

At the beginning of the search, the literature search engine yielded numerous irrelevant results. It was also noted that researchers used different spellings for the term "Wi-Fi." After multiple attempts and filters, the search strings "wifi direct," "wi fi direct," or "wi-fi direct," and "service discovery" were employed to gather only articles describing Wi-Fi Direct. Table 4.1 lists the search strings used for three databases: IEEE Xplore, ACM Digital Library, and Scopus. The search results were exported as BibTeX and imported into Mendeley for filtering and screening. As

depicted in Figure 4.1, the search string returned a total of 186 records. Mendeley was utilized to filter out the 21 duplicate articles. The screening process then involved reading the titles and abstracts. Only 39 articles were deemed relevant to the research based on the abstract explanations. Among these, 38 full texts were downloaded, and 1 article was inaccessible. The 38 full-text articles were skimmed, and three were excluded—one due to its irrelevance to the research topic, and two for having duplicate content. Ultimately, 35 articles were reviewed, and the findings were tabulated, as shown in Table 4.2. The review process focused on the proposed solutions, measurement parameters, research work limitations and suggestions. The details and relevance of the research findings are elaborated in section 2.5 of CHAPTER II.

Table 4.1 Search string used for individual database

| Database | Search string |
|---|---|
| ieeexplore | (“Full Text & Metadata”:”wifi direct” or “wi fi direct” or “wi-fi direct”) AND (“Full Text & Metadata”:”service discovery”) 2009-2023 |
| ACM digital library | “query”: {Title:(“wi-fi direct” OR “wifi direct” OR “wi fi direct”) AND “service discovery”} OR Abstract:(“wi-fi direct” OR “wifi direct” OR “wi fi direct”) AND “service discovery”} OR Keyword:(“wi-fi direct” OR “wifi direct” OR “wi fi direct”) AND “service discovery”} OR Fulltext:(“wi-fi direct” OR “wifi direct” OR “wi fi direct”) AND “service discovery”}} “filter”: {Publication Date: (01/01/2009 TO 12/31/2022)} |
| Scopus | ALL ( ( “wifi direct” OR “wi-fi direct” OR “wi fi direct” ) AND “service discovery” ) AND PUBYEAR > 2008 AND PUBYEAR > 2008 |

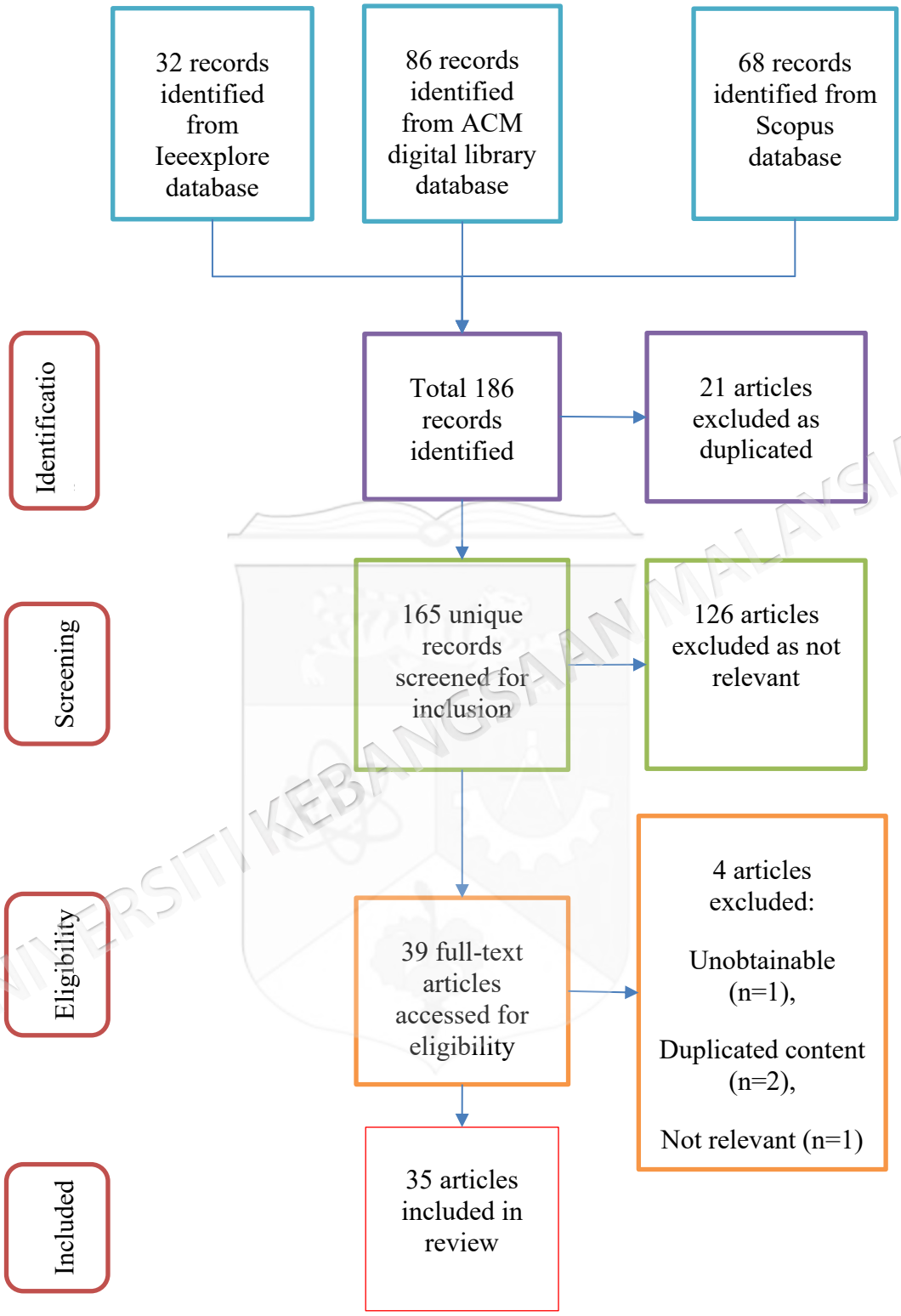



Figure 4.1 PRISMA-SCR flowchart of study selection process for scoping review

Table 4.2 Tabulated information from the literature collected from scoping review.

| Title (Year) | WFD \|SD | Paper Aim | Parameter Measures | Contributions | Limitations and Suggestions* |
|---|---|---|---|---|---|
| A Collaborative Video Download Application Based on Wi-Fi Direct (2016) | Y\|N | A protocol to offload the video download task to peers and combine the downloaded chunks with the group owner | Video Data Rate (KBps), Downlink Rate (KBps), WFD Transferring Rate (KBps), Connection Period (s), Disconnection Period (s), Number of Group Members, Buffered Playback time (s), Playback duration of a Video Chunk (s), Rounds for the GO Receiving Video Chunks from the Group Members n, Total Number of Chunks, Time for Playing Video in, The Extra Buffered Video Content A(KB) | Broadcast video chunk download service request through SD in WFD. Peers around performed the download task and transferred it to the GO through the WFD protocol. | Limited testing data. *Could be improved by testing in real environment involving more devices |
| A Framework for Enabling Internet Access Using Wi-Fi Peer-To-Peer Links & A Framework for Hotspot Support Using WFD Based D2D Links (2019) | Y\|Y | Introduced a protocol that allows users to send data to the Internet without requiring Internet access to be enabled on the smartphone. | GPS coordinates, actual time for peer to discover the service (n), waiting time for peer to rebroadcast or forward message (m), propagation time Tpr, transmission time Ttr, computation time Tc, n=Tpr+Ttr+Tc<br>Tpr = Distance/wave speed (3.10x10^8 speed of light)<br>Ttr= packet size/bit rate<br>overhead = generated messages/successful messages<br>Simulation with WiDiSi | SD was used to disseminate the message to the server through gateway nodes with an Internet connection. | One-way communication to the server and there is no mechanism to guarantee the successful transmission of data. *Could be enhanced by implementing two-way communication and adding a mechanism to ensure guaranteed message delivery |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| A Framework for P2P Networking for Smart Devices Using WFD (2014) | Y\|Y | Introduced a framework to support the formation and management of groups of communication devices | N= max length(bits) of heartbeat message<br>M=max number of group members including GO<br>N*M=max length of peers' list message<br>total number of messages around GO (in and out)<br>=(M-1)/alpha + 1/beta messages per second<br>total number of messages around a client<br>=1/alpha + 1/beta messages per second<br>Bandwitdh by GO<br>= ((M-1)/alpha x N) + (1/beta x N x M) = ((M-1)/alpha + M/beta) x N bps<br>bandwidth by client<br>=(1/alpha x N) + (1/beta x N x M) = (1/alpha + M/beta) x N bps | Used the WFD protocol to establish communication between peers via GO.<br>Utilized SD to broadcast uniqueID, device name, and user name to propagate the service.<br>Assuming N=500, M=20, alpha=1, beta=5, bandwidth by GO = 11.5Kbps, and bandwidth by the client = 2.5Kbps—significantly lower than the standard WFD speed of 54Mbps—so overhead can be ignored.<br>Adding a new peer takes 1 and 6 seconds for the GO and clients to process, respectively.<br>Removing a peer takes 30 and 55 seconds for the GO and clients to handle, respectively. | Communication within the group alone.<br>*Could be improved by introducing intra-group and inter-group data exchange. |
| A Group-Less and Energy Efficient Comm Scheme Based on WFD Tech for Emergency Scenes (2019) | Y\|N | Proposed is a communication scheme to enable message delivery during emergencies. | Simulation with self-developed Python simulator | Modified the Android source code's device discovery procedure. Utilized only the find phase and omitted the scan phase. The search and listen operations in the find phase were employed for data exchange. Modified probe request/response packets to include a device name field (32 bytes) for carrying the information. The grid-quorum concept was implemented in scheduling the device to ensure that the search and listen states of the devices avoided overlapping. | Although message delivery latency was reduced, developers must recompile the Android source for implementation.<br>*Could exploit the WFD API to avoid the rooting and recompiling process. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| A Local Communication System Over WFD: Implementation and Performance Evaluation (2020) | Y\|N | Introduced intergroup and intragroup communication methods with WFD at the application layer. | Intragroup<br>-average throughput (Mbps), Packet loss rate(%), over lod (Mbps 0-100)<br>average throughput(Mbps), Packet loss rate(%), delay(ms) over distance (0-42m)<br>Intergroup<br>-average throughput(Mbps), Packet loss rate(%), delay(ms) over 0-9 interference group (2 devices) | Intragroup Communication:<br>GO periodically broadcasts GM list information (IP) to all GMs in the group, ensuring that all members receive a copy of the other members in the group.<br>GO-GM communication outperforms GM-GM communication.<br>A fuzzy logic-based, formalized quantitative decision algorithm was applied to the self-adaptive GO and GM mobile-controlled handover (MCHO) mechanism for the handover procedure.<br>GM and GO handover mechanisms with fuzzy logic have reduced handover decision delays. | Might introduce additional overhead with the implementation of the fuzzy logic algorithm.<br>Handover execution requires disconnection, discovery, and formation (average 7.8 seconds).<br>*Could be improved by applying other algorithms such as neural networks, decision trees, Bayesian methods, Genetic algorithms, etc. |
| A Measurement Study on D2d Comm Tech For IIOT (2021) | N\|N | measured the performance and energy consumption of WFH,WFD and BT | -power consumption<br>-transmission performance<br>-TCP/UDP performance Iperf was used to measure TCP/UDP performance. | BT in the working state consumed almost the equal power compared to WFH and WFD in the idle states. The power consumption of WFH was the same as WFD. File sending showed a higher current than file receiving for BT, WFH, and WFD.<br>At 5 GHz, consumption of power is higher than at 2.4 GHz. WFD in the scanning state showed the lowest current consumption. The average transmission rate of BT between two devices was 1.6 Mbps, compared to 50 Mbps for WFH and WFD. | *Could include a comparison with other protocols, such as Zigbee and NFC. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Ad-Hoc Collaboration Space for Distributed Cross Device Mobile Application Development (2020) | Y\|Y | Introduced an Ad hoc collaboration space to be used by developers who are developing apps that require the app to interact with other devices in Ad hoc mode and ensure synchronization of data across devices. | -SD and Group formation time<br>-Synchronization time across devices connected<br>-Impact of distance on data synchronization<br>-state synchronization time on device joined later<br>-state update synchronization time on device rejoined<br>-memory overhead, memory used during collaboration activities | The framework simplified the existing WiFi Direct protocol by introducing abstraction and three classes to manage services, events, and socket connections. A drawing app was developed and used as the testing application to measure performance. The app demonstrated synchronization on UI updates when users on different devices performed various operations (e.g., drawing). | Framework limited to the local area with a restricted number of devices only.<br>Details about how the recorded time was not explained.<br>*The framework testing could be expanded with more devices for a thorough evaluation. |
| Alert Dissemination Protocol Using SD in WFD (2015) | Y\|Y | proposed a protocol that utilizes the SD mechanism in WFD to disseminate alert | -assumed transmission speed 54mbps(standard54-600Mbps), distance 120m(indoor 70m,outdoor 250m)<br>-assumed discovery request and response frames have same size<br>-T(time needed to deliver a service discovery frame), Tp(propagation delay) & Tt(transmission time) | Introduced a local alert management and remote alert management scheme. Embedded the alert data in SD records.<br>-Tp = 120/(speed of light) = 120/(3x10^8)=0.4 microS<br>-Tt(depends on transmission rate&length of frame)<br>Tt= 5000x8 (bit)/54000000 = 0.7ms<br>T = 0.4microS + 0.7ms ~ 0.7ms | Assuming SD request and response frames have the same length of L bytes, which might not be true.<br>*Could test the protocol in simulation or the real world with more devices. |
| Benchmarking Wireless Protocols for Feasibility in Supporting Crowdsourced Mobile Computing (2016) | Y\|N | To study D2D multimedia content dissemination with Wi-Fi Client server, Wi-Fi Mobile server, Wi-Fi direct and Wi-Fi TDLS. | -file download time over number of devices<br>-traffic (MB) handled by the AP over number of devices for each protocol<br>-download time clients over number of servers for each protocol<br>-traffic (MB) handled by the server for each protocol | H1-file retrieval can be accelerated through D2D protocols<br>H2-D2D can help in reducing congestion or load on Access Point / server side<br>h1 not true,<br>h2 true, decentralized D2D technique reduce 65% traffic at the access points | Most of the testing results did not cover the WFD protocol.<br>*Could include WFD protocols in testing for better comparisons |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| BWMesh: Multi-Hop Connectivity Framework on Android For Proximity Service (2015) | Y\|N | Introduced a heterogeneous network scheme (BT+WFD) to facilitate multi-hop networking. | ability to pass message to peers | WFD as the additional communication interface other than BT. Device A connected to device B via BT while concurrently connecting to device C via WFD.<br>A prototype was developed to enable chatting among users in proximity in a multi-hop way without an internet connection. | Switching between BT and WFD might impose overhead.<br>The proposed schema requires user interaction during device pairing.<br>*Could be improved by automating the connection procedures |
| Content Sharing Using P2PSIP Protocol in WFD (2012) | Y\|N | Applied the P2PSIP protocol on top of the WFD network for multimedia sharing. | number of participants over transmission duration in ms | Typical WFD formation operations: create a group, and client devices join the group. Then, the application leverages the P2PSIP protocol for sharing operations | Required the user's manual operation.<br>*Could be improved by automating the connection procedures |
| Context-Aware Configuration and Management of WFD Groups for Real Opp Networks (2017) | Y\|Y | Introduced a WFD-GM protocol to enhance network connectivity and the message dissemination process. | -frequency of the messages exchanged.<br>-the percentage of success message dissemination,<br>-battery consumption.<br>Simulation with one simulator | The protocol leveraged SD to exchange context information (computational resources and battery status) to elect the GO. GO exchanged its credentials with nodes to skip the manual WPS provisioning process. Devices periodically exchange context to update group configuration. The WFD-GM protocol improves network connectivity and message dissemination compared to the baseline protocol in low- and medium-mobility environments. Performance is the same as baseline in a high mobility scenario. | Lack of experiment on real devices.<br>*Could be improved by conducting a real-world experiment and comparing the results between simulation and reality. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Development of MANET Over WFD With Off-The-Shelf Android Phones (2016) | Y\|N | Introduced an algorithm for multihop communication by leveraging WFD's GO and client connection procedures. | no measurement involved, only demonstrated interface of routing table formation and message delivery process | focused on the GO and client connection models.<br>All devices were programmed to be GO when there was no transmission, so that all devices were discoverable and ready to be connected.<br>When devices were discovered, routing tables were generated based on the Mac addresses exchanged. | Routing table formation time and message delivery time were not measured.<br>*Include routing table formation and message delivery time to ensure thorough analysis. |
| Development of Offline Chat Application Framework for Resilient Disaster Management (2018) | Y\|Y | Developed a framework for communication during disaster. | likert's scale in terms of functionality, reliability, usability, efficiency, maintainability, and portability | Wireless SD of WFD was used for peer to discover the service available.<br>Then connect to the network in Wi-Fi infrastructure mode with router | Framework requires additional hardware to work.<br>*Could implement virtualization to diminish the dependency on hardware. |
| Efficient Multigroup Formation and Communication Protocol for Wifi Direct (2015) | Y\|Y | Introduced a protocol for multi-group communication | no measurement involved, only demonstrated interface of a chat app | Leverage SD protocol for peers to exchange rank (calculated based on battery info, device with higher rank to become GO) and Soft AP credential. GO then used the information to choose and appoint a GM to be a proxy member to forward messages between groups. | Altering the system's source code is not practical.<br>No mechanism in place to keep the network running when a Group Owner (GO) or client with a specific role leaves.<br>*Could enhance the protocol with GO-client management. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Device to Device Communications With WFD: Overview and Experiment (2013) | Y\|Y | Experimental evaluation of the WFD's performance in real scenarios, in terms of the delays to be expected in practice when WFD devices discover each other and establish a connection. Evaluated the performance of the WFD power-saving protocol known as the Notice of Absence. | -group formation delay<br>-performance-energy<br>-trade-offs of NOA energy saving protocol<br>-Simulation with event-driven simulator (ns3 from reference) | Standard and persistent group formation used the same baseline for discovery, but persistent used an invitation mechanism and standard used GO negotiation; persistent used past recorded data for WPS provisioning. In autonomous mode, the device announces itself as "GO" and sets up a group for clients. Initial scan delayed all procedures by at least 3 seconds. The CDF graph showed randomness in the discovery delay (1-7s) for three different group formations. The discovery time in autonomous mode was constant (3s). Autonomous and standard showed similar delays because GO negotiation took less time than WPS provisioning, so the overall time taken for formation was equal.80% of the time, autonomous group formation took less than 5s; persistent and standard took about 8–9s. | The experiment was conducted only on laptops. Connections were established in a controlled environment.<br>*Could be improved by incorporating mobile phones and conducting the same experiment, followed by an analysis between phones and laptops. |
| Efficient Data Dissemination for Wifi P2p Networks by Unicasting Among Wifi P2p Groups (2018) | Y\|Y | Reduce the load of (GO) in the Working Flow Diagram (WFD) group by dividing the main WFD group into several smaller groups. Then, use store-carry-forward mechanisms to disseminate data. | -network formation time<br>-data dissemination time<br>-network repair time<br>-SD discovery time<br>-Simulation with even-based WiFi p2p network simulator | SD protocol to exchange information for group formation and repair procedures. Switching between GO and client to disseminate data with store-and-forward algorithm. In simulation, average network formation time increased when the number of devices increased and reduced when the maximum number of devices in a group at higher value. SD Discovery Time Experiment >8 devices provide 1-4 kinds of services; a device is asked to find a service from a service provider. Run for 20 times.<br>SD times varied; longest = 15.2s, average = 4s | In SD implementation, peer might not be able to identify the correct service in time, or conflicts between services might occur.<br>*Could be improved by incorporating procedures to capture and verify services, enhancing the protocol's stability. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Efficient Multigroup Formation and Communication Protocol for Wifi Direct (2015) | Y\|Y | Introduced a protocol for multi-group communication. | no measurement involved, only demonstrated interface of a chat app | Leverage SD protocol for peers to exchange rank (calculated based on battery info, device with higher rank to become GO) and Soft AP credential. GO then used the information to choose and appoint a GM to be a proxy member to forward messages between groups. | The source code was altered to enable devices to be on different subnets and to allow static assignment for validation-related devices (though this isn't practical in reality).<br>*Could be improved by adding GO handling mechanism to ensure connection stability. |
| Enhancing The QOS of Mobile-Based SW Over WFD (2021) | Y\|N | Introduced a model for choosing the primary GO (Group Owner) and backup device in a P2P (Peer-to-Peer) network. | group formation (conventional method) and reformation time with 2,3,4 devices | The ERRM model was used to identify a backup node to replace GO during group formation.<br>Service information was injected into a probe request to reduce the time and procedures of the service discovery phase.<br>group reformation time with ERRM reduced by 80% compared to standard group formation time | The state is dynamic, so the backup that would replace GO might not be suitable by the time it needs to replace GO, or when it needs to replace GO, there might be another device with a more potential value for the parameters.<br>*Could be improved by implementing an automating scheme for GO selection evaluation. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Hychat: A Hybrid Interactive Chat System for Mobile Social Networking In Proximity (2015) | Y\|N | Introduced a hybrid interactive chat system that enables p2p communication. Switching between offline communication with WFD and online over the Internet. | no measurement involved, only demonstrated interface of a chat app | Profile matching occurs prior to building connections among devices.<br>Profile (user_name and interest) information was encoded into the device name field of the WFD. During the group forming stage, profiles were exchanged and a decision to form a connection was made based on the interest fields in the profiles.<br>When the WiFi Direct was disconnected, the application switched to online mode. | The periodic detection time might affect the efficiency of mode switching.<br>*Could be improved by utilizing the device discovery protocol of WFD to disseminate parameters for mode switching decision. |
| Inter-Cars Safely Communication System Based on Android Smartphone (2014) | Y\|N | Introduced a system to alert nearby vehicles via SMS or WFD when an emergency occurs nearby. | no measurement involved, only demonstrated interface of a chat app | standard WFD connection formation and exchanging alert and coordinate information through the connection formed. | Required manual user intervention.<br>*Could be improved by adopting heterogenous connection to provide alternative connection interfaces. |
| Multi-Group Message Communication on Android Smartphones Via WFD (2017) | Y\|N | Introduced a model that enables multi-group message communication by exploiting the group formation capabilities of WFD to exchange information between multiple groups. | delay over size of file transferred (image 1-40mb), audio & video (1-100mb) | Each device generates an identifier and holds a server socket that constantly listened for any mode to be activated. Modes: register, update, transfer, and scatter. Register, the peer must register itself to GO. Devices moved into a listening server socket after registration. In update mode, GO forwarded a hash table with to all peers in the group. In transfer mode, GO operated as the middleware to forward data. In scatter mode, peers disconnected from GO and connected to another GO, then went back to the original GO | Device identifier duplication issue.<br>*Could be improved by combining various parameters to generate unique identifiers. This might involve incorporating details such as MAC addresses, timestamps, etc. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Network-Assisted D2D Comm: Implementing A Tech Prototype for Cellular Traffic Offloading (2014) | Y\|N | Introduced a model to offload cellular traffic to D2D connection. | -sustainable rate<br>-stream latency<br>-signalling latency<br>-activation delay<br>Simulation with system-level simulations with a self-designed simulation tool | Cellular networks were used to exchange mobile information (to the server) to assist the D2D device discovery and connection establishment stages. Therefore, the GO negotiation was skipped, and the GO was elected by the server.<br>When the peers were at proximity, data exchanged over WFD | The model requires an active connection to the cellular network. Root access is required on Android devices.<br>*Could exploit the WFD API to avoid the rooting process. |
| Nextcontact: Neighbour Discovery Mechanism for OppNet And Node Movement Based Neighbour Discovery in Opp Net (2017) | Y\|N | Introduce a node discovery scheme called "NextContact" for neighbor discovery, based on an equation used to calculate the probing interval. | Tprob = 2R/(coEf*~V).<br>R=communication range of device (wifi range),<br>0<coEf<1,<br>~V = ~D/(T+Tpause)<br>~D=accumulative distance travelled<br>T=total time travel<br>Tpause=time paused during ~D<br>parameters for ONE simulator: simulation area, number of nodes transmission range, transmission rate, movement speed<br>, mobility model, message, generation rate, size of messages | The author introduced an equation to calculate the probe interval and tested the idea in ONE simulator<br>Result showed peer detection of more than 80% with a coEff value of 0.4 and above.<br>When compared with the PISTON v2 algorithm, NextContact shows better performance in terms of a higher percentage of peer discovery, lower energy consumption, and a higher message delivery ratio. | Device discovery cannot guarantee message delivery ratios, as there are other effects that need to be considered in the WFD protocol.<br>*Could consider including the device's time pause when calculating the probe interval to achieve a more thorough measurement. |
| Performance Evaluation of The Dynamic Multi-Hop in Proximity Radio Access Network (2022) | N\|N | Proposed a model for multi-hop communications networks based on the WFD protocol | -throughput rate (Mbps)<br>-energy consumption J<br>-Simulation with 5G toolbox and WLAN toolbox in MATLAB R2021b | Introduced the theoretical P-RAN (Proximity Radio Access Network) mechanism in a simulator to simulate multihop communication. Data was offloaded to the neighbouring devices and then relayed to the cellular network to reach its destination. | The experiment on the multihop model was conducted in the MATLAB simulation platform.<br>*Could develop a prototype for testing in a real environment. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Privateshare: Measuring D2D User Behaviour and Transmission Quality (2016) | Y\|N | Developed a prototype called PrivateShare for D2D content sharing. Real-time data and usage behaviour were recorded for the use of other researchers. | -duration of device being in range<br>-time available to communicate<br>-signal strength<br>-upstream and downstream bandwidth<br>-success of D2D content transmissions<br>-number of errors encountered while using the app | Applied QR or NFC to exchange credentials for a D2D connection.<br>Used standard WFD connection procedures to form D2D connections and data exchange.<br>After the WFD connection was formed, the device automatically synchronized content to be shared with peers. | The prototype utilized the existing WFD protocol and only worked for intragroup data exchange.<br>*Could be enhanced to introduce an intergroup data exchange mechanism. |
| Quality-Aware Traffic Offloading in Wireless Network (2017) | Y\|Y | Introduced a framework called QATO which offloads network tasks to peers with better service quality over cell stations. | -uplink and downlink throughput of different carrier<br>-power consumption when using carrier connection.<br>-Simulation with Trace-driven simulation | DNS-SD was used to exchange network information. The network task was sent to the offload engine module. The offload engine compared information from the local and neighbouring networks before making an offload decision. Evaluation showed energy savings and less delay in task completion. Evaluation of SD showed that devices took an average of 2 seconds to discover a peer. | *Could develop a prototype to test the theoretical model and implement it in a real environment setting. |
| Seamless Group Reformation in Wifi P2p Network Using Dormant Backend Links (2015) | Y\|N | Proposed a concept of seamless WFD group reformation to reduce group disruption time by implementing the Dormant Backend Link method. | group formation time | In the GO negotiation request/response frame, an extra field is added with the existing GO Intent value for EGO. When group reform is required, the peers in the group have access to the EGO list and automatically know which GO to connect to next.<br>Experiment conducted with 3, 7, and 10 devices show that more than 90% of group reformations using the SGR+DBL model are completed in less than 1 sec, while group reformations without the model take more than 1 sec. | The concept was tested on a laptop, and the findings could not accurately represent the real behaviour of all mobile devices.<br>*Could develop a prototype to test the theoretical model in a real-world environment. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Social-Aware D2D Offloading Based on Experimental Mobility and Content Similarity Models (2018) | Y\|N | Introduced a model for D2D content sharing that considers mobility parameters, content-related parameters, and social network relationships when identifying potential D2D peers for connection | -number of contacts between users<br>-contact duration between users<br>-inter-contact duration<br>-content-related parameters<br>-social network relationships,<br>-social relationship strength<br>-the number of mutual friends between participants<br>-correlation between mobility parameters,<br>-content similarity,<br>-social network links | A model was proposed to identify the optimal D2D connection for data sharing by considering the social network relationships between users, their content interests, and the frequency and duration of contacts between paired users over time and space. | Privacy could be a concern, for example, user authentication to a personal social network account is required for the system to collect information such as a user ID and friends list to evaluate the level of trust for seamless content sharing.<br>*Could also integrate an encryption algorithm to protect the exchanged data |
| Testing Nearby P2P Mobile Apps At Large (2019) | N\|N | Introduced a test framework for developers to conduct reproducible and automated tests on P2P apps, which are installed on physical or virtual devices connected to the testing framework environment. | -the number of bugs detected (permission bugs, scalability bugs, protocol bugs, pervasive bugs)<br>-number of users that received a message sent | Physical mobile devices were connected to the development environment, along with virtual emulators. Experimentation demonstrated that the framework was capable of detecting bugs in the apps installed on these devices. The framework could also assess whether an app was capable of detecting peers, determine the number of peers detected, establish connections, and send messages to the maximum number of users. Furthermore, the framework allowed testers to fine-tune the discovery delay and connection strategy for both WiFi (1:M) and Bluetooth (M:M) scenarios. | Limited measurements have been taken on wireless performance parameters.<br>*Could be improved by introducing more performance parameters for measurement, such as bandwidth, latency, success and failure rates, etc. |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Towards Cloudless Co-Located Social Media On Android (2016) | Y\|Y | WFD for intragroup, BT for intergroup connections | -delivery rate<br>-message load | WFD protocol was utilized for SD and GO negotiation. Connection was formed either with a WFD or BT connection.<br>Gradient-routing GR algorithm was used for unicast message dissemination. The flooding algorithm and GBS algorithm were compared for performance evaluation.<br>In 1:M broadcast scenario. Delivery ratios with flooding and CBS were about 90% in 300, 600, and 1200 messages.<br>CBS generates 40% of messages generated by flooding. In the M:1 scenario. Delivery ratios with GR and CBS are about 97%, in 300, 600, and 1200 messages, respectively. GR generates 40-50% of messages generated by CBS. In both scenario flooding algorithms created broadcast storms. | BT limits intergroup connection distance.<br>Interconnectivity between peers not maintained.<br>Disconnections are not automatically detected or repaired in BT.<br>*Could be improved by selecting connections with BT or WFD based on the QoS parameters from peers, i.e. distance or battery. |
| Usable - A Communication Framework for Ubiquitous Systems (2014) | N\|N | Introduced a multihop framework with carry-and-forward routing algorithms for developers. | -RTT of the message using AODV<br>-The number of messages lost<br>-The number of messages received<br>-The ratio between the total number of messages sent and received | The connection-aware layer was responsible for neighbour discovery, connection, message exchange, and neighbour disconnection.<br>The network layer had the responsibility of routing messages to destinations via BT or WiFi, using the multi-hop algorithm such as flooding or AODV<br>The Application layer managed application instance operations, message sending and receiving procedures. message in JSON structure. It was found that RTT for AODV was around 750 ms for 3 hops. For each additional hop, the time increased by around 250 ms. The standard deviation was around 200ms. | The experiment did not consider group formation time and the mobility of devices.<br>*Could enhance the accuracy of performance evaluation by including group formation time |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| WD2: An Improved WFD Group Formation Protocol (2014) | Y\|N | Introduced a group formation protocol based on the WFD protocol designed for larger groups of devices. | -Throughput<br>-Group formation time | The RSSI values collected from peers during the device discovery were used for calculating the intent value that was then encoded in the probe request frame before forward. The device with the highest IV become the GO. The average throughput of the group, determined by the GO elected by WD2, exceeded that of Android's default random GO selection mechanism. The group formation time on Android was longer than WD2. The simulator replicates the behaviour of WFD on Android using the WFD API, which includes wifiP2pmanager, NodeP2pinfo, and eventListener. | *Could be improved by integrating the protocol into a practical application and measuring its stability and performance in a real-life scenario. |
| WiDiSi: A WFD Simulator (2016) | Y\|N | A WFD simulator on top of PeerSim was built for the Android platform. | -package drop rate<br>-delay for package transfer | The simulator replicates the behaviour of WFD on Android using the WFD API of Android, which includes wifiP2pmanager, NodeP2pinfo and eventListener. | The simulator only supports Bonjour SD.<br>Channels and channelListeners are not available.<br>Listener callbacks for the 'success' method are not implemented. Implement listeners only to inform the app that the required information is ready to be picked up.<br>*Could be improved by including the implementation of more listeners for callback handling." |

to be continued…

… continuation

| | | | | | |
|---|---|---|---|---|---|
| Wi-Fi Direct Performance Evaluation for V2P Communications (2020) | Y\|N | Proposed a solution to increase connection time in WFD | -communication range,<br>-Packet Delivery Rate<br>-Packet Inter-Reception Time<br>-Simulation with OMNet, INET | The proposed method of stuffing the beacon involves overloading fields in the beacon to carry information. A string of bytes containing the message identifier, fragment number, flag (indicating the presence of more fragments), and message contents was embedded into the beacon. The information was embedded into the 32-byte device name, following the structure: device name + device ID, coordinates, speed, and travel direction. Testing the proposed method resulted in a packet delivery rate of 99% and an inter-reception time of 1 second for packets. | The total amount of information to be transmitted is limited to 32 bytes.<br>*Could be improve by exploiting other parameters of the beacon to increase the length of data data for dissemination. |
| Wi-Fi Direct Research - Current Status and Future Perspectives (2017) | Y\|N | A review article about Wi-Fi Direct | -delay of standard Group Formation scheme presented in normalized CDF<br>-Simulation with mac80211_hwsim | Focused on speeding up Group formation. The author proposed to encode the IV&Peer list in IE for Probe request and respond so devices can use it to decide GO without going through the GO negotiation stage.<br>Simulation finding<br>1. standard group formation time:<br>average device discovery time required 1070 ms<br>>negotiate GO took 850ms-9000ms, average 2198ms (50% time spent to form a group)<br>>group forming took 903ms (median 873ms)<br>2. implement proposed scheme Group formation time:<br>2devices- overall Group formation delay improved by 20%<br>5devices- Group formation average 8000ms (almost 3x faster) | The proposed solution was theory-based, with no real devices involved in testing.<br>Could develop a prototype based on the proposed model to test both the theoretical model and its real-world implementation. |

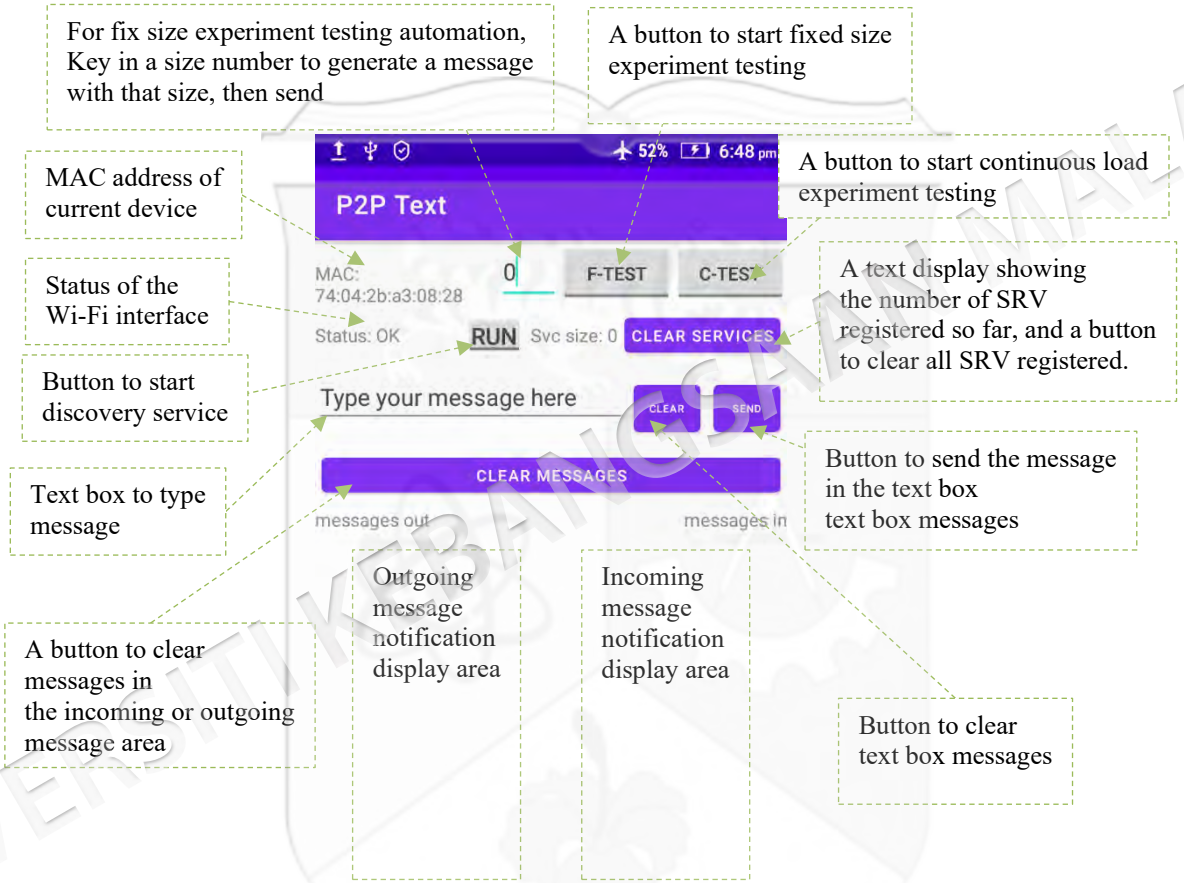



Figure 4.2 The screenshot of the prototype developed and deployed to Lenovo A5000

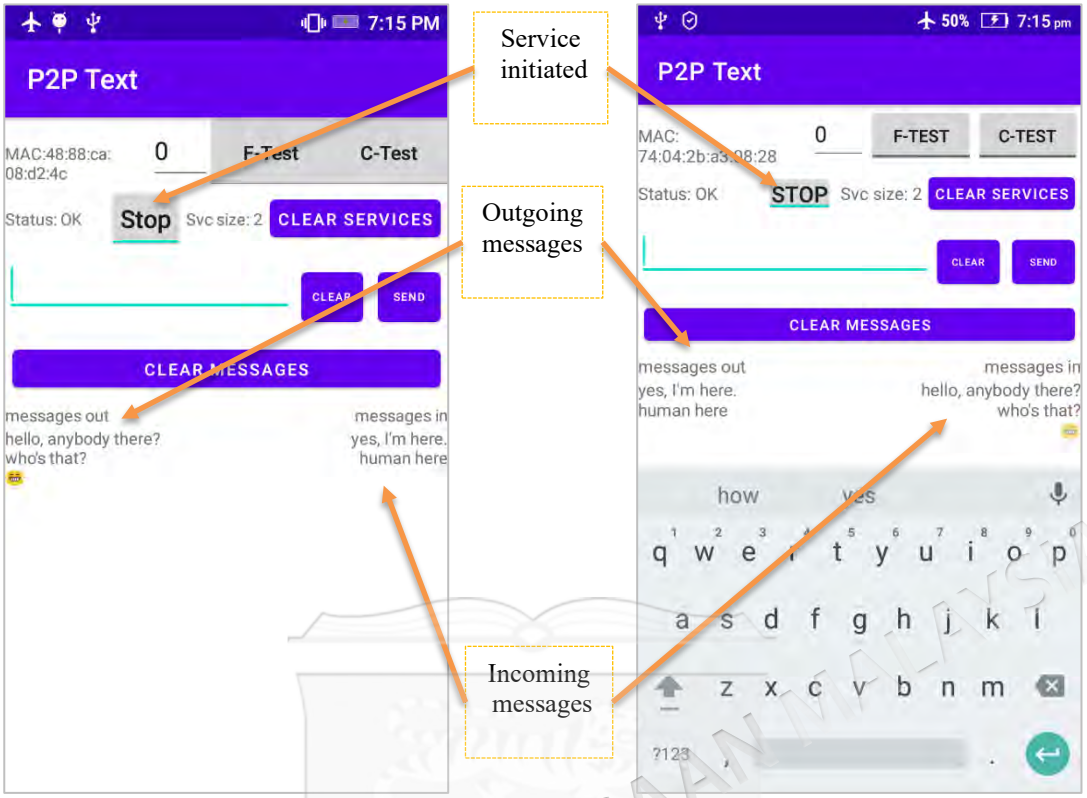
## 4.3 MODEL AND PROTOTYPE DEVELOPED

### 4.3.1 UI for final prototype

Figure 4.2 depicts a screenshot of the final prototype developed and installed on the Lenovo A5000. The MAC address of the device is shown below the label 'MAC.' The status label represents the Wi-Fi status of the device. When the Wi-Fi is turned on and the device is ready for the P2P service, the status will be labeled as 'OK'; otherwise, the status will be labeled as 'BUSY.' The RUN toggle button must be triggered to start the P2P service. The toggle buttons F-TEST and C-TEST are used for experimental testing operations. When the F-TEST button is pressed, the application will fire a fixed-size message based on the number entered in the text box next to the F-TEST button every 5 seconds. When the C-TEST button is pressed, the application will fire a constantly incrementing size message of 10 every 5 seconds. The label 'Svc size' represents the service instances generated by the device and placed in the message queue. The 'CLEAR SERVICES' button is used to clear the service instances in the queue. The text box to enter the message is placed below the RUN button. The 'SEND' and 'CLEAR' buttons are used to send and clear the messages in the text box. The incoming and outgoing messages are listed at the bottom of the UI, labeled as 'messages in' and 'messages out.' The 'CLEAR MESSAGES' command is used to clear the recorded incoming and outgoing messages.

### 4.3.2 Screenshot of messaging process between devices

Figure 4.3 illustrates the sample messaging process between Lenovo A2020 and Lenovo A5000. The text on the RUN button of both devices changed to STOP after the button was pressed. Lenovo A2020 initiated the chat process and sent the first message. Lenovo A2020 received the message and replied. The outgoing messages from Lenovo A5000 are listed under the "messages out" column, and the incoming messages from Lenovo A2020 are listed under the "messages in" column.




a) Device Lenovo A2020 initiated P2P Text service, sent 3 messages

b) Device Lenovo A5000 initiated P2P Text service, received and replied 3 messages

Figure 4.3 The screenshot of data exchange illustration between two devices.

### 4.3.3 Screenshot of messaging Wi-Fi interface or service not ready

The P2P service will not be available when the Wi-Fi interface is disabled or there are runtime errors in the system's Wi-Fi framework. While the system is recovering from the error, the application continues to update the status on the UI. Figure 4.4 shows a symbol with a circulating line, and the status is replaced by 'Busy'. The status will reset to OK once the system has recovered from the error or the Wi-Fi interface has been re-initiated.

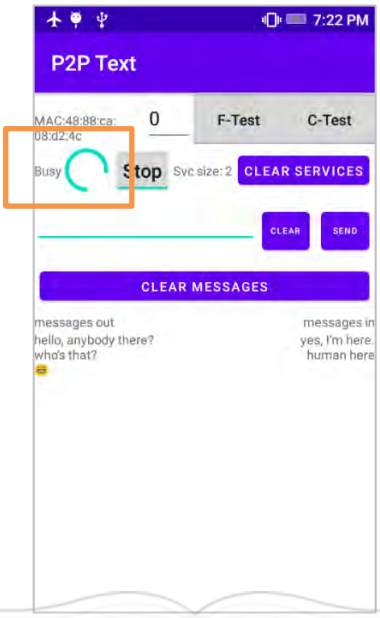
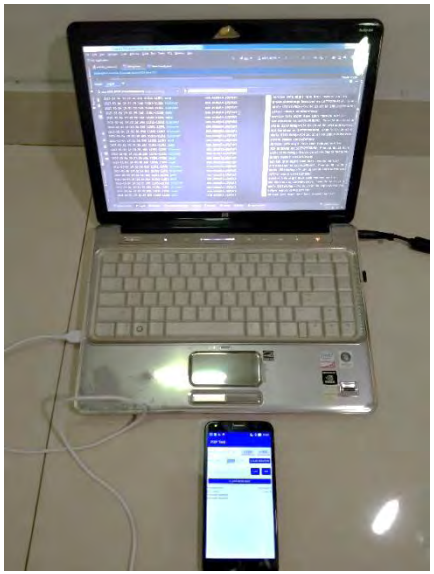



Figure 4.4 Screenshot of the status update when Wi-Fi interface was turned off or P2P service not available.

## 4.4 MEASUREMENT AND TESTING

Before testing was conducted, the final prototype was installed on the smartphones, and the time of the devices was synchronized through the system settings. The Android Studio Logcat tool was used to record the logs generated by the smartphones when the service was running. Each smartphone was connected to a laptop during the test. As explained in 3.5.2, three smartphones and three laptops were used for the experimental testing. Figure 4.5 illustrates the smartphones and laptops that were employed for experimental testing and data collection. The first devices to be paired were the Lenovo A2020 and the HP Pavilion dv4 laptop via a micro-USB cable. The second paired device was the Lenovo A5000, connected to the Lenovo ThinkBook 14 laptop using a micro-USB cable. The third paired device was an Asus X013D, connected to the HP Pavilion 14 laptop using a micro-USB cable.



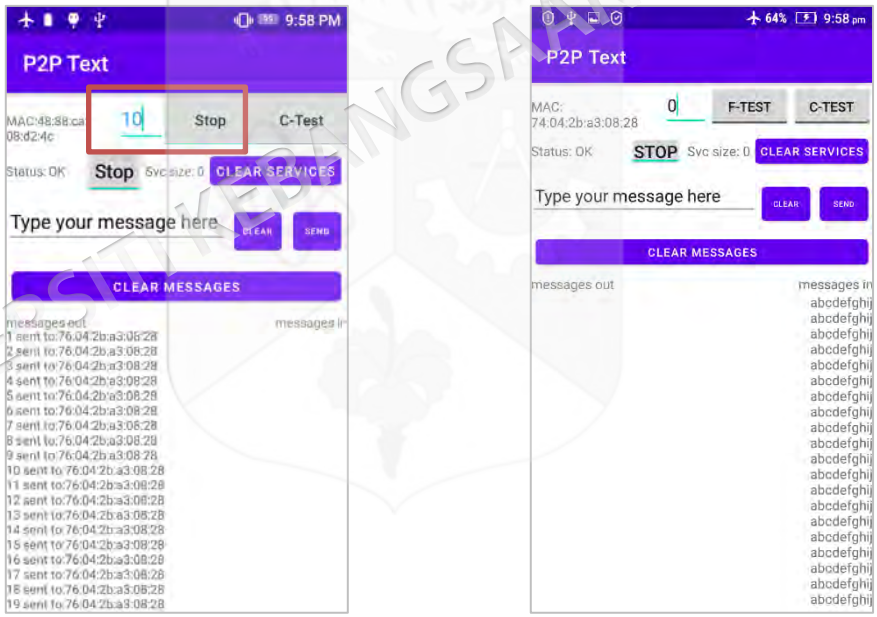
a) Lenovo A2020 connected to HP Pavilion dv4

b) Lenovo A5000 connected to Lenovo ThinkBook 14

c) Asus X013D connected to HP Pavilion 14

Figure 4.5 Testbed and experiment setup with mobile phones connected to individual laptop

Figure 4.6 demonstrates a screenshot of the UI for experimental testing with the 'constant message load testing' test case, as listed in section 3.5.2c.i. The screenshot on the left was captured after a few seconds when the F-TEST button on the Lenovo A2020 was pressed. The Lenovo A2020 was used as the sender device, with a constant size of 10 filled in the text box. After pressing the F-TEST button, the device started sending messages generated by the model to a nearby peer. The messaging operation was executed continuously until the F-TEST button was pressed again. The model attempted to send a subsequent message every 5 seconds, and the message was sent only when the ACK message of the previous message was received. The screenshot on the right was captured after a few seconds when the receiving device, a Lenovo A5000, received the constant size messages sent by the Lenovo A2020.




a) Device Lenovo A2020 initiated F-Test and sent messages of size 10

b) Device Lenovo A5000 received F-Test messages of size 10

Figure 4.6 Screenshot showing Lenovo A2020 sending constant message load to Lenovo A5000 with F-TEST

Figure 4.7 demonstrates a sample screenshot of the log generated by the Lenovo A2020. The log was recorded by Logcat, running on an HP Pavilion dv4 while the F-TEST was being executed. The logged data includes the log date, log time, process number, tag name, package, log type, action, smartphone timestamp

when the message was sent, the sent message, and the message length. The screenshot of the log file logged from a sender device for the data exchange test case before and after tabulation can be viewed in Appendix E.

Figure 4.8 demonstrates the screenshot of the log generated by the Lenovo A5000. The log was recorded by Logcat, which was running on the Lenovo ThinkBook 14 after the F-TEST was executed on the Lenovo A2020. The logged data includes the log date, log time, process number, tag name, package, log type, action, smartphone timestamp when the message was received, source device MAC, message UID, received message, and message length. The screenshot of the log file logged from a receiver device for the data exchange test case before and after tabulation can be viewed in Appendix F.

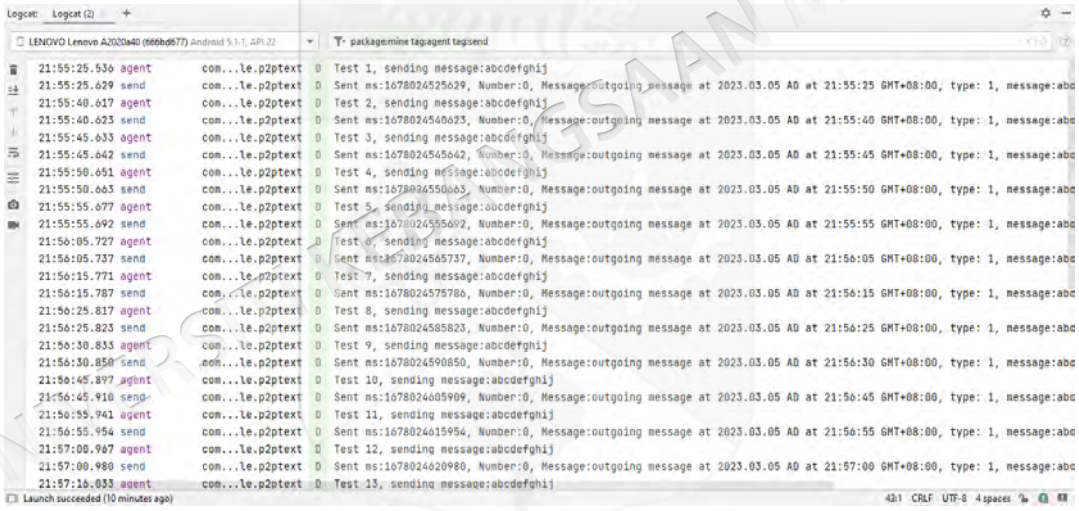



Figure 4.7 The screenshot of Logcat recording F-TEST sender data

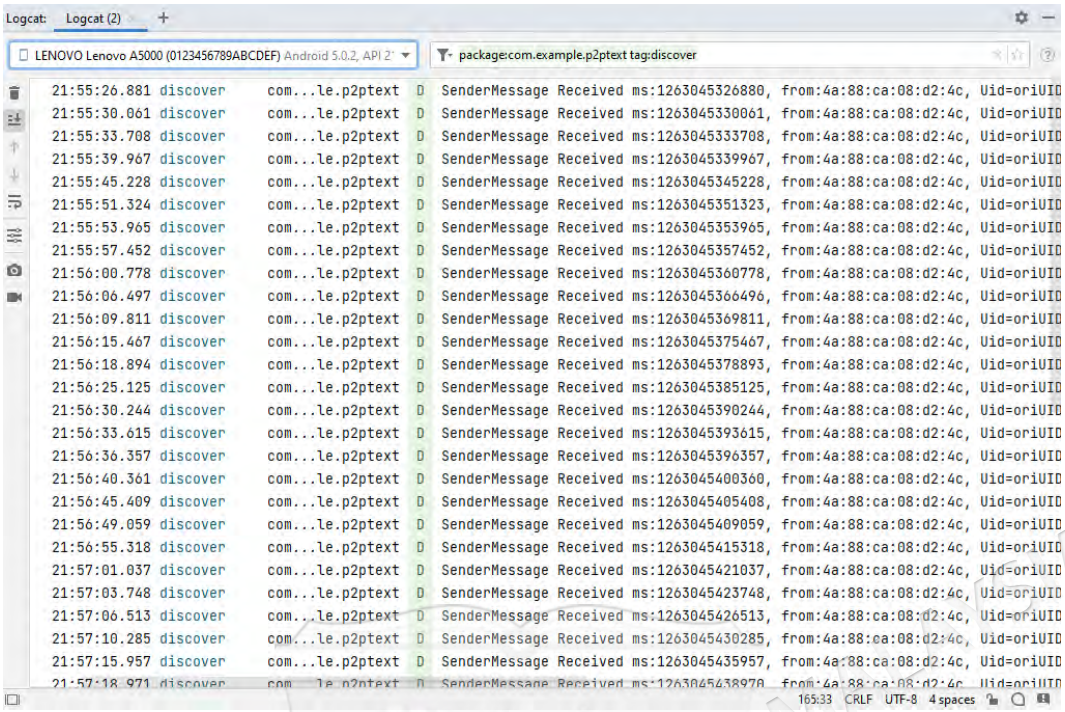
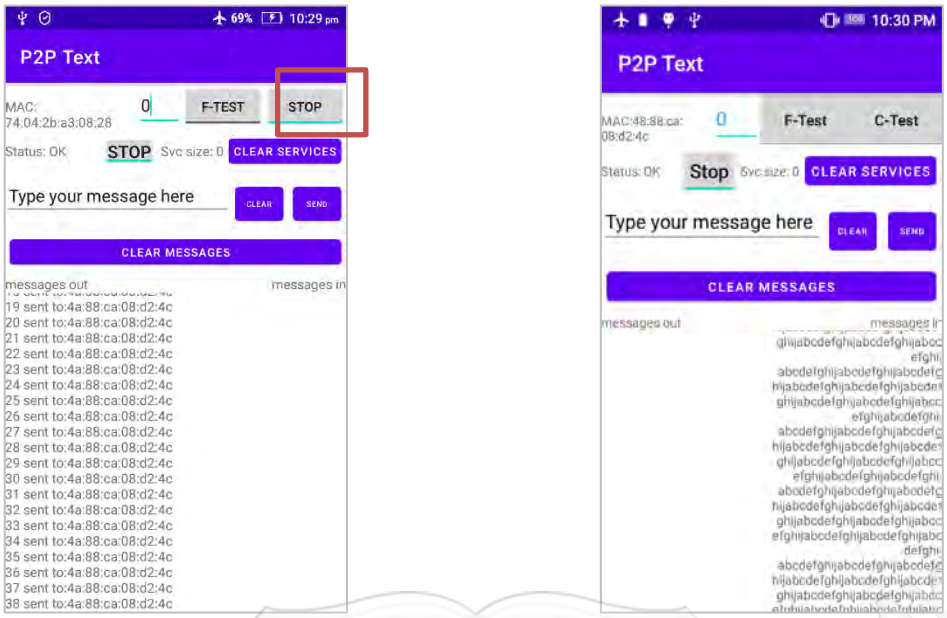



Figure 4.8 The screenshot of Logcat recording F-TEST receiver data

Figure 4.9 illustrates a screenshot of the UI for experimental testing with the 'Constant increment message load testing' test case, as listed in section 3.5.2c.i. The screenshot on the left was captured after a few seconds when the C-TEST button on the Lenovo A5000 was pressed. The device began sending a message generated by the model to a nearby peer after the C-TEST button was pressed. The messaging operation continued uninterrupted until the C-TEST button was pressed again. The model attempted to send a subsequent message at intervals of every 5 seconds, and the message was only sent when the ACK message of the previous message was received. With each iteration, the message size increased by ten. The screenshot on the right was taken a few seconds after the Lenovo A2020 device received the messages sent by the Lenovo A5000.




a) Device Lenovo A5000 initiated C-Test and sent messages with constant size increment

b) Device Lenovo A2020 received C-Test messages

Figure 4.9 Screenshot showing Lenovo A5000 sending constant increment message load to Lenovo A2020 with C-TEST

Figure 4.10 depicts a screenshot of the log generated by the Lenovo A5000. The log was recorded by Logcat, running on a Lenovo ThinkBook 14, during the execution of C-TEST. The data logged follows the same format as the log generated from F-TEST. The screenshot of the log file logged from a sender device for the data exchange test case, before and after tabulation, can be found in Appendix E.

Figure 4.11 presents a screenshot of the log generated by the Lenovo A2020. The log was recorded by Logcat, which was running on an HP Pavilion dv4, after the execution of C-TEST on the Lenovo A5000. The format of the logged data is identical to the log generated by F-TEST. The screenshot of the log file logged from a receiver device for the data exchange test case, before and after tabulation, can be viewed in Appendix F.

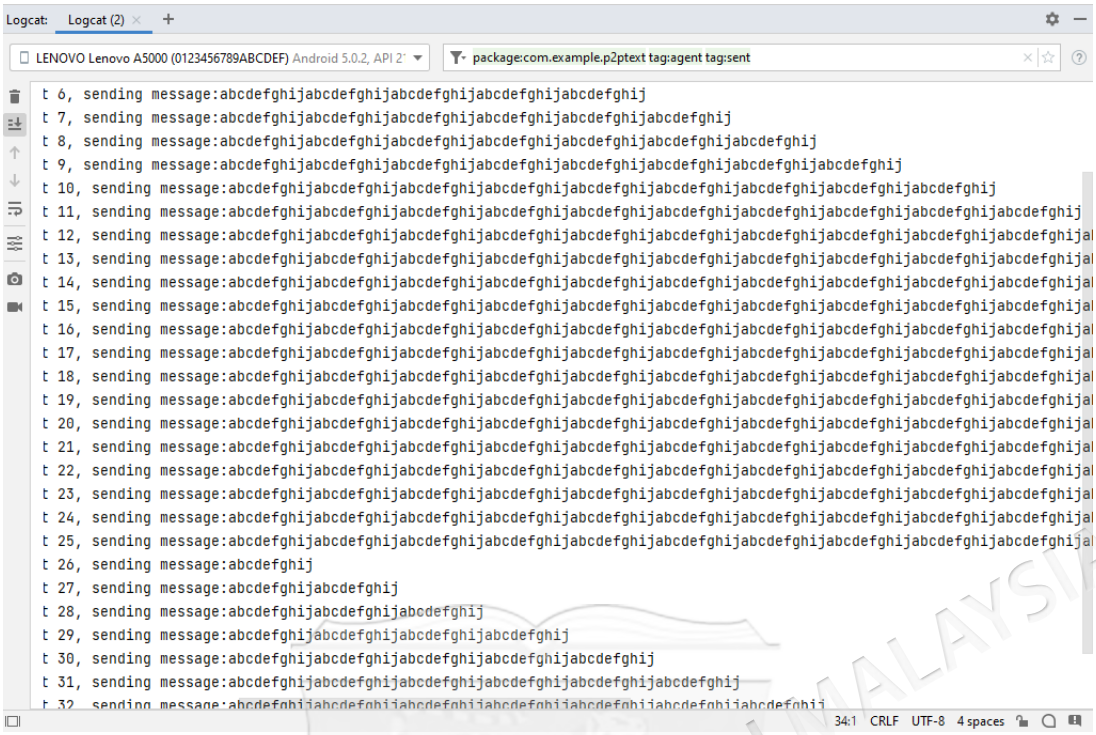



Figure 4.10 The screenshot of Logcat recording C-TEST sender data

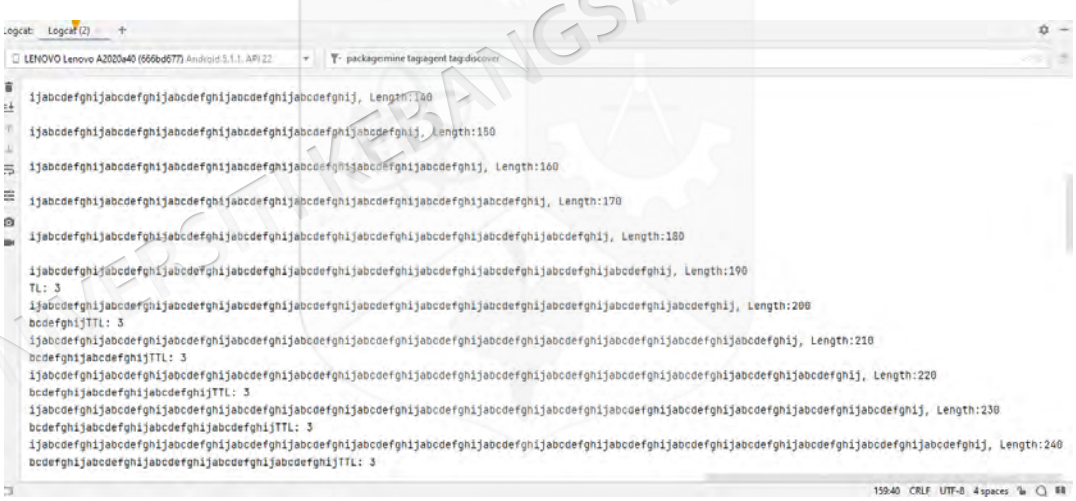



Figure 4.11 The screenshot of Logcat recording F-TEST receiver data

The screenshot of a log file logged from a sender device for the forwarding test case before and after tabulation can be viewed in Appendix G. The screenshot of a logged file logged from a forwarder or the middle device for the data forwarding test case before and after tabulation can be viewed in Appendix H. The screenshot of the log file logged from a receiver device for the data forwarding test case before and after tabulation can be viewed in Appendix I.

### 4.4.1 Test Result Description

As explained in section 3.5.2b, the performance parameters to be measured for each test case include goodput, average transmission time, and frame loss rate. This section explains the parameters measured from the experimental testing, according to the planned test cases.

**a. Result of test case 1 – data exchange**

**i. Constant message load**

The first test case consisted of two parts. The first part tested the data exchange performance between two devices with a constant message load of 60, 120, 180, and 240. The data exchange was sent from device A to device B 105 times, and vice versa. When filtering the dataset, the five with the highest divergence from the average were excluded from the results.

**Transmission time for constant message load**

Table 4.3 lists the data collected from the constant message load test cases with two smartphones. Figure 4.12 illustrates the column chart of the tabulated data. The maximum transmission time of all combined data recorded was 5698 ms. In other words, the slowest transmission time between two devices recorded was approximately 5.7 seconds. The minimum transmission time of all combined data recorded was 1 ms. In other words, the fastest transmission time between two devices recorded was approximately 0.001 seconds.

Figure 4.13 depicts the cumulative distribution frequency (CDF) chart of the overall transmission time recorded. It was noted that 80% of the transmission time was below 2525 ms. However, a significant variation between the maximum and average was observed. Therefore, the frequency data range listed in Table 4.4 was generated to identify the outliers. Figure 4.14 depicts the CDF chart of the overall transmission time recorded. It was noted that 80% of the transmission time was below 2525 ms. However, a significant variation between the maximum and average was observed. Therefore, the frequency data range listed in.

Table 4.3 Tabulated transmission time (constant message load)

| Transmission Time / Latency (ms) | | | | | |
|---|---|---|---|---|---|
| | **60** | **120** | **180** | **240** | **All** |
| Max | 1633 | 1855 | 1607 | 1676 | 1693 |
| Min | 4056 | 5698 | 4716 | 4445 | 5698 |
| Average | 1 | 117 | 32 | 69 | 1 |
| Std | 983 | 1068 | 955 | 898 | 983 |
| N | 200 | 200 | 200 | 200 | 800 |

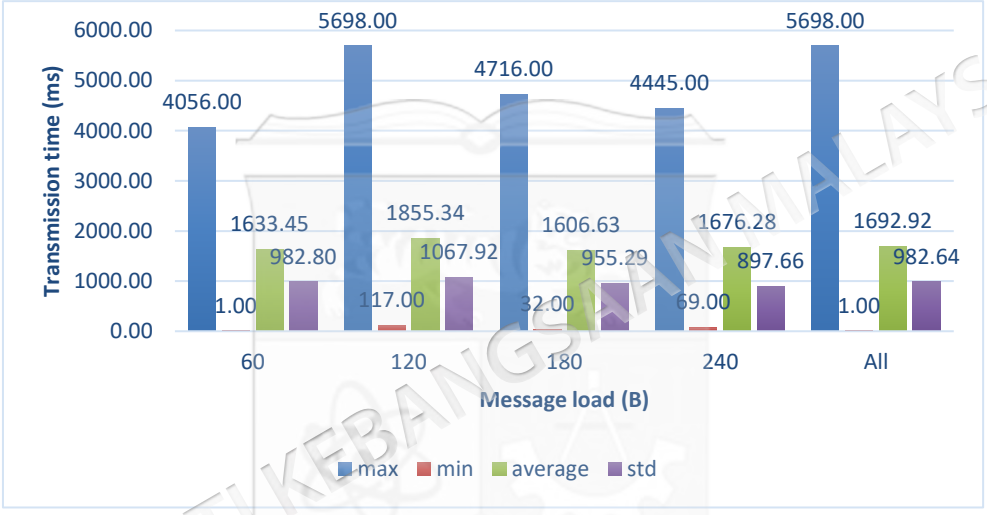



Figure 4.12 Column chart of transmission time (constant message load)

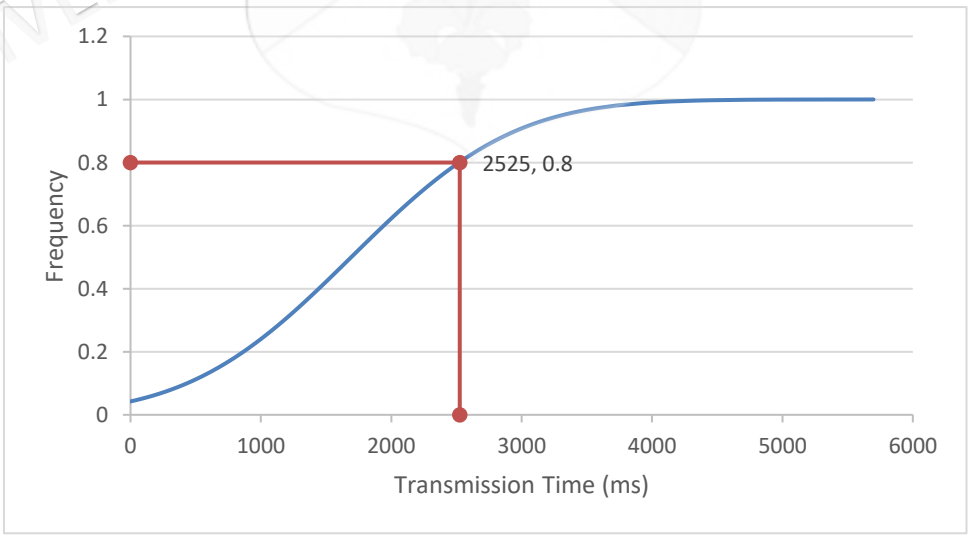



Figure 4.13 Cumulative Distribution Frequency graph of transmission time (constant message load)

Table 4.4 Tabulated transmission time in range of every 500 ms with the outliers highlighted (constant message load)

| **Transmission Time** | **n count for each message Load** | | | | |
|---|---|---|---|---|---|
| **Range (ms)** | 60 | 120 | 180 | 240 | Total |
| 1-500 | 31 | 20 | 28 | 23 | 102 |
| 501-1000 | 30 | 33 | 35 | 27 | 125 |
| 1001-1500 | 33 | 28 | 33 | 38 | 132 |
| 1501-2000 | 26 | 29 | 33 | 33 | 121 |
| 2001-2500 | 35 | 41 | 29 | 35 | 140 |
| 2501-3000 | 26 | 19 | 25 | 30 | 100 |
| 3001-3500 | 15 | 17 | 14 | 11 | 57 |
| 3501-4000 | 2 | 4 | 2 | 1 | 9 |
| 4001-4500 | 2 | 6 | | 2 | 10 |
| 4501-5000 | | 1 | 1 | | 2 |
| 5001-5500 | | 1 | | | 1 |
| 5501-6000 | | 1 | | | 1 |
| Total | 200 | 200 | 200 | 200 | 800 |

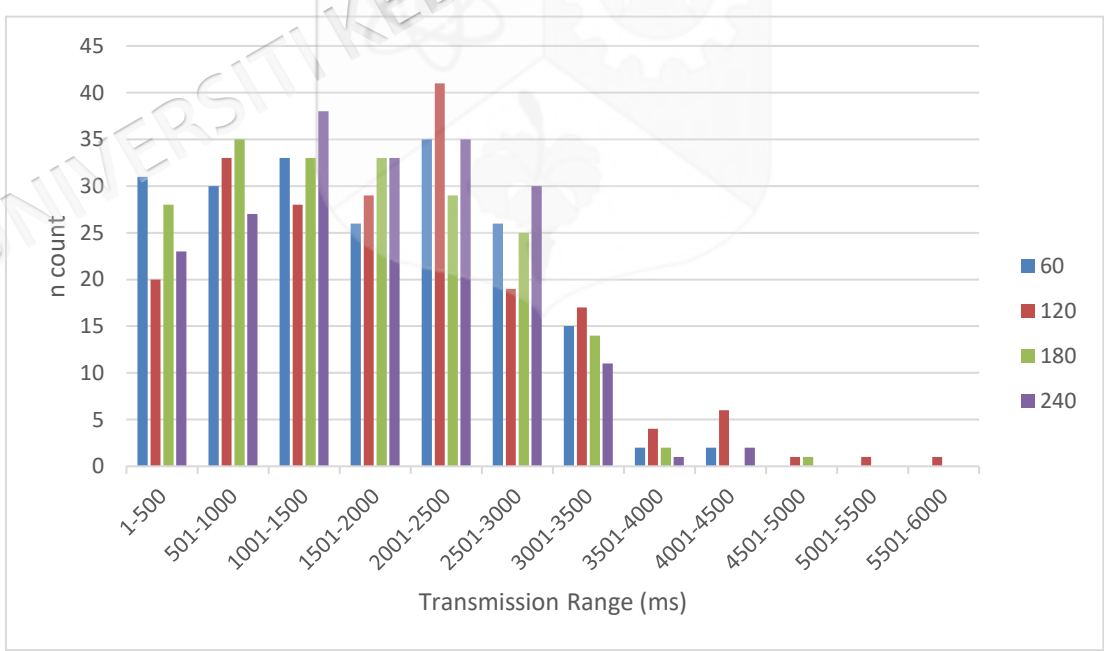



Figure 4.14 Column chart of transmission time counted in range of every 500 ms (constant message load)

Table 4.5 shows the calculated values after removing the 23 transmission time outliers. The average transmission time after the removal of outliers did not show a significant difference. However, the maximum transmission time and standard deviation calculated had decreased. Figure 4.15 illustrates the new CDF chart after the removal of outliers. It was noted that 80% of the transmission time was below 2379 ms, which was approximately 2.4 seconds.

Table 4.5 Transmission time data after filtered outliers (constant message load)

| **Transmission Time / Latency (ms)** | **After remove outlier** | **Before remove outlier** |
|---|---|---|
| Average | 1621.073 | 1693 |
| Max | 3495 | 5698 |
| Min | 1 | 1 |
| Std | 898.1864 | 983 |
| n | 777 | 800 |

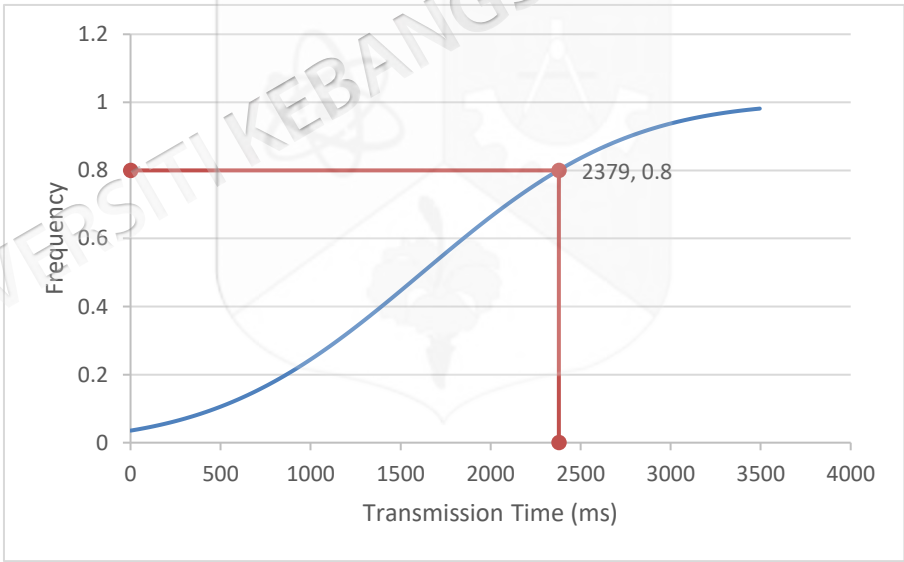



Figure 4.15 Cumulative Distribution Frequency graph of transmission time after filtered outliers (constant message load)

**Goodput for constant message load**

Table 4.6 lists the goodput calculated based on the data collected from the constant message load test cases with two smartphones. Figure 4.16 illustrates the column chart of the tabulated data. The maximum goodput of all combined data recorded was

468.75 kbps. The maximum goodput represents the fastest transmission speed of a message load between two devices. The minimum goodput of all combined data recorded was only 0.12 kbps. Figure 4.17 depicts the CDF chart of the overall transmission time recorded. It was noted that 80% of the goodput was below 17.2 kbps. It was observed that there was a significant variation between the maximum and average. Therefore, the frequency data range listed in Table 4.7 was generated to identify the outliers. Figure 4.18 illustrates the column chart of the frequency data range table. The goodput ranges beyond 5 kbps, with 39 data counts were identified as the outliers since the total count of the data range was the lowest among all others in the table.

Table 4.6 Tabulated goodput (constant message load)

| | Goodput (kbps) | | | | |
|---|---|---|---|---|---|
| | **60** | **120** | **180** | **240** | **All** |
| Max | 468.75 | 8.01 | 43.95 | 27.17 | 468.75 |
| Min | 0.12 | 0.16 | 0.30 | 0.42 | 0.12 |
| Average | 3.89 | 0.89 | 2.19 | 2.12 | 2.27 |
| Std | 33.84 | 1.04 | 4.94 | 3.22 | 17.22 |
| n | 200 | 200 | 200 | 200 | 800 |

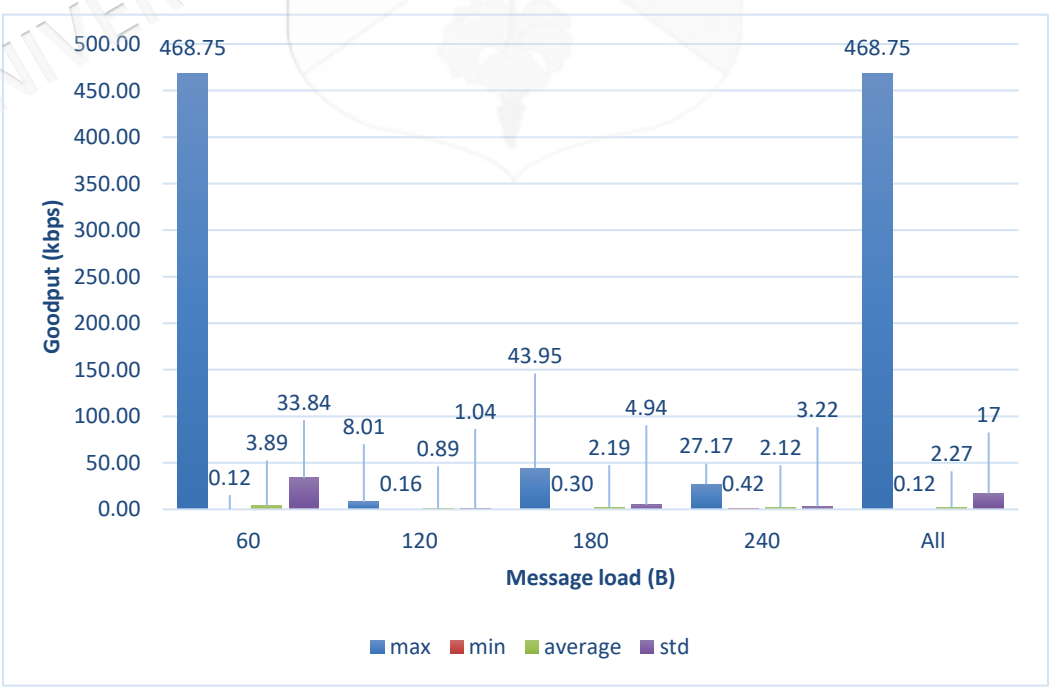



Figure 4.16 Column chart of goodput (constant message load)

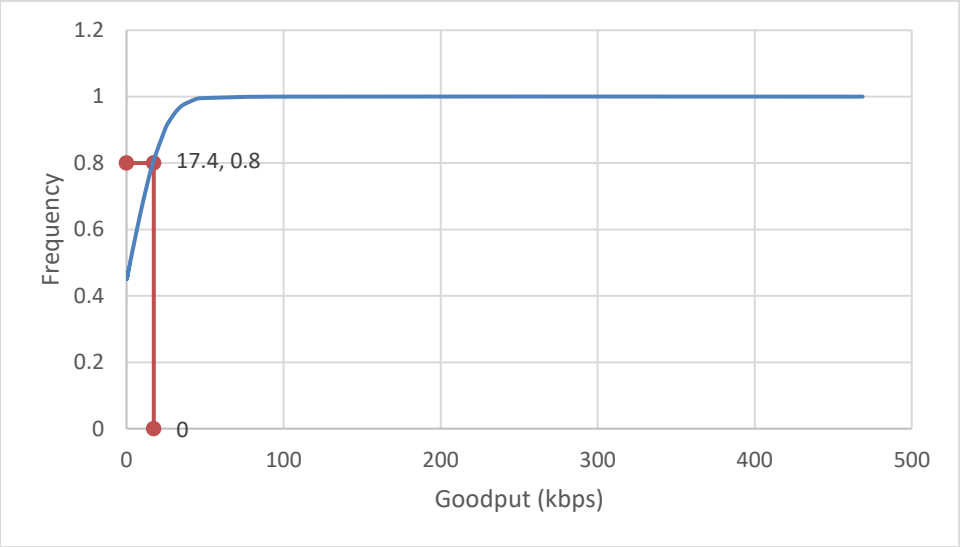



Figure 4.17 Cumulative Distribution Frequency graph of goodput (constant message load)

Table 4.7 Tabulated goodput in range of every 5 kbps with the outliers highlighted (constant message load)

| **Goodput** | **n count for each message Load** | | | | |
|---|---|---|---|---|---|
| **Range kb/s** | **60** | **120** | **180** | **240** | **Total** |
| 0-5 | 191 | 198 | 187 | 185 | 761 |
| 5-10 | 5 | 2 | 6 | 11 | 24 |
| 10-15 | | | 3 | | 3 |
| 15-20 | | | 1 | 2 | 3 |
| 20-25 | | | | 1 | 1 |
| 25-30 | | | | 1 | 1 |
| 30-35 | 1 | | 1 | | 2 |
| 35-40 | | | 1 | | 1 |
| 40-45 | | | 1 | | 1 |
| 45-50 | 1 | | | | 1 |
| 90-95 | 1 | | | | 1 |
| 465-470 | 1 | | | | 1 |
| Grand Total | 200 | 200 | 200 | 200 | 800 |

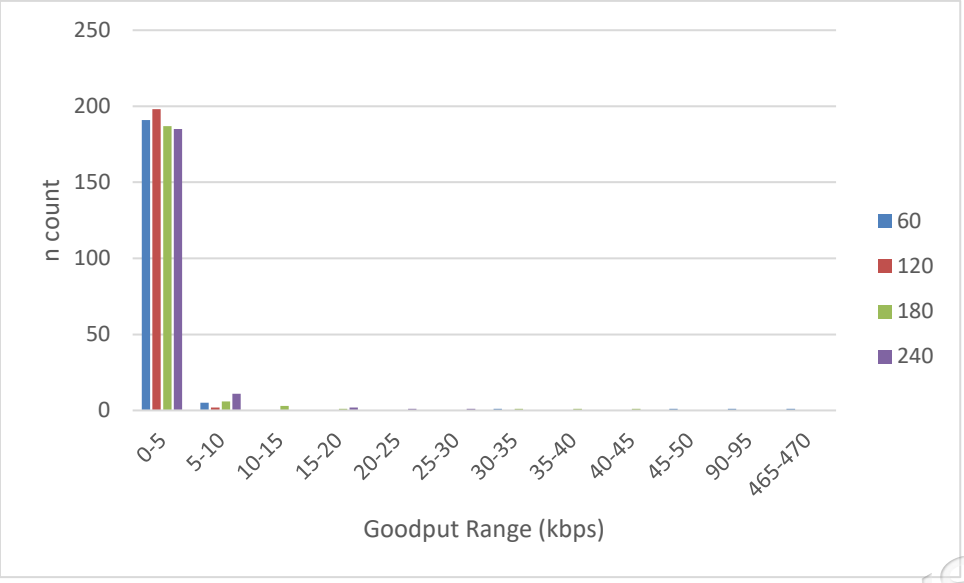



Figure 4.18 Column chart of goodput counted in range of every 5 kbps (constant message load)

Table 4.8 shows the calculated values after removing the 39 goodput outliers. The average goodput after the removal of outliers had dropped to approximately 56%. The maximum transmission time and standard deviation calculated had decreased approximately 99% and 94% respectively. Figure 4.19 illustrates the new CDF chart after the removal of outliers. It was noted that 80% of the goodput had dropped to 1.77 kbps from 17.4 kbps, which was approximately 90%.

Table 4.8 Goodput data after filtered outliers (constant message load)

| **Goodput (kbps)** | **After remove outlier** | **Before remove outlier** |
|---|---|---|
| Average | 0.989984 | 2.270664 |
| Max | 4.973475 | 468.75 |
| Min | 0.11557 | 0.11557 |
| Std | 0.916604 | 17 |
| n | 761 | 800 |

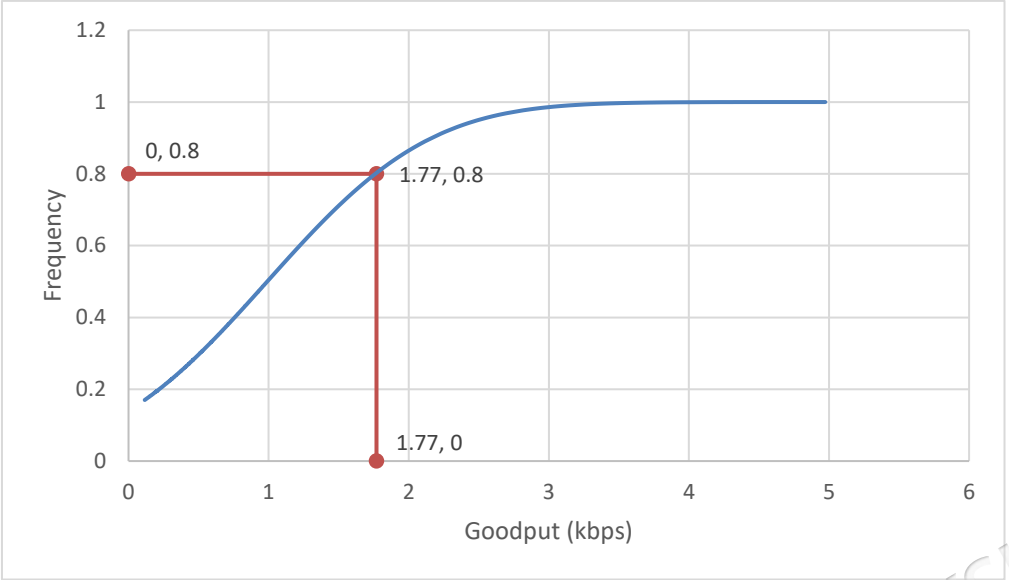



Figure 4.19 Cumulative Distribution Frequency graph of goodput after filtered outliers (constant message load)

**Frame loss rate for constant message load**

Table 4.9 presents the frame loss rate and delivery success rate of the testing with a constant message load. The frame loss rate for all tests of four different message loads was 0% because no data was dropped, and all data sent from sender was received by the receiver.

Table 4.9 Frame loss rate and delivery success rate (constant message load)

| Message Length | Message sent | Message received | Frame loss rate (%) | Data delivery Success rate (%) |
|---|---|---|---|---|
| 60 | 200 | 200 | 0 | 100 |
| 120 | 200 | 200 | 0 | 100 |
| 180 | 200 | 200 | 0 | 100 |
| 240 | 200 | 200 | 0 | 100 |

**ii. Constant increment message load**

The second part tested the data exchange performance between two devices, with the constant increment message load of 10 each time, until 240.

**Transmission time for increment message load**

Table 4.10 presents data collected from increment message load test cases with two smartphones. The average transmission time for all the combined recorded data was 2793 ms, equivalent to 2.8 seconds. The maximum transmission time for all combined recorded data was 75506 ms. In other words, the slowest transmission time between the two devices recorded was approximately 76 seconds. The minimum transmission time for all combined recorded data was 26 ms. In other words, the fastest transmission time between the two devices recorded was approximately 0.026 seconds.

Table 4.10 Tabulated transmission time (increment message load)

| | **Transmission time (ms)** | **Goodput (kb/s)** |
|---|---|---|
| Average | 2793 | 1.37 |
| Max | 75506 | 24.55 |
| Min | 26 | 0.01 |
| Std | 6751 | 3 |
| N | 240 | 240 |

Significant variation between the maximum and average transmission times was observed. Consequently, a frequency data range for transmission times was generated and listed in Table 4.11 to identify outliers. Figure 4.20 illustrates a column chart depicting the frequency data range. Transmission times exceeding 3500 ms, totalling 22 data points, were identified as outliers due to their comparatively lower count within the data range table.

Table 4.11 Tabulated transmission time in range of every 500 ms with the outliers highlighted (increment message load)

| **Transmission Time Range (ms)** | **n count for each transmission time range** |
|---|---|
| 1-500 | 38 |
| 501-1000 | 38 |
| 1001-1500 | 38 |

to be continued…

… continuation

| | |
|---|---|
| 1501-2000 | 29 |
| 2001-2500 | 33 |
| 2501-3000 | 31 |
| 3001-3500 | 11 |
| 3501-4000 | 2 |
| 4001-4500 | 1 |
| 5501-6000 | 5 |
| 6001-6500 | 1 |
| 7001-7500 | 1 |
| 7501-8000 | 1 |
| 8001-8500 | 1 |
| 9001-9500 | 1 |
| 9501-10000 | 1 |
| 12501-13000 | 1 |
| 15501-16000 | 2 |
| 17001-17500 | 1 |
| 19001-19500 | 1 |
| 38001-38500 | 1 |
| 55001-55500 | 1 |
| 75501-76000 | 1 |
| Grand Total | 240 |

Table 4.12 shows the calculated values after removing the 22 goodput outliers. The average transmission time after the removal of outliers had dropped to approximately 45%. The maximum transmission time and standard deviation calculated had decreased approximately 95% and 86% respectively. Figure 4.21 depicts the new CDF chart after the removal of outliers. It was noted that 80% of the transmission time was below 2305 ms.

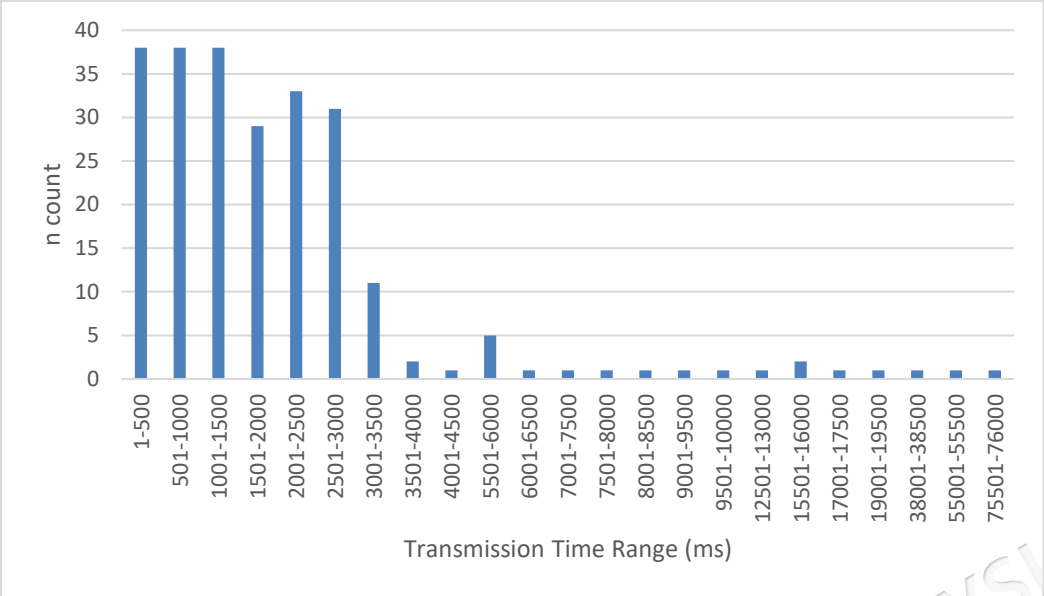



Figure 4.20 Column chart of transmission time counted in range of every 500 ms (increment message load)

Table 4.12 Transmission time data after filtered outliers (increment message load)

| Transmission Time / Latency (ms) | After remove outlier | Before remove outlier |
|---|---|---|
| Average | 1523 | 2793 |
| Max | 3475 | 75506 |
| Min | 26 | 26 |
| Std | 917 | 6751 |
| n | 218 | 240 |

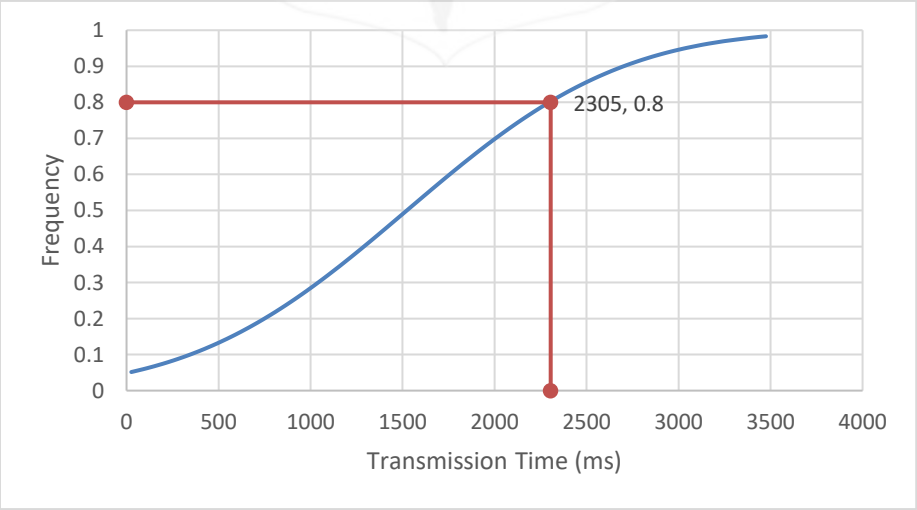



Figure 4.21 Cumulative Distribution Frequency graph of transmission time after filtered outliers (increment message load)

**Goodput for increment message load**

The frequency data range of the goodput listed in Table 4.13 was generated to identify the outliers. The goodput ranges above 5kbps with 14 data counts were identified as the outliers since the total count of the data range showed the lowest among all others in the table. Table 4.14 shows the calculated values after removing the 14 goodput outliers. The average goodput after the removal of outliers had dropped to approximately 39%. The maximum goodput and standard deviation calculated had decreased by approximately 81% and 67%, respectively. Figure 4.22 depicts the new CDF chart after the removal of outliers. It was noted that 80% of the goodput was below 1.587 kbps.

Table 4.13 Tabulated goodput in range of every 5 kbps (increment message load)

| Goodput (kbps) | n count for each Goodput range |
|---|---|
| 0-5 | 226 |
| 5-10 | 9 |
| 10-15 | 2 |
| 15-20 | 2 |
| 20-25 | 1 |
| Grand Total | 240 |

Table 4.14 Goodput data after filtered outliers (increment message load)

| **Goodput (kbps)** | **After remove outlier** | **Before remove outlier** |
|---|---|---|
| Average | 0.83 | 1.37 |
| Max | 4.64 | 24.55 |
| Min | 0.01 | 0.01 |
| Std | 1 | 3 |
| n | 226 | 240 |

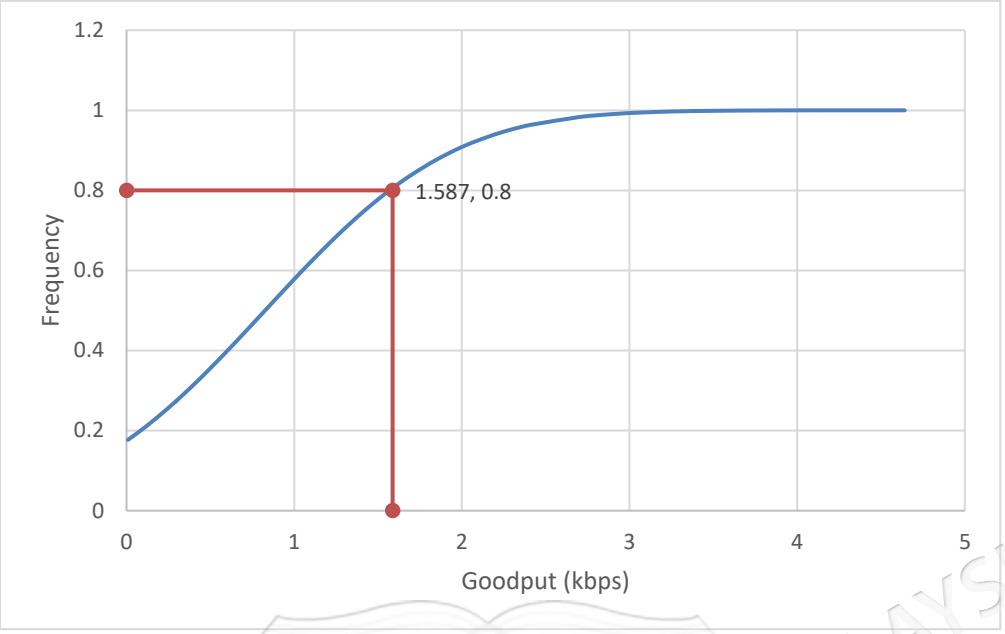



Figure 4.22 Cumulative Distribution Frequency graph of goodput after filtered outliers (increment message load)

**Frame loss rate for increment message load**

Table 4.9 presents the frame loss rate and delivery success rate of the testing with a progressively increasing message load. The frame loss rate for all test of four different message load in the increment message load test was 0% because no data was dropped. All data sent from sender was received by the receiver.

Table 4.15 Frame loss rate and delivery success rate (increment message load)

| Message Length | Message sent | Message received | Frame loss rate (%) | Data delivery Success rate (%) |
|---|---|---|---|---|
| From 10 to 240 | 240 | 240 | 0 | 100 |

**b. Result of test case 2 – data forwarding**

The second test case assessed data forwarding performance among three devices with a constant message load of 60, 120, 180, and 240.

### i. Transmission time for constant message load forwarding

Table 4.16 Tabulated transmission time (constant message load forwarding)

| Transmission Time / Latency (ms) | | | | | |
|---|---|---|---|---|---|
| | **60** | **120** | **180** | **240** | **All** |
| Max | 51092 | 110621 | 32829 | 42177 | 110621 |
| Min | 561 | 105 | 1287 | 764 | 105 |
| Average | 10337 | 12153 | 9483 | 7765 | 9935 |
| Std | 11826 | 20544 | 8070 | 8688.053 | 13346 |
| N | 100 | 100 | 100 | 100 | 400 |

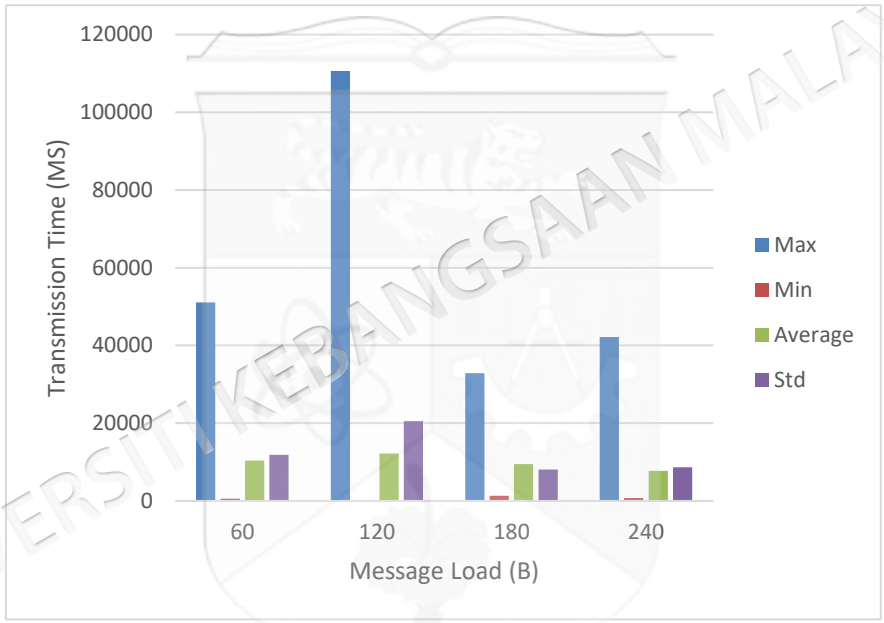



Figure 4.23 Column chart of transmission time (constant message load forwarding)

Table 4.16 lists the data collected from the constant message load forwarding test cases with three smartphones. Figure 4.23 illustrates the column chart of the tabulated data. The maximum transmission time of all combined data recorded was 110621 ms. The minimum transmission time of all combined data recorded was 105 ms. Figure 4.24 depicts the CDF chart of the overall transmission time recorded. It was noted that 80% of the transmission time was below 21370 ms. However, it was noted that there was a big variation between the maximum and average. Therefore, the frequency data range listed in Table 4.17 was generated to identify the outliers. Figure 4.25 depicts the column chart of the frequency data range table. The transmission time ranges

beyond 50000 ms with 8 data counts were identified as outliers since the total count of the data range showed the lowest among all others in the table.

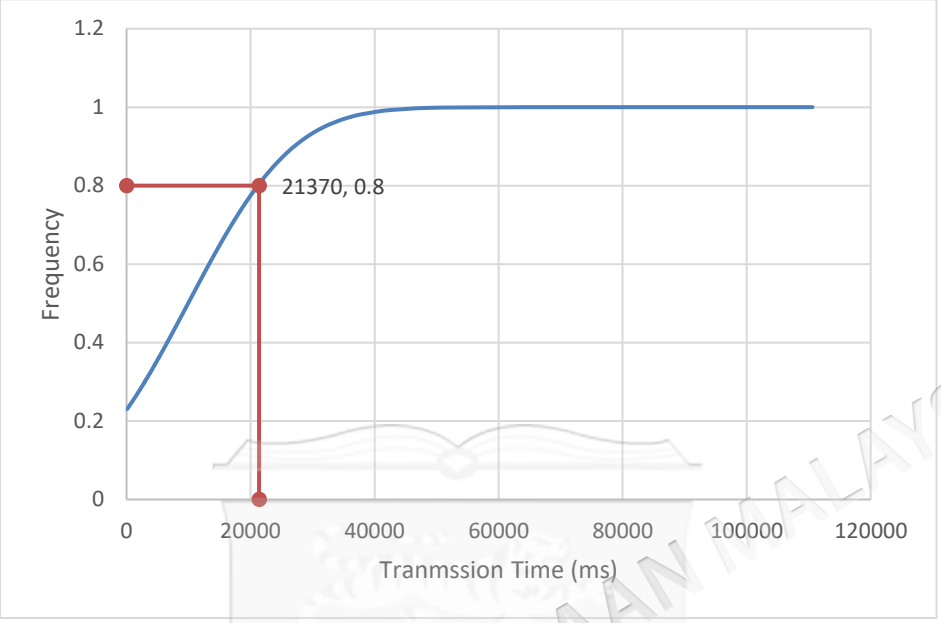



Figure 4.24 Cumulative Distribution Frequency graph of transmission time (constant message load forwarding)

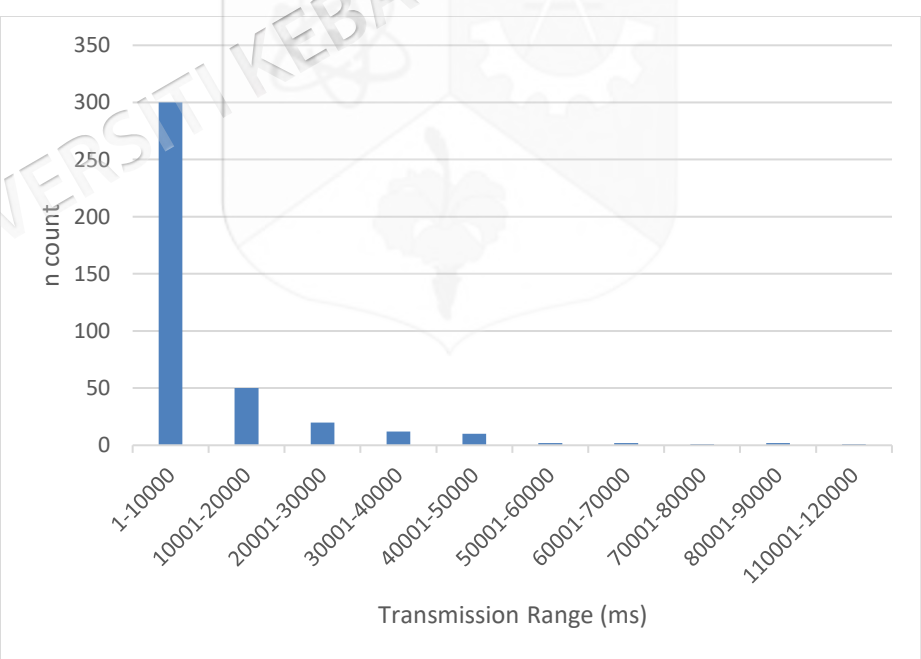



Figure 4.25 Column chart of transmission time counted in range of every 10000 ms (constant message load forwarding)

Table 4.17 Tabulated transmission time in range of every 10000 ms (constant message load forwarding)

| Transmission Time Range (ms) | n count for each transmission time range |
|---|---|
| 1-10000 | 300 |
| 10001-20000 | 50 |
| 20001-30000 | 20 |
| 30001-40000 | 12 |
| 40001-50000 | 10 |
| 50001-60000 | 2 |
| 60001-70000 | 2 |
| 70001-80000 | 1 |
| 80001-90000 | 2 |
| 110001-120000 | 1 |
| Grand Total | 400 |

Table 4.18 displays the calculated values after removing the eight transmission time outliers. The average transmission time, following the outlier removal, had decreased to around 13%. The maximum transmission time and the calculated standard deviation had both dropped by about 55% and 29%, respectively. Figure 4.26 illustrates the new CDF chart post-outlier removal. It was observed that 80% of the transmission time remained below 16737 ms.

Table 4.18 Transmission time data after filtered outliers (constant message load forwarding)

| Transmission Time / Latency (ms) | After remove outlier | Before remove outlier |
|---|---|---|
| Max | 49394 | 110621 |
| Min | 105 | 105 |
| Average | 8638 | 9935 |
| Std | 9511.059 | 13346 |
| n | 392 | 400 |

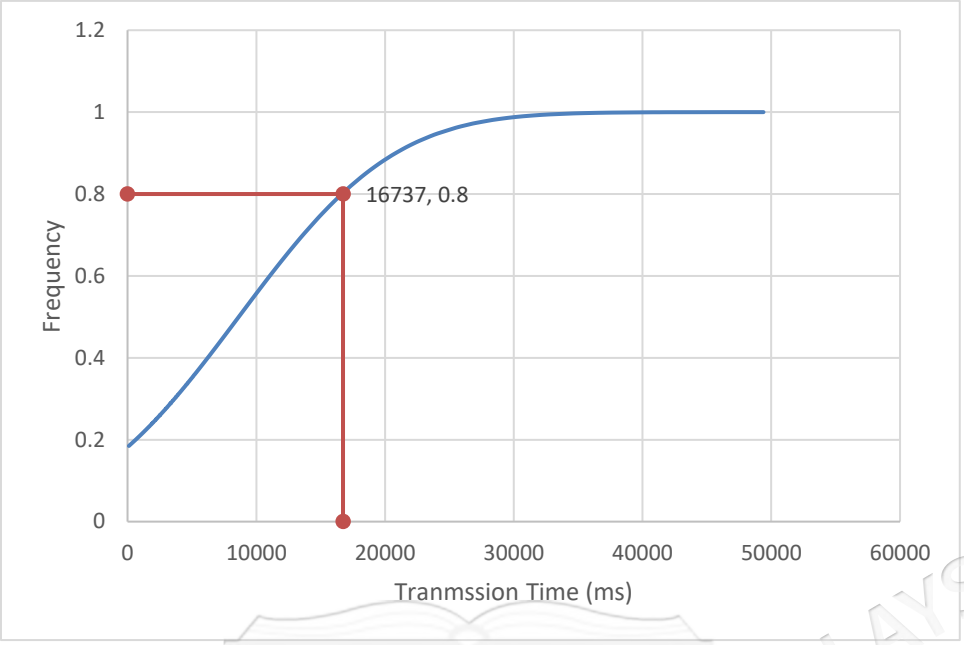



Figure 4.26 Cumulative Distribution Frequency graph of transmission time after filtered outliers (constant message load forwarding)

### ii. Goodput for constant message load forwarding

Table 4.19 lists the goodput calculated from the constant message load forwarding test cases with three smartphones. Figure 4.27 illustrates the column chart of the tabulated data. The maximum goodput of all combined data calculated was 8.9286 kbps. The minimum goodput of all combined data recorded was only 0.0085 kbps.

Table 4.19 Tabulated goodput (constant message load forwarding)

| | Goodput (kbps) | | | | |
|---|---|---|---|---|---|
| | **60** | **120** | **180** | **240** | **All** |
| Max | 0.8356 | 8.9286 | 1.0927 | 2.4542 | 8.9286 |
| Min | 0.0092 | 0.0085 | 0.0428 | 0.0445 | 0.0085 |
| Average | 0.1242 | 0.3215 | 0.2903 | 0.5190 | 0.3138 |
| Std | 0.1459 | 0.888443 | 0.23368 | 0.395195 | 0.524417 |
| n | 100 | 100 | 100 | 100 | 400 |

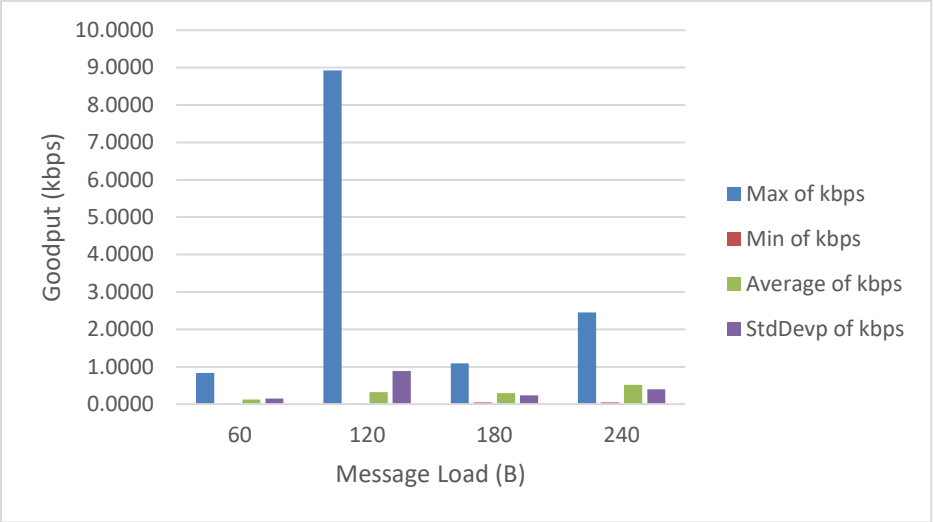



Figure 4.27 Column chart of goodput (constant message load forwarding)

Figure 4.28 depicts the CDF chart of the overall goodput recorded. It was noted that 80% of the transmission time was below 0.76 kbps. However, there was a significant variation between the maximum and average. Therefore, the frequency data range listed in Table 4.20 was generated to identify the outliers. Figure 4.29 depicts the column chart of the frequency data range table. The goodput ranges beyond 1 kbps with 15 data counts were identified as outliers since the total count of the data range showed the lowest among all others in the table.

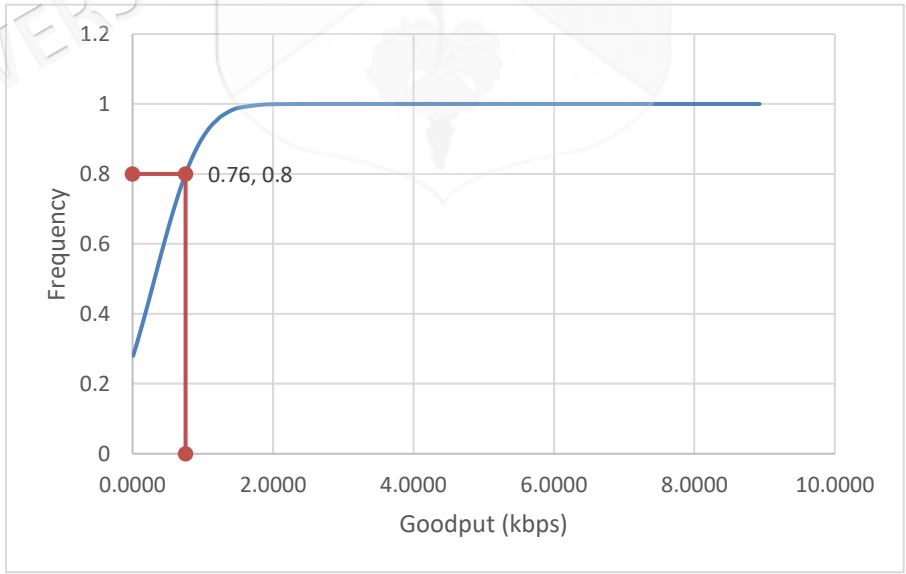



Figure 4.28 Cumulative Distribution Frequency graph of goodput (constant message load forwarding)

Table 4.20 Tabulated goodput in range of every 0.5 kbps (constant message load forwarding)

| Goodput Range (kbps) | n count for each goodput range |
|---|---|
| 0-0.5 | 332 |
| 0.5-1 | 53 |
| 1-1.5 | 11 |
| 1.5-2 | 2 |
| 2-2.5 | 1 |
| 8.5-9 | 1 |
| Grand Total | 400 |

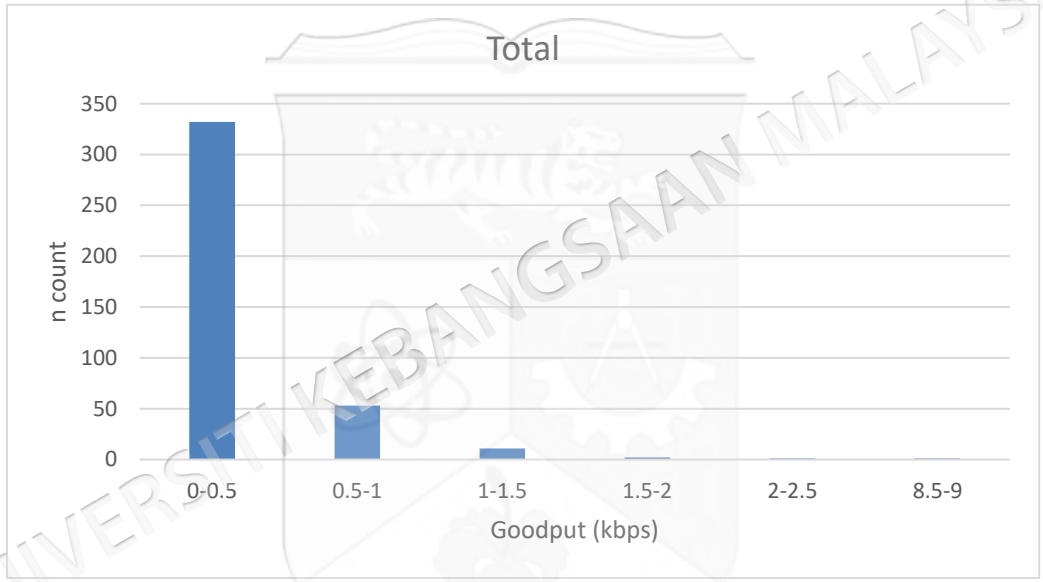



Figure 4.29 Column chart of goodput counted in range of every 0.5 kbps (constant message load forwarding)

Table 4.21 shows the calculated values after removing the 15 goodput outliers. The average goodput after the removal of outliers had dropped to approximately 18%. The maximum goodput and standard deviation calculated had decreased by approximately 89% and 58%, respectively. Figure 4.30 depicts the new CDF chart after the removal of outliers. It was noted that 80% of the goodput was below 0.44 kbps.

Table 4.21 Goodput data after filtered outliers (constant message load forwarding)

| Goodput (kbps) | After remove outlier | Before remove outlier |
|---|---|---|
| Max | 0.9638 | 8.9286 |
| Min | 0.0085 | 0.0085 |
| Average | 0.2554 | 0.3138 |
| Std | 0.219276 | 0.524417 |
| n | 385 | 400 |

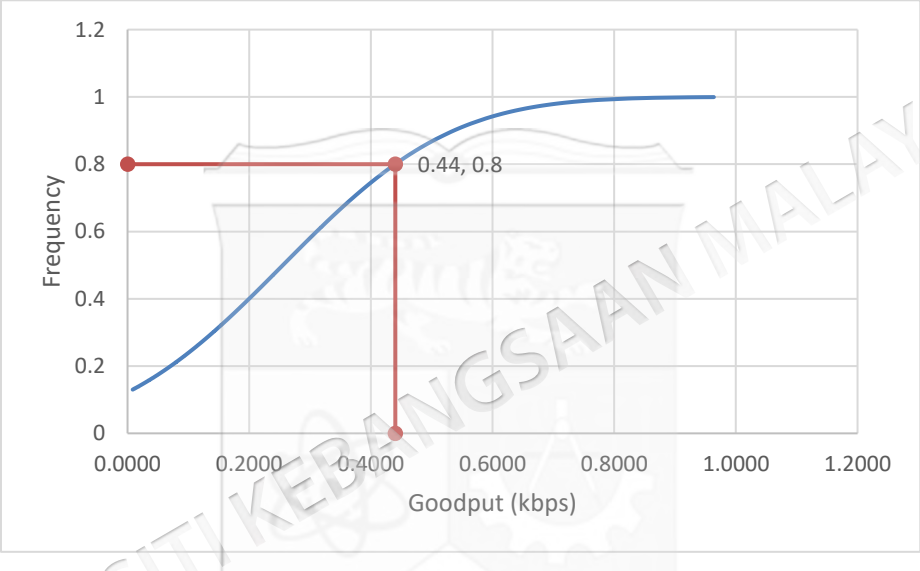



Figure 4.30 Cumulative Distribution Frequency graph of goodput after filtered outliers (constant message load forwarding)



### iii. Frame loss rate for constant message load forwarding

Table 4.22 Frame loss rate and delivery success rate (constant message load forwarding)

| Message Length | Message sent | Message received | Frame loss rate (%) | Data delivery Success rate (%) |
|---|---|---|---|---|
| 60 | 119 | 114 | 4.20 | 95.80 |
| 120 | 112 | 110 | 1.79 | 98.21 |
| 180 | 123 | 115 | 6.50 | 93.50 |
| 240 | 127 | 114 | 10.24 | 89.76 |

Table 4.22 lists the frame loss rate and data delivery success rate of the data forwarding test cases. The rates are tabulated in accordance with the four sets of

message loads. All four sets had a frame loss rate of less than 11%, implying that the data delivery success rates were all greater than 89%.

## 4.5 SUMMARY

This chapter was divided into three parts to align with the three objectives and the methodology outlined in CHAPTER III. In the first part, the results of the scoping review, which were part of the preliminary analysis process, were presented in tabular form. The second part involved the development of the prototype interface, along with screenshots of the prototype's operating procedures. In the third part, the data recorded from the experimental testing were presented in tables and charts. Based on the generated CDF graphs, the number of observations below a frequency of 0.8 for the measured parameters in each test case was identified.

The transmission times for constant message load and incremental message load data exchange between two devices were found to be under 2379 ms and 2305 ms, respectively. The transmission time for constant message load data forwarding among three devices was below 16737 ms, with an average of 8638 ms. The goodput for constant message load and incremental message load data exchange between two devices was found to be below 1.77 kbps and 1.587 kbps, respectively. The goodput for constant message load data forwarding among three devices was below 0.44 kbps, with an average of 0.2554 kbps.

The frame loss rate for constant message load and incremental message load data exchange between two devices was 0%. The frame loss rate for constant message load data forwarding among three devices was below 11%. The next chapter will explain the overall research summary, limitations, and future works.

.

# CHAPTER V

# CONCLUSION

## 5.1 INTRODUCTION

This chapter summarizes the overall research work, including the findings, limitations identified during the research activities, and future work to be done for the research. Figure 5.1 illustrates the overall research flow. Figure 5.2 lists the organization of the thesis.

## 5.2 RESEARCH FINDINGS

There were three objectives set for this research. The first objective aimed at identifying the constraints, limitations, and possible solutions of forming WCN. The objective was achieved through preliminary analysis, scoping review, and a comparative study of existing solutions conducted. The advantages and limitations of the works of other researchers in the same field were identified based on the preliminary analysis and scoping review conducted. In addition, the existing products available in the markets were systematically tested, and the strengths and weaknesses of each product were observed. The findings from the first objective were adopted for the second objective, which was to develop a new model of data exchange. The model developed mainly exploited the DNS SD protocol of WFD, utilizing the SRV TXT as the payload for data exchange. The developed model was realized in a prototype. Finally, a systematic experiment with test cases was conducted to demonstrate proof of concept for the model developed. The measurement and testing were conducted with the prototype deployed on smartphones, and experiments were carried out in a real-world environment. The three parameters measured were transmission time (latency), goodput, and frame loss rate. With the results and findings from the experimental testing, the third objective was completed.

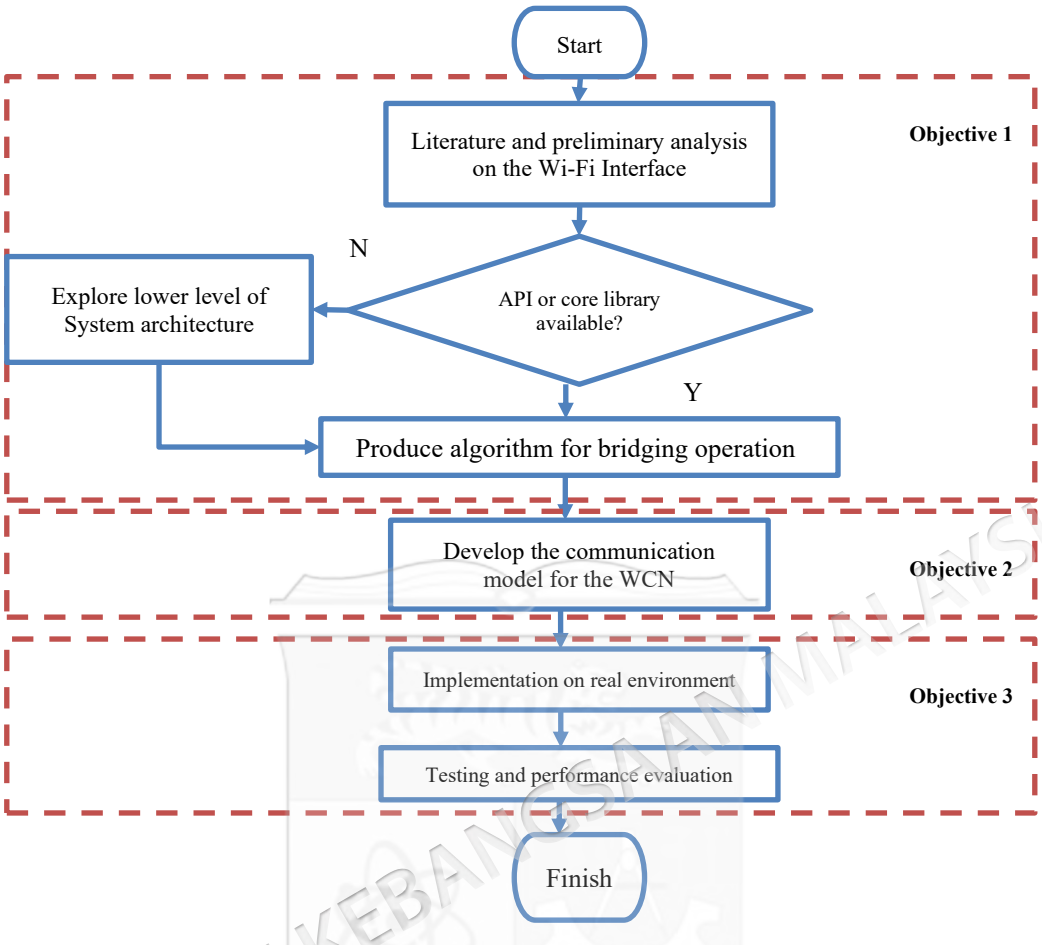



Figure 5.1 Overall research flow

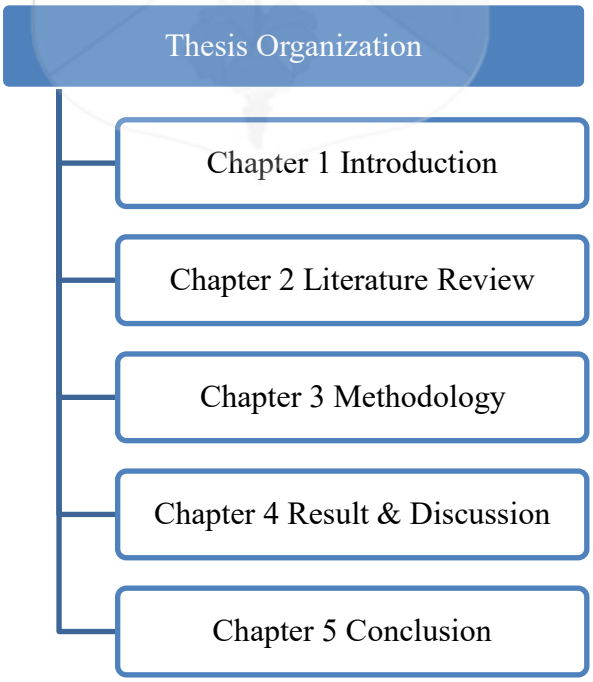



Figure 5.2 Thesis organization for the research

Based on the generated CDF chart, at a frequency of 0.8, the transmission times for the constant message load were under 2379 ms, and for the increment message load, they were under 2305 ms. This suggests that the data exchange between the two devices took less than 2.4 seconds with an 80% confidence level. Despite the relatively long transmission time of 16737 ms recorded in the constant message load data forwarding experiment involving three devices, this extended time was deemed acceptable due to the data forwarding model's reliance on the store-and-forward mechanism, falling within the DTN research domain.

The goodput for the data exchange of constant message loads and increment message loads between two devices was found to be below 1.77 kbps and 1.587 kbps, respectively. The goodput for constant message load data forwarding between three devices was below 0.44 kbps, with an average of 0.2554 kbps. Although the goodput rate appears low compared to the standard bandwidth of a Wi-Fi interface, which ranges between 1 and 2400 Mbps, such a comparison is not warranted. This is because the Wi-Fi protocol operates on a reliable data transmission protocol that ensures the consistency of the data transmission process. Additionally, the developed model was designed based on the best available communication medium within the OPPNet domain. The introduced model has contributed to the OPPNet research domain.

The frame loss rate for the data exchange of constant message loads and increment message loads between two devices was 0%. The frame loss rate for constant message load data forwarding between three devices was below 11%. These rates indicate that the developed model was capable of achieving a high success rate in delivering data.

## 5.3 LIMITATIONS

For the first objective, a systematic scoping review was conducted to gather the relevant literature for the research. However, it's important to note that according to the manual (JBI), the a priori protocol generated should be registered or published in

the repository of systematic reviews, such as JBI Evidence Synthesis[20], Cochrane Database of Systematic Reviews[21], and PROSPERO[22]. However, the a priori protocol developed in this research was not registered, considering that the conducted scoping review was not comprehensive. Nonetheless, the a priori protocol developed will be enhanced in the future with the inclusion of more robust objectives and questions.

For the second objective, the proposed model was developed and realized through a prototype. The exploited service searching and discovery procedures in the DNS SD protocol are based on the channel scanning mechanism. Inconsistency of the channels on devices will affect the transmission time of the proposed method. Channel selection is not controllable by the developer, as it is implemented at the lower layer of the Android framework. Therefore, the transmission time is dependent on the scanning and searching time. This was the main reason for the outliers in the collected transmission time dataset.

In the experimental testing, only three devices were used. Hence, the generated results could not be used to represent the performance of the model in a larger population. One possible solution to this is to implement simulators for experimental testing. Furthermore, all measurements were taken in an indoor workspace, and the nodes were stationary during the experiments. It's acknowledged that in real life, nodes are generally mobile and may enter or leave a network. Therefore, a simulator such as THE ONE can be used in future work. The experimental testing only measured three parameters for the purpose of demonstrating the proof of concept. The energy consumption rate has not been included as part of the measured parameters.



20 JBI Evidence Synthesis. (n.d.). https://journals.lww.com/jbisrir/pages/default.aspx

21 Cochrane Database of Systematic Reviews: all issues | Cochrane Library. (n.d.). https://www.cochranelibrary.com/cdsr/table-of-contents

22 PROSPERO. (n.d.). https://www.crd.york.ac.uk/PROSPERO/

## 5.4 CONCLUSION

In this chapter, the significance of the research findings is explained. Additionally, the limitations and future work of the research are highlighted. Connectivity issues due to traffic overload or weak infrastructure usually result in frustration. Furthermore, the failure of the communication network during a disaster is potentially fatal. Therefore, the developed model has contributed to the research domain OPPNet, and the prototype has been proven to work, potentially serving as a temporary alternative form of an ad-hoc communication network.

## 5.5 FUTURE WORKS

In recent years, with the rise of artificial intelligence (AI), many researchers have tried to apply artificial intelligence (AI) technology to the networking domain. As reported by Gan et al. (2022), the IEEE802.11 Working Group is exploring the feasibility of using Machine Learning (ML) to enhance the performance of Wi-Fi 7, also referred to as 802.11be. Szott et al. (2022) presented a summary of research articles with the aim of improving Wi-Fi performance using ML. The authors listed the application of ML in multi-hop networks, including Ad hoc networks, Mesh networks, sensor networks, and vehicular networks.

Yao et al. (2023) reported that other researchers have introduced several intelligent algorithm-based routing methods. For example, Tripathi et al. (2021) introduced the optimal routing with node prediction algorithm, which utilizes a neural network to predict the optimal routing in MANET or DTN. Dudukovich & Papachristou (2018) use ML classifiers in DTNs to determine neighbouring nodes that are likely to successfully deliver messages to their destinations. Apart from routing, ML can be implemented in multi-hop wireless settings to address other problems such as channel assignment, network deployment, and link quality prediction, as mentioned by Szott et al. (2022). The model introduced in this research can be combined with ML paradigms proposed by other researchers to manage data exchange operations in WCNs.

On-device AI is another emerging paradigm that aims to allow mobile devices such as smartphones, tablets, smartwatches, or even embedded systems to independently run AI algorithms without requiring a constant Internet connection or remote server (Feng & Feng, 2021; Hou et al., 2023). However, the efficiency and performance of the overall system solely depend on the processing power of the isolated devices (Flores et al., 2019). Therefore, networking plays an important role here to allow device-to-device (D2D) communication between the devices for data exchange, resource sharing, or task offloading to form a broader intelligent system. For the intelligent system to work smoothly, D2D communication should be faster, reliable, and precise (Wu & Hu, 2022). The proposed model in this research can be integrated into the on-device AI paradigm for more complex applications.

Since recent research has demonstrated a promising outcome through the utilization of ML in the MANET domain. For future work, intelligent routing protocols using the ML paradigm will be integrated into the model proposed in this study to form an intelligence based WCN. The routing protocols for DTN can be designed by integrating ML algorithms with well-known protocols in DTN, such as First Contact Routing, Direct Delivery Routing, PROPHET, Spray and Wait, among others(Yao et al., 2023).

In addition to improving the routing mechanism, the prototype developed in this study can also be enhanced by providing a list of nearby peers and sending messages only to specified peers. The battery consumption rate can also be included as an additional parameter to focus on, aiming to produce a solution with minimal power consumption. In terms of security, implementing an encryption algorithm can encrypt the message before transmission, ensuring end-to-end encryption.

A comprehensive experiment can be set up involving more devices. Additionally, test cases should consider the mobility of the devices. Nonetheless, a simulator such as ONE Simulator or NS3 can be utilized for a simulation covering a wider area and more devices.

**REFERENCES**


Abraham, T. (2011). Lessons from the pandemic: the need for new tools for risk and outbreak communication. *Emerging Health Threats Journal*, *4*(1), 7160. https://doi.org/10.3402/ehtj.v4i0.7160

Accenture. (2012). *Mobile Web Watch Internet Usage Survey 2012*. 11. http://www.accenture.com/sitecollectiondocuments/pdf/accenture-mobile-web-watch-internet-usage-survey-2012.pdf

Adeyeye, M., & Gardner-Stephen, P. (2011). The Village Telco project: a reliable and practical wireless mesh telephony infrastructure. *EURASIP Journal on Wireless Communications and Networking*, *2011*(1), 78. https://doi.org/10.1186/1687-1499-2011-78

Alami, M. El, Benamar, N., Younis, M., Shahin, A. A., El Alami, M., Benamar, N., Younis, M., & Shahin, A. A. (2017). A framework for hotspot support using Wi-Fi direct based device-to-device links. *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*, *December*, 552–557. https://doi.org/10.1109/IWCMC.2017.7986345

Alhamry, M., & Alomary, A. (2022). Exploring Wi-Fi WPA2-PSK protocol weaknesses. *2022 International Conference on Data Analytics for Business and Industry, ICDABI 2022*, 190–195. https://doi.org/10.1109/ICDABI56818.2022.10041465

Almeida, T. T., Júnior, J. G. R., Campista, M. E. M., & Costa, L. H. M. K. (2020). Wi-Fi direct performance evaluation for V2P communications. *Journal of Sensor and Actuator Networks*, *9*(2). https://doi.org/10.3390/JSAN9020028

*Android Studio Dolphin* (2021.3.1 Patch 1). (2023). Google LLC.

Aoude, L., Dawy, Z., Sharafeddine, S., Frenn, K., Jahed, K., & Ma, M. (2018). Social-Aware Device-to-Device Offloading Based on Experimental Mobility and Content Similarity Models. *Wirel. Commun. Mob. Comput.*, *2018*. https://doi.org/10.1155/2018/5606829

Arksey, H., & O'Malley, L. (2005). Scoping studies: towards a methodological framework. *International Journal of Social Research Methodology*, *8*(1), 19–32. https://doi.org/10.1080/1364557032000119616

Arnaboldi, V., Campana, M. G., & Delmastro, F. (2017). Context-Aware Configuration and Management of WiFi Direct Groups for Real Opportunistic Networks. *Proceedings - 14th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2017*, 266–274. https://doi.org/10.1109/MASS.2017.40

Aromataris, E. (2021). JBI Manual for Evidence Synthesis. In Z. Munn (Ed.), *JBI* (Issue April).

Bahia, K., & Suardi, S. (2019). Connected Society The State of Mobile Internet Connectivity 2019. In *Gsma*. http://www.comscore.com/Press_Events/Press_Releases/2010/6/comScore_Reports_April_2010_U.S._Mobile_Subscriber_Market_Share

Baresi, L., Derakhshan, N., & Guinea, S. (2016). WiDiSi: A Wi-Fi Direct Simulator. *2016 IEEE Wireless Communications and Networking Conference*, 1–7. https://doi.org/10.1109/WCNC.2016.7565169

Beech, M. (2020). COVID-19 Pushes Up Internet Use 70% And Streaming More Than 12%, First Figures Reveal. *Forbes*. https://www.forbes.com/sites/markbeech/2020/03/25/covid-19-pushes-up-internet-use-70-streaming-more-than-12-first-figures-reveal/

Bellavista, P. (2014). *Resource Sharing in Mobile Environments : the Next Frontier of Social Networking Novel Middleware for Green Exploitation of Social Sharing in Opportunistic Mobile Environments*. 2–3.

Bellavista, P., Corradi, A., & Giannelli, C. (2010). The real ad-hoc multi-hop Peer-to-peer (RAMP) middleware: An easy-to-use support for spontaneous networking. *Proceedings - IEEE Symposium on Computers and Communications*, 463–470. https://doi.org/10.1109/ISCC.2010.5546785

Bellavista, P., & Giannelli, C. (2009). Middleware solutions for self-organizing multi-hop multi-path internet connectivity based on bluetooth. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, *7 LNICST*, 58–71. https://doi.org/10.1007/978-3-642-01802-2_5

Biswas, J., & Veloso, M. (2010). WiFi localization and navigation for autonomous indoor mobile robots. *Proceedings - IEEE International Conference on Robotics and Automation*, 4379–4384. https://doi.org/10.1109/ROBOT.2010.5509842

Bluetooth Special Interest Group. (2014). Specification of the Bluetooth System Covered Core Package Version 4.2. *Bluetooth SIG*, *0*(April), 2272.

Bluetooth Special Interest Group. (2021). Bluetooth Core Specification v5.3. *Bluetooth SIG*, *July*. https://www.bluetooth.com/specifications/specs/core-specification-5-3/

Boldrini, C., Lee, K., Önen, M., Ott, J., & Pagani, E. (2014). Opportunistic networks. *Computer Communications*, *48*(December 2019), 1–4. https://doi.org/10.1016/j.comcom.2014.04.007

Camp, M. T. (2000). Development of the Bluetooth version 1.0 specification. *2000 IEEE Emerging Technologies Symposium on Broadband, Wireless Internet Access*. https://doi.org/10.1109/ETS.2000.916530

Camps-Mur, D., Garcia-Saavedra, A., & Serrano, P. (2013). Device to device communications with WiFi Direct: overview and experimentation. *IEEE Wireless Communications*, *20*(3), 96–104. http://enjambre.it.uc3m.es/~agsaaved/papers/2012_camps_wircommag.pdf

Cao, S., Chen, X., & Yuan, B. C. (2022). Overview of Short-range Wireless Communication Protocol. *2022 7th International Conference on Computer and Communication Systems, ICCCS 2022*, 519–523. https://doi.org/10.1109/ICCCS55155.2022.9846125

Casetti, C., Chiasserini, C. F., Duan, Y., Giaccone, P., & Manriquez, A. P. (2017). Data Connectivity and Smart Group Formation in Wi-Fi Direct Multi-group Networks. *IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT*. https://doi.org/10.1109/TNSM.2017.2766124

Casetti, C. E., Chiasserini, C. F., Duan, Y., Giaccone, P., & Perez Manriquez, A. (2018). Data connectivity and smart group formation in Wi-Fi direct multi-group networks. *IEEE Transactions on Network and Service Management*, *15*(1), 245–259. https://doi.org/10.1109/TNSM.2017.2766124

Chaki, P., Yasuda, M., & Fujita, N. (2015). Seamless Group Reformation in WiFi Peer to Peer network using dormant backend links. *2015 12th Annual IEEE Consumer Communications and Networking Conference, CCNC 2015*, 773–778. https://doi.org/10.1109/CCNC.2015.7158075

Chandra, R., Bahl, P., & Bahl, P. (2004). MultiNet: connecting to multiple IEEE 802.11 networks using a single wireless card. *IEEE INFOCOM 2004*, *2*, 882–893 vol.2. https://doi.org/10.1109/INFCOM.2004.1356976

Chang, W., Huang, B., Jia, B., Li, W., & Xu, G. (2023). Online Public Transit Ridership Monitoring Through Passive WiFi Sensing. *IEEE Transactions on Intelligent Transportation Systems*, *24*(7), 7025–7034. https://doi.org/10.1109/TITS.2023.3257835

Cisco. (2019). Cisco Visual Networking Index (VNI) Global Mobile Data Traffic Forecast Update, 2017-2022. In *Cisco*. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html

Conti, M., Boldrini, C., Kanhere, S. S., Mingozzi, E., Pagani, E., Ruiz, P. M., & Younis, M. (2015). From MANET to people-centric networking: Milestones and open research challenges. *Computer Communications*, *71*, 1–21. https://doi.org/10.1016/j.comcom.2015.09.007

Conti, M., Delmastro, F., Minutiello, G., & Paris, R. (2013). *Experimenting opportunistic networks with WiFi Direct*.

Daniel Lyons. (2011, February 1). Hackers' Egypt Rescue: Get Protesters Back Online. *NEWSWEEK DIGITAL LLC*. https://www.newsweek.com/hackers-egypt-rescue-get-protesters-back-online-68643

Djajadi, A., & Putra, R. J. (2014). Inter-cars safety communication system based on Android smartphone. *ICOS 2014 - 2014 IEEE Conference on Open Systems*, 12–17. https://doi.org/10.1109/ICOS.2014.7042402

Dubois, D. J., Bando, Y., Watanabe, K., Miyamoto, A., Sato, M., Papper, W., & Bove,

V. M. (2015). Supporting heterogeneous networks and pervasive storage in mobile content-sharing middleware. *2015 12th Annual IEEE Consumer Communications and Networking Conference, CCNC 2015*, 841–847. https://doi.org/10.1109/CCNC.2015.7158086

Dudukovich, R., & Papachristou, C. (2018). Delay Tolerant Network Routing as a Machine Learning Classification Problem. *2018 NASA/ESA Conference on Adaptive Hardware and Systems, AHS 2018*, 96–103. https://doi.org/10.1109/AHS.2018.8541460

ExactTarget. (2014). *2014 Mobile Behavior Report: Combining Mobile Device Tracking and Consumer Survey Data to Build a Powerul Mobile Strategy*. 1–36.

Felice, M. Di, Bedogni, L., & Bononi, L. (n.d.). *The Emergency Direct Mobile App : Safety Message Dissemination over a Multi-Group Network of Smartphones using Wi-Fi Direct*. 99–106.

Feng, R., & Feng, X. (2021). Robot, Edge Intelligence and Data Survey. *Proceedings - 2021 IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing and International Conference on Cyber Science and Technology Congress, DASC/PiCom/CBDCom/CyberSciTech 2021*, 843–848. https://doi.org/10.1109/DASC-PICom-CBDCom-CyberSciTech52372.2021.00140

Flores, H., Nurmi, P., & Hui, P. (2019). AI on the Move: From On-Device to On-Multi-Device. *2019 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2019*, 310–315. https://doi.org/10.1109/PERCOMW.2019.8730873

Gan, M., Yang, D. X., Au, E., AboulMagd, O., Montemurro, M., & McCann, S. (2022). *Look ahead to next generation*. https://mentor.ieee.org/802.11/dcn/22/11-22-0685-00-0wng-discussion-on-next-generation-wi-fi.pptx

Gardner-Stephen, P., Challans, R., Lakeman, J., Bettison, A., Gardner-Stephen, D., & Lloyd, M. (2013). The serval mesh: A platform for resilient communications in disaster & crisis. *Proceedings of the 3rd IEEE Global Humanitarian Technology Conference, GHTC 2013*, 162–166. https://doi.org/10.1109/GHTC.2013.6713674

Gardner-Stephen, P., Lakeman, J., Challans, R., Wallis, C., Stulman, A., & Haddad, Y. (2012). MeshMS: Ad Hoc Data Transfer within Mesh Network. *International Journal of Communications, Network and System Sciences*, *05*(08), 496–504. https://doi.org/10.4236/ijcns.2012.58060

Gardner-Stephen, P., & Palaniswamy, S. (2011). Serval mesh software-WiFi multi model management. *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief - ACWR '11*, *JANUARY 2011*, 71. https://doi.org/10.1145/2185216.2185245

Gebru, K., Rapelli, M., Rusca, R., Casetti, C., Chiasserini, C. F., & Giaccone, P.

(2022). Edge-based passive crowd monitoring through WiFi Beacons. *Computer Communications*, *192*(May), 163–170. https://doi.org/10.1016/j.comcom.2022.06.003

Global System for Mobile Communications (GSMA). (2019). *GSMA Mobile Connectivity Index*. https://www.mobileconnectivityindex.com/#year=2018&secondaryMenu=about-the-index

Gomes, T., Tapolcai, J., Esposito, C., Hutchison, D., Kuipers, F., Rak, J., De Sousa, A., Iossifides, A., Travanca, R., Andre, J., Jorge, L., Martins, L., Ugalde, P. O., Pasic, A., Pezaros, D., Jouet, S., Secci, S., & Tornatore, M. (2016). A survey of strategies for communication networks to protect against large-scale natural disasters. *Proceedings of 2016 8th International Workshop on Resilient Networks Design and Modeling, RNDM 2016*, *2011*, 11–22. https://doi.org/10.1109/RNDM.2016.7608263

Google LLC, & JetBrains. (2020). *Android Studio* (4.0). https://developer.android.com/studio

goTenna. (2023). *The World's Leading Mobile Mesh Networking Platform*. https://gotenna.com/#

Han, H., Liu, Y., Shen, G., & Member, S. (2014). *Design, Realization, and Evaluation of DozyAP for Power-Efficient Wi-Fi Tethering* (Vol. 22, Issue 5, pp. 1672–1685).

Han, H., Liu, Y., Shen, G., & Member, S. (2014). *DozyAP for PowerEfficient WiFi Tethering.pdf*. *22*(5), 1672–1685.

Holst, A. (2019). *Global Smartphone Sales to End Users from 1st Quarter 2009 to 2nd Quarter 2018, by Operating System*. Statista.Com. https://www.statista.com/statistics/266219/global-smartphone-sales-since-1st-quarter-2009-by-operating-system/

Holzer, A., De Tiberge, G., & Gillet, D. (2016). Towards Cloudless Co-Located Social Media on Android. *Proceedings of the 7th Annual Symposium on Computing for Development*. https://doi.org/10.1145/3001913.3006625

Hou, J., Zhang, E., & Long, Y. (2023). Measuring pedestrian flows in public spaces: Inferring walking for transport and recreation using Wi-Fi probes. *Building and Environment*, *230*(September 2022), 109999. https://doi.org/10.1016/j.buildenv.2023.109999

Hu, W., & Cao, G. (2017). Quality-Aware Traffic Offloading in Wireless Networks. *IEEE Transactions on Mobile Computing*, *16*(11), 3182–3195. https://doi.org/10.1109/TMC.2017.2690296

IEEE Computer Society. (2016). Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. In *The Institute of Electrical and Electronics Engineers*.

http://ieeexplore.ieee.org/servlet/opac?punumber=6178209

IEEE Computer Society. (2020). IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. In *The Institute of Electrical and Electronics Engineers*. http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html.%0Ahttps://standards.ieee.org/standard/802_11ax-2021.html

Intel Corporation. (n.d.). *Different Wi-Fi Protocols and Data Rates*. Intel Corporation. Retrieved February 8, 2023, from https://www.intel.com/content/www/us/en/support/articles/000005725/wireless/legacy-intel-wireless-products.html

International Telecommunication Union. (2019). *The World Telecommunication/ICT Indicators Database*. https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx

ITU-T Focus Group on Disaster Relief Systems Network Resilience and Recovery. (2013). *Technical report on Telecommunications and Disaster Mitigation* (Vol. 0).

Jiang, D., & Liu, G. (2017). An Overview of 5G Requirements. In W. Xiang, K. Zheng, & X. (Sherman) Shen (Eds.), *5G Mobile Communications* (pp. 3–26). Springer International Publishing. https://doi.org/10.1007/978-3-319-34208-5_1

Jundee, T., Phithakkitnukoon, S., & Ratti, C. (2023). Inferring Trips and Origin-Destination Flows from Wi-Fi Probe Data: A case study of campus Wi-Fi network. *IEEE Access*, *11*(May), 63351–63364. https://doi.org/10.1109/ACCESS.2023.3288283

Jung, W., Ahn, H., & Ko, Y. (2014). *Designing Content-Centric Multi-hop Networking over Wi-Fi Direct on Smartphones*. 2976–2981.

Junio, O. M., & Chavez, E. P. (2018). Development of Offline Chat Application: Framework for Resilient Disaster Management. *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, 510–514. https://doi.org/10.1109/IICSPI.2018.8690351

Kalbarczyk, T., & Julien, C. (2018). Omni: An Application Framework for Seamless Device-to-Device Interaction in the Wild. *Proceedings of the 19th International Middleware Conference*, 161–173. https://doi.org/10.1145/3274808.3274821

Keränen, A., Ott, J., & Kärkkäinen, T. (2009). The ONE simulator for DTN protocol evaluation. *SIMUTools 2009 - 2nd International ICST Conference on Simulation Tools and Techniques*. https://doi.org/10.4108/ICST.SIMUTOOLS2009.5674

Khaled, Z. E. L., & Mcheick, H. (2019). *Case studies of communications systems during harsh environments : A review of approaches , weaknesses , and limitations to improve quality of service*. *15*(2).

https://doi.org/10.1177/1550147719829960

Khan, M. A., Cherif, W., Filali, F., & Hamila, R. (2017). *Realization of Dual-hop Networks in Wi-Fi Direct and Performance Evaluation*. 4–11. https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2017.87

Khan, M. A., Cherif, W., Filali, F., & Hamila, R. (2017). Wi-Fi Direct Research - Current Status and Future Perspectives. *Journal of Network and Computer Applications*, *93*, 245–258. https://doi.org/10.1016/j.jnca.2017.06.004

Khawaja, I. A., Abid, A., Farooq, M. S., Shahzada, A., Farooq, U., & Abid, K. (2020). Ad-Hoc Collaboration Space for Distributed Cross Device Mobile Application Development. *IEEE Access*, *8*, 62800–62814. https://doi.org/10.1109/ACCESS.2020.2980319

Lee, T. S., & Sulaiman, R. (2019). Preliminary analysis of wireless collaborative network on mobile devices. *Journal of Information and Communication Technology*, *18*(3), 327–343. https://doi.org/10.32890/jict2019.18.3.5

Lee, T. S., & Sulaiman, R. (2016). Wireless Collaborative Network on Mobile Devices. *3rd Visual Informatics International Seminar (VIIS 2016)*, 1–6. http://www.iir4.ukm.my/the-visual-informatics-international-seminar-2016

Lee, T. S., Sulaiman, R., & Ali, N. M. (2018). A Peer to Peer Internet Sharing Technique on MANET Through Wi-Fi Interface. *4th Visual Informatics International Seminar (VIIS 2018)*, 1–12. https://www.researchgate.net/publication/334697264_A_Peer_to_Peer_Internet_Sharing_Technique_on_MANET_Through_Wi-Fi_Interface

Levac, D., Colquhoun, H., & O'Brien, K. K. (2010). Scoping studies: advancing the methodology. *Implementation Science*, *5*(1), 69. https://doi.org/10.1186/1748-5908-5-69

Li, F., Guo, Z., Liang, B., Yi, X., Wang, X., Li, W., & Wang, Y. (2021). A measurement study on device-to-device communication technologies for IIoT. *Computer Networks*, *192*. https://doi.org/10.1016/j.comnet.2021.108072

Li, F., Wang, X., Wang, Z., Cao, J., Liu, X., Bi, Y., Li, W., & Wang, Y. (2020). A local communication system over Wi-Fi direct: Implementation and performance evaluation. *IEEE Internet of Things Journal*, *7*(6), 5140–5158. https://doi.org/10.1109/JIOT.2020.2976114

Liemhetcharat, S., & Veloso, M. (2012). Multi-robot map-merging-free connectivity-based positioning and tethering in unknown environments. *Progress in Artificial Intelligence*, *1*(1), 3–23. https://doi.org/10.1007/s13748-011-0007-1

Links, P., El, M., Nabil, A., Mohamed, B., & Ahmed, Y. (2019). A Framework for Enabling Internet Access Using Wi - Fi. *Wireless Personal Communications*, *109*(1), 521–538. https://doi.org/10.1007/s11277-019-06577-7

Liu, K., Shen, W., Yin, B., Cao, X., Cai, L. X., & Cheng, Y. (2016). Development of

Mobile Ad-hoc Networks over Wi-Fi Direct with off-the-shelf Android phones. *2016 IEEE International Conference on Communications, ICC 2016*. https://doi.org/10.1109/ICC.2016.7511190

Liu, Y., Huang, Z., Li, W., & Ji, Y. (2016). Game theory-based mode cooperative selection mechanism for device-to-device visible light communication. *Optical Engineering*, *55*(3). https://doi.org/10.1117/1.OE.55.3.030501

Lonetti, F., & Marchetti, E. (2018). Emerging Software Testing Technologies. *Advances in Computers*, *108*, 91–143. https://doi.org/10.1016/BS.ADCOM.2017.11.003

Maia, M. E. F., Andrade, R. M. C., Filho, C. A. B. D. Q., Braga, R. B., Aguiar, S., Mateus, B. G., Nogueira, R., & Toorn, F. (2014). USABle - A communication framework for ubiquitous systems. *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, 81–88. https://doi.org/10.1109/AINA.2014.162

Malaysian Communications and Multimedia Commission. (2016). Internet Users Survey 2016. In *Malaysian Communications and Multimedia Commission*. https://doi.org/ISSN 1823-2523

Malaysian Communications and Multimedia Commission. (2017). Internet Users Survey 2017. In *Malaysian Communications and Multimedia Commission*.

Mao, Z., Ma, J., Jiang, Y., & Yao, B. (2017). Performance evaluation of WiFi Direct for data dissemination in mobile social networks. *Proceedings - IEEE Symposium on Computers and Communications*, 1213–1218. https://doi.org/10.1109/ISCC.2017.8024690

Michel Lombera, I., Moser, L. E., Melliar-Smith, P. M., & Chuang, Y. T. (2014). Peer-to-peer publication, search and retrieval using the Android mobile platform. *Computer Networks*, *65*, 56–72. https://doi.org/10.1016/j.comnet.2014.03.002

Mockapetris, P. V. (1987). *Domain names - implementation and specification*. http://tools.ietf.org/html/rfc1035

Moissinac, K., Ramos, D., Rendon, G., & Elleithy, A. (2021). Wireless Encryption and WPA2 Weaknesses. *2021 IEEE 11th Annual Computing and Communication Workshop and Conference, CCWC 2021*, 1007–1015. https://doi.org/10.1109/CCWC51732.2021.9376023

Motlagh, N. H. (2015). *Near Field Communication ( NFC ) - A technical Overview* (Issue November) [University of VAASA]. https://doi.org/10.13140/RG.2.1.1232.0720

New Straits Times. (2020, April 9). #TECH: Slow internet speed as more Malaysians go online during MCO. *New Straits Times Online*. https://www.nst.com.my/lifestyle/bots/2020/04/582739/tech-slow-internet-speed-more-malaysians-go-online-during-mco

Niranjan, K., Nithyesh, S., Nitin, P., Prashanth, S., & Venkatasubramanian, S. (2015). WiFiTether: A software based approach to Wi-Fi tethering. *International Journal of Applied Engineering Research*, *10*(20), 18047–18051. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84942779717&partnerID=40&md5=9ba54524e7a4b0c8466b063803389001

Ocana, M., Bergasa, L. M., Sotelo, M. A., Nuevo, J., & Flores, R. (2005). Indoor robot localization system using WiFi signal measure and minimizing calibration effort. *IEEE International Symposium on Industrial Electronics*, *IV*, 1545–1550. https://doi.org/10.1109/ISIE.2005.1529162

Olson, P. (2014, June). Could This App Create A Free, Secret Web? *Forbes*. https://www.forbes.com/sites/parmyolson/2014/06/05/could-this-app-create-a-free-secret-web/?sh=53e96a0d4602

Osman, M. A., Talib, A. Z., & Sanusi, Z. A. (2012). *A Study of the Trend of Smartphone and its Usage Behavior in Malaysia*. *2*(1), 275–286. https://doi.org/10.1007/978-3-642-22603-8_35

Oughton, E., Amaglobeli, D., & Moszoro, M. (2023). Estimating Digital Infrastructure Investment Needs to Achieve Universal Broadband. In *IMF Working Papers* (Vol. 2023, Issue 027). https://doi.org/10.5089/9798400233623.001

Pahlavan, K., & Krishnamurthy, P. (2021). Evolution and Impact of Wi-Fi Technology and Applications: A Historical Perspective. *International Journal of Wireless Information Networks*, *28*(1), 3–19. https://doi.org/10.1007/s10776-020-00501-8

Pan, M.-S., & Lin, Y.-P. (2018). Efficient Data Dissemination for Wi-Fi Peer-to-Peer Networks by Unicasting among Wi-Fi P2P Groups. *Wirel. Netw.*, *24*(8), 3063–3081.

Pan, M.-S., & Lin, Y.-P. (2018). Efficient Data Dissemination for Wi-Fi Peer-to-Peer Networks by Unicasting among Wi-Fi P2P Groups. *Wirel. Netw.*, *24*(8), 3063–3081.

Pan, M.-S., & Wang, C.-M. (2019). A Group-Less and Energy Efficient Communication Scheme Based on Wi-Fi Direct Technology for Emergency Scenes. *IEEE Access*, *7*, 31840–31853. https://doi.org/10.1109/ACCESS.2019.2903228

Pan, M. S., & Lin, Y. P. (2018). Efficient data dissemination for Wi-Fi peer-to-peer networks by unicasting among Wi-Fi P2P groups. *Wireless Networks*, *24*(8), 3063–3081. https://doi.org/10.1007/s11276-017-1522-1

Parziale, L., Liu, W., Matthews, C., Rosselot, N., Davis, C., Forrester, J., Britt, D. T., & Redbooks, I. B. M. (2006). *TCP/IP Tutorial and Technical Overview*. IBM Redbooks. https://books.google.com.my/books?id=TgwWAgAAQBAJ

Pyattaev, A., Johnsson, K., Surak, A., Florea, R., Andreev, S., & Koucheryavy, Y.

(2016). Network-assisted D2D communications: Implementing a technology prototype for cellular traffic offloading. *IEEE Wireless Communications and Networking Conference, WCNC*, 3266–3271. https://doi.org/10.1109/WCNC.2014.6953070

Raj, M., Kant, K., & Das, S. K. (2014). E-DARWIN: Energy Aware Disaster Recovery Network using WiFi Tethering. *Proceedings - International Conference on Computer Communications and Networks, ICCCN*. https://doi.org/10.1109/ICCCN.2014.6911770

Raj, M., Kant, K., & Das, S. K. (2014). E-DARWIN: Energy Aware Disaster Recovery Network using WiFi Tethering. *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, 1–8. https://doi.org/10.1109/ICCCN.2014.6911770

Ramezanpour, K., Jagannath, J., & Jagannath, A. (2023). Security and privacy vulnerabilities of 5G/6G and WiFi 6: Survey and research directions from a coexistence perspective. *Computer Networks*, *221*(November 2021). https://doi.org/10.1016/j.comnet.2022.109515

Reich, J., Misra, V., Rubenstein, D., & Zussman, G. (2008). *Spreadable Connected Autonomic Networks ( SCAN ) Technical Report CUCS-016-08*.

Rodrigues, J., Silva, J., Martins, R., Lopes, L., Drolia, U., Narasimhan, P., & Silva, F. (2016). Benchmarking wireless protocols for feasibility in supporting crowdsourced mobile computing. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9687*, 96–108. https://doi.org/10.1007/978-3-319-39577-7_8

Rusca, R., Sansoldo, F., Casetti, C., & Giaccone, P. (2023). What WiFi Probe Requests can tell you. *Proceedings - IEEE Consumer Communications and Networking Conference, CCNC*, *2023-Janua*, 1086–1091. https://doi.org/10.1109/CCNC51644.2023.10060447

S. Bradner. (1991). *Benchmarking Terminology for Network Interconnection Devices (RFC1242)*.

S. Bradner, & McQuaid, J. (1999). *Benchmarking Methodology for Network Interconnect Devices (RFC2544)*.

S. Cheshire, & Krochmal, M. (2013). *DNS-Based Service Discovery (RFC 6763)*.

Schoonwinkel, D. (2016). *Practical Measurements of Wi-Fi Direct in Content Sharing , Social and Gaming Android Applications*. University of Stellenbosch.

Scott, M. (2020, March 6). Empty Offices, Full Homes: Covid-19 Might Strain the Internet. *Bloomberg Technology*. https://www.bloomberg.com/news/articles/2020-03-06/empty-offices-full-homes-covid-19-might-strain-the-internet

Sha, H., Tasiopoulos, A. G., Psaras, I., & Pavlou, G. (2016). A collaborative video

download application based on Wi-Fi direct. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9674*, 147–158. https://doi.org/10.1007/978-3-319-33936-8_12

Shahin, A. A., & Younis, M. (2015). Alert dissemination protocol using service discovery in Wi-Fi direct. *IEEE International Conference on Communications*, *2015-Septe*, 7018–7023. https://doi.org/10.1109/ICC.2015.7249445

Shahin, A. A., & Younis, M. (2015). Efficient Multi-Group Formation and Communication Protocol for Wi-Fi Direct. *Proceedings of the 2015 IEEE 40th Conference on Local Computer Networks (LCN 2015)*, 233–236. https://doi.org/10.1109/LCN.2015.7366314

Sit, T. C. H., Liu, Z., Ang, M. H., & Seah, W. K. G. (2007). Multi-robot mobility enhanced hop-count based localization in ad hoc networks. *Robotics and Autonomous Systems*, *55*(3), 244–252. https://doi.org/10.1016/j.robot.2006.08.005

*Socket Programming*. (n.d.). Https://Www.Cs.Dartmouth.Edu/. Retrieved February 25, 2023, from https://www.cs.dartmouth.edu/~campbell/cs60/socketprogramming.html

Solodo, I., & Malarić, K. (2013). Wi-Fi Parameter Measurements and Analysis. *MEASUREMENT 2013, Proceedings of the 9th International Conference, Smolenice, Slovakia*, 339–342.

State Emergency Management Committee-Western Australia. (2012). *Emergency Preparedness Report 2012* (Issue October). http://pandora.nla.gov.au/pan/134965/20131022-0008/www.dpc.wa.gov.au/Publications/Documents/WA+State+Emergency+Preparedness+Report+2012.pdf

Stevens, G., Ilba, D., & Wildy, S. (2014). Telecommunications Tower. *IEEE 2014 Global Humanitarian Technology Conference*.

Sunil, S., Mukhopadhyay, A., & Gujjar, C. (2020). *Multi-Group Message Communication on Android Smartphones via WiFi Direct*. 1994–1999.

Sunny, N., & Shobha, S. (2016). *NFC and NFC Payments : A Review*.

Szott, S., Kosek-Szott, K., Gawlowicz, P., Gomez, J. T., Bellalta, B., Zubow, A., & Dressler, F. (2022). Wi-Fi Meets ML: A Survey on Improving IEEE 802.11 Performance with Machine Learning. *IEEE Communications Surveys and Tutorials*, *24*(3), 1843–1893. https://doi.org/10.1109/COMST.2022.3179242

The Star. (2020, March 27). Covid-19: Will virus outbreak bump web traffic into the slow lane? *STAR MEDIA GROUP*. https://www.thestar.com.my/tech/tech-news/2020/03/27/covid-19-will-virus-outbreak-bump-web-traffic-into-the-slow-lane

Tricco, A. C., Lillie, E., Zarin, W., O'Brien, K. K., Colquhoun, H., Levac, D., Moher,

D., Peters, M. D. J., Horsley, T., Weeks, L., Hempel, S., Akl, E. A., Chang, C., McGowan, J., Stewart, L., Hartling, L., Aldcroft, A., Wilson, M. G., Garritty, C., Lewin, S., Godfrey, C. M., MacDonald, M. T., Langlois, E. V., Soares-Weiser, K., Moriarty, J., Clifford, T., Tunçalp, Ö., & Straus, S. E. (2018). PRISMA extension for scoping reviews (PRISMA-ScR): Checklist and explanation. *Annals of Internal Medicine*, *169*(7), 467–473. https://doi.org/10.7326/M18-0850

Tripathi, A., Mekathoti, V. K., Waghulde, I., Patel, K., & Balasubramanian, N. (2021). Neural network aided optimal routing with node classification for adhoc wireless network. *CEUR Workshop Proceedings*, *2889*, 43–54.

Vidakis, K., Mavrogiorgou, A., Kiourtis, A., & Kyriazis, D. (2020). A Comparative Study of Short-Range Wireless Communication Technologies for Health Information Exchange. *2nd International Conference on Electrical, Communication and Computer Engineering, ICECCE 2020*. https://doi.org/10.1109/ICECCE49384.2020.9179478

Wan, H., Wang, S., Du, J., Li, L., & Zhang, J. (2023). Trace analysis using Wi-Fi probe positioning and virtual reality for commercial building complex design. *Automation in Construction*, *153*(November 2022), 1–23. https://doi.org/10.1016/j.autcon.2023.104940

Wang, Y., Tang, J., Jin, Q., & Ma, J. (2016). BWMesh: A multi-hop connectivity framework on android for proximity service. *Proceedings - 2015 IEEE 12th International Conference on Ubiquitous Intelligence and Computing*, 278–283. https://doi.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.60

Wang, Y., Tang, J., Jin, Q., & Ma, J. (2016). BWMesh: A multi-hop connectivity framework on android for proximity service. In N. H. Y. L. T. Ma J. Li A. (Ed.), *Proceedings - 2015 IEEE 12th International Conference on Ubiquitous Intelligence and Computing, 2015 IEEE 12th International Conference on Advanced and Trusted Computing, 2015 IEEE 15th International Conference on Scalable Computing and Communications, 20* (pp. 278–283). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.60

WHO, HPA, & UNISDR. (2011). Disaster Risk Management for Health OVERVIEW. *Disaster Risk Management for Health Fact Sheet*, *May*, 2–3. http://www.who.int/hac/techguidance/preparedness/risk_management_overview_17may2013.pdf?ua=1

Wi-Fi Alliance. (2014). *Wi-Fi Peer-to-Peer Services Print ( P2Ps-Print ) Technical Specification ( for Wi-Fi Direct ® services certification )*.

Wi-Fi Alliance. (2016). *Wi-Fi Peer-to-Peer (P2P) Technical Specification Version 1.7, Wi-Fi Alliance, P2P Technical Group*. https://wi-fi.org/discover-wi-fi/wi-fi-direct

Wi-Fi Alliance. (2023). *Wi-Fi Alliance certified products*. https://www.wi-fi.org/product-finder-results?sort_by=certified&sort_order=desc

World Health Organization. (2020). *Rolling updates on coronavirus disease (COVIC-19)*. World Health Organization Newsletter. https://www.who.int/emergencies/diseases/novel-coronavirus-2019/events-as-they-happen

Wu, Y., & Hu, J. (2022). Towards Independent On-device Artificial Intelligence. *Proceedings of IEEE Computer Society Annual Symposium on VLSI, ISVLSI*, *2022-July*(11), 351. https://doi.org/10.1109/ISVLSI54635.2022.00077

Yao, L., Bai, X., & Wang, Z. (2023). An Overview of Intelligent Algorithm-Based Routing in Delay Tolerant Networks. *2023 8th International Conference on Computer and Communication Systems (ICCCS)*, 345–350. https://doi.org/10.1109/icccs57501.2023.10150521

Yash, V., & Nainesh, N. (2017). ScienceDirect ScienceDirect AON : A Survey on Emergency Communication Systems during a Catastrophic Disaster. *Procedia Computer Science*, *115*, 838–845. https://doi.org/10.1016/j.procs.2017.09.166

Yasser, E., Salah, A., & Kamel, A. (2021). Enhancing the Quality of Service of Mobile-Based Software over Wi-Fi Direct. *2021 The 4th International Conference on Software Engineering and Information Management*, 1–9. https://doi.org/10.1145/3451471.3451472

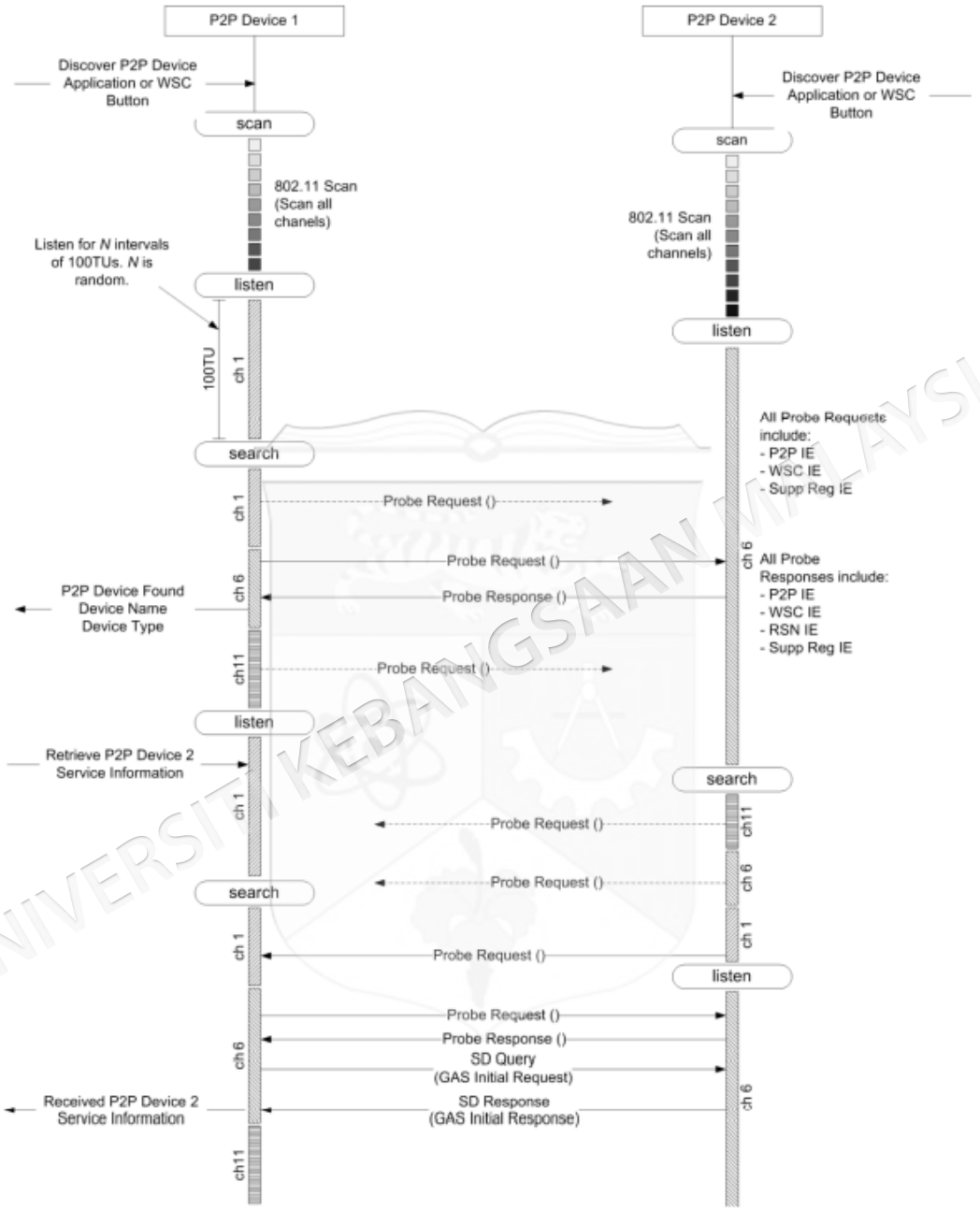
Yi, X., Pan, L., Jin, Y., Liu, F., & Chen, M. (2020). eDirect: Energy-Efficient D2D-Assisted Relaying Framework for Cellular Signaling Reduction. *IEEE/ACM Transactions on Networking*, *28*(2), 860–873. https://doi.org/10.1109/TNET.2020.2973540

Zhang, H., Wang, Y., & Tan, C. C. (2014). WD2: An Improved Wifi-Direct Group Formation Protocol. *Proceedings of the 9th ACM MobiCom Workshop on Challenged Networks*, 55–60. https://doi.org/10.1145/2645672.2645674

Zuo, J., Wang, Y., Jin, Q., & Ma, J. (2015). HYChat: A hybrid interactive chat system for mobile social networking in proximity. In W. Y. D. M. H. R. C. H. X. F. D. Y. Liu X. Wang P. (Ed.), *Proceedings - 2015 IEEE International Conference on Smart City, SmartCity 2015, Held Jointly with 8th IEEE International Conference on Social Computing and Networking, SocialCom 2015, 5th IEEE International Conference on Sustainable Computing and Communic* (pp. 471–477). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/SmartCity.2015.115

# APPENDIX A

## P2P DISCOVERY PROCEDURE




Source: (Wi-Fi Alliance, 2016)

## APPENDIX B

## GAS INITIAL REQUEST ACTION FRAMES

| | GAS Initial Request action Frame | | | | | |
|---|---|---|---|---|---|---|
| **fields** | **category** | **action** | **dialog token** | **advertisement protocol IE** | **Query Request Length** | **Query Request** |
| value | 4 (Public Action Frame - fixed) | 10 (GAS Initial Request - fixed) | token ID (set by device) | | depends on the octets in Query Request field | |
| size(octet) | 1 | 1 | 1 | 4 | 2 | variable |

| | advertisement protocol IE | | | | |
|---|---|---|---|---|---|
| **fields** | **Element ID** | **Length** | **Query Response Length Limit** | **PAME-BI** | **Advertisement Protocol ID** |
| value | 108 (Adv Protocol-fixed) | 2 (2 octets for next 3 fields fixed) | 0000000 (fixed) | 0 (fixed) | 0(ANQP-fixed) |
| size(octet) | 1 | 1 | 7bits | 1bit | 1 |

| | Query Request Field | | | |
|---|---|---|---|---|
| **fields** | **Info ID** | **length** | **OI** | **vendor specific content** |
| value | 56797(ANQP vendor specific list-fixed) | depends on next 2 fields | 0x50 6F 9A (Wi-FI Alliance OUI-fixed) | |
| size(octet) | 2 | 2 | 3 | variable |

| | vendor specific content | | TLV | | | |
|---|---|---|---|---|---|---|
| **fields** | **OUI Subtype** | **Service Update Indicator** | **length** | **service protocol type** | **service transaction ID** | **Query Data** |
| value | 9 (fixed) | counter-change when there's update | depends on next 3 fields | eg. bonjour or uPnP | depend on protocol, to patch request/response TLVs | based on service protocol type |
| size(octet) | 1 | 2 | 2 | 1 | 1 | variable |

## APPENDIX C

## GAS INITIAL RESPONSE ACTION FRAME

| | GAS Initial Response Action Frame | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **fields** | **category** | **action** | **dialog token** | **status code** | **GAS comeback delay** | **Advertisement Protocol IE** | **Query Response Length** | **Query response** |
| value | 4 (Public Action Frame - fixed) | 11 (GAS Initial Response - fixed) | copied from request frame | 0-107 | delay time value in TU | | 0 if GAS fragmentation is to be used, or the size of query response | |
| size(octet) | 1 | 1 | 1 | 2 | 2 | variable | 2 | variable |

| | | Advertisement Protocol Tuple | | | |
|---|---|---|---|---|---|
| **fields** | **Element ID** | **Length** | **Query response length limit** | **PAME-BI** | **Advertisement Protocol ID** |
| value | 108 (advertisement protocol-fixed) | 2 (next two octets-fixed) | 127 (fixed) | 0 (fixed) | 0(ANQP-fixed) |
| size(octet) | 1 | 1 | 7 bits | 1 bit | 1 |

| | ANQP Query Response Field | | | |
|---|---|---|---|---|
| **fields** | **Info ID** | **length** | **OI** | **Vendor-specific content** |
| value | 56797 (ANQP-fixed) | depends on next 2 fields | 0x50 6F 9A | |
| size(octet) | 2 | 2 | 3 | variable |

| | Vendor- specific content field | TLV | | | | | |
|---|---|---|---|---|---|---|---|
| **fields** | **OUI subtype** | **service update indicator** | **length** | **service protocol type** | **service transaction ID** | **status code** | **response data** |
| value | 9 (fixed) | counter-change when there's update | depend on next 4 fields | refer table 78 eg. bonjour | depend on protocol, to patch request/response TLVs | 0-255 | |
| size(octet) | 1 | 2 | 2 | 1 | 1 | 1 | variable |

## APPENDIX D

## CODE SNIPPET TO INITIALIZE THE UI COMPONENTS

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    mManager = (WifiP2pManager)
getSystemService(Context.WIFI_P2P_SERVICE);
    mContext = this.getApplicationContext();
    mChannel = mManager.initialize(mContext, getMainLooper(), null);

    toggleServiceButton =
(ToggleButton)findViewById(R.id.toggleServieButton);
    toggleContTest =
(ToggleButton)findViewById(R.id.toggleContTestButton);
    toggleFixTest = findViewById(R.id.toggleFixTestButton);
    send = findViewById(R.id.send_button);

    clearMsgBox = findViewById(R.id.clear_button);
    clearMsgReceive = findViewById(R.id.clear_Messages);
    clearServices = findViewById(R.id.clear_services);

    input_message = findViewById(R.id.input_edittext);
    msgSize = findViewById(R.id.testSize);
    out_message = findViewById(R.id.messages_out);
    in_message = findViewById(R.id.messages_in);

    out_scrollView = (ScrollView)findViewById(R.id.SCROLLER_OUT);
    in_scrollView = (ScrollView)findViewById(R.id.SCROLLER_IN);

    progressBar = (ProgressBar)findViewById(R.id.progressBar);
```

**APPENDIX D.1**

**CODE SNIPPET OF THE SEND BUTTON IMPLEMENTATION**

```
send.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        if(!TextUtils.isEmpty(input_message.getText().toString()))
        {
            userInput = ""+input_message.getText().toString();
            if(userInput.length()>256)
            {
                Toast.makeText(getApplicationContext(),
                        "Message length to long, please write a shorter
message",Toast.LENGTH_LONG).show();
            }
            else {
                myAgent.postMessage(userInput, senderMessage, everyone,
"ME", maxTTL);
                input_message.setText("");
                mainUpdateView.UpdateMessageView(userInput, left, out);
            }
        }
        else
        {
            Log.d(TAG, "msgbox is empty");
        }
    }
});
```

## APPENDIX D.2

## CODE SNIPPET OF THE METHODS FOR THE EXPERIMENTAL TESTING PROCEDURES

```
public void runTest(int testMsgSize, char type)
{
    testMessageSize=testMsgSize;
    testMode=true;
    testType=type;
    ackReturned=true;
    if(testType=='F')
        getNextMessage();
    testRunnable = new startTestRunnable();
    mhandler.post(testRunnable);
}
public void stopTest(char type)
{
    if(type=='C'||type=='F')
    {
        testMode=false;
        testMessage="";
        testMessageSize=0;
        testNumber=0;
        mhandler.removeCallbacksAndMessages(null);
        testRunnable = null;
    }
}
private void startTest()
{
    //send message every 5sec
    if(testMode&&ackReturned){
        if(testType=='C')//continuous test, get next message and post
            this.postTestMessage(getNextMessage(), senderMessage,
                    "targetMAC");
        else if(testType=='F')//fixed test, post
          this.postTestMessage(testMessage,senderMessage,
      "targetMAC");

        testNumber++;
        ackReturned=false;//wait for ack return
        mhandler.postDelayed(testRunnable,callBackTime);//post second
message after 5 second, callBackTime = 5
    }
    else if(testMode&&!ackReturned)
    {
        mhandler.postDelayed(testRunnable,callBackTime);//post second
message after 5 second, callBackTime = 5
        pSend.showWholeMap();
    }
}
```

to be continued…

… continuation

```
private class startTestRunnable implements Runnable {
    @Override
    public void run() {
        startTest();
    }
}


//code to generate next message for testing
private String getNextMessage()
{
    int a=97;//before 'a'
    if (testMessage == "")//geneate testMessageSize of alphabet,
first time
    {
        if (testMessageSize != 0) {
            for (int i = 0; i < testMessageSize; i++) {
                testMessage = testMessage + (char) (a);
                if (a >= 122)//already z
                    a = 97;
                else
                    a++;
            }
        }
        if(testMessageSize == 0) {
            for (int i = 0; i < 10; i++) {
                testMessage = testMessage + (char) (a);
                if (a >= 122)//already z
                    a = 97;
                else
                    a++;
            }
        }
    } else {//message previously generated, continue
        for (int i = 0; i < 10; i++) {
            testMessage = testMessage + (char) (a);
            if (a >= 122)//already z
                a = 97;
            else
                a++;
        }
    }
    return testMessage;
}
```

## APPENDIX D.3

## CODE SNIPPET OF KEY METHODS FOR SENDING MESSAGES

```
public void clearService(String oriUID) {
    pSend.clearService(oriUID);
}

public void postMessage(String inMessage, int type, String tarAdd,
String ORIUID, int nextTTL)
{
    Message msg=Message.obtain();
    msg.what = type;
    msg.obj = inMessage;
    pSend.postMessage(msg,tarAdd, ORIUID, nextTTL);
}

private void postAck(String uid, int msgType, String target)
{
    Message msg=Message.obtain();
    msg.what = msgType;
    msg.obj = uid;
    pAck.postAckMessage(target, msg);
}

public void clearAllServices()
{
    pSend.clearAllService();
    pAck.clearAllService();
}

@RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN_MR2)
public void cleanup(){
    pSend.clearAllService();
    pDiscovery.clearALLMAP();
    pDiscovery.quit();
    pSend.quit();
    pAck.quit();
}
```

## APPENDIX D.4

## CODE SNIPPET OF KEY METHODS IN P2PSENDMSGTHREAD

a.) Code snippet for the postMessage method in P2PSendMsgThread to prepare the message for delivery.

```
public void postMessage(Message msg, String targetAddress, String
ORIUID, int lastTTL) {
    switch (msg.what) {
        case senderMessage:case broadCastMessage:{
            mhandler.post(new
P2PSendMsgThread.postServiceRunnable(msg, targetAddress, ORIUID,
lastTTL));
        }
    }
}
```

b.) Code snippet for the message encoding and procedure of creating a service info instance.

```
Map record = new HashMap();
record.put("oriUID",oriUID);
record.put("uid",uid);
record.put("type", String.valueOf(type));
record.put("message",sendMessage);
record.put("TTL",String.valueOf(this.TTL));
record.put("target",targetAddress);
```

c.) Code snippet for registering the created service instance with the Wi-Fi P2P framework.

```
mManager.addLocalService(mChannel, serviceInfo, new
WifiP2pManager.ActionListener() {
    public void onSuccess() {    }
    public void onFailure(int reason) {     }
});
```

d.) Code snippet for removing the service instance from the Wi-Fi P2P framework.

```
mManager.removeLocalService(mChannel,
serviceInfoMap.get(serviceInfokey), new
WifiP2pManager.ActionListener() {
    public void onSuccess() {    }
    public void onFailure(int i) {    }
});
```

## APPENDIX D.5

## CODE SNIPPET OF KEY METHODS IN P2PSENDACKTHREAD

a.) Code snippet for the ACK message encoding and the procedure of creating a service info instance.

```
Map record = new HashMap();
record.put("uid", ackuid);
record.put("oriUID", uid);
record.put("type", type);

ackServiceInfo = WifiP2pDnsSdServiceInfo.newInstance("ack",
"_presence._udp.local", record);
```

b.) Code snippet for registering the created service instance and removing it after a certain interval of time.

```
mManager.addLocalService(mChannel, ackServiceInfo, new
WifiP2pManager.ActionListener() {

    public void onSuccess() {
        mhandler.postDelayed(new Runnable() {
            @Override
            public void run() {
                clearService(ackuid);
            }
        }, callBack
    }

    public void onFailure(int reason) {
        // Command failed.  Check for P2P_UNSUPPORTED, ERROR, or BUSY
    }
});
```

## APPENDIX D.6

## CODE SNIPPET OF KEY METHODS IN P2PDISCOVERYTHREAD

a.) Code snippet of the methods in the Agent class that invokes startDiscovery() and stopDiscovery() of P2PDiscoveryThread.

```
public void startDiscovery() {
    pDiscovery.startDiscovery();
}

public void stopDiscovery() {
    pDiscovery.stopDiscovery();
}
```

b.) Code snippet in the discoverService() method to execute the discovery procedures.

```
private void discoverService()
{

    if(serviceRequest!=null)
    {
        mManager.discoverServices(mChannel, new
WifiP2pManager.ActionListener() {
            @Override
            public void onSuccess() {
                mhandler.postDelayed(discoveryRunnable,callBackTime);
                mainUpdateView.ShowProgressBar(false);
            }

            @Override
            public void onFailure(int code) {
                String messageCode="";
                mainUpdateView.ShowProgressBar(true);

                if (code == WifiP2pManager.P2P_UNSUPPORTED) {
                    Log.d(TAG, "P2P isn't supported on this device.");
                } else if (code == WifiP2pManager.ERROR) {
                    messageCode="Error";
                    resetSR();
                } else if (code == WifiP2pManager.BUSY) {
                    messageCode="Busy";
                    resetSR();
                } else if (code==WifiP2pManager.NO_SERVICE_REQUESTS){
                    messageCode="No Service";
                    resetSR();
                }
                mainUpdateView.ShowStatusMessage(messageCode);
            }
        });
    }
}
```

c.) Code snippet for the Runnable object to be passed to the message queue in P2PDiscoveryThread.

```
private class startDiscoveryRunnable implements Runnable {
    @RequiresApi(api = Build.VERSION_CODES.O)
    @Override
    public void run() {

        if (!listenerReady) {
            listenerReady = setupListeners();
        }

        discoverService();
    }
}
```

d.) Code snippet of the listener for DNS TXT records.

```
txtListener = (fullDomainName, record, wifiP2pDevice) -> { // lambda
style coding onDnsSdTxtRecordAvailable

    messageReceivedTime = System.currentTimeMillis();
    msg.obj = record;
    msg.what=Integer.parseInt(record.get("type"));
```

e.) Code snippet of the DNS TXT listener to handle the sender type messages decoded from a peer.

```
switch (msg.what) {
    case senderMessage:{//1

if((!messageUIDCollected.containsKey((String)record.get("uid")))&&(!m
essageUIDCollected.containsValue((String)record.get("oriUID"))))
        {
            messageUIDCollected.put((String) record.get("uid"),
(String) record.get("oriUID"));
            this.agentBridge.messageReceived(msg, wifiP2pDevice,
messageReceivedTime, false);
        }
        else
        {
            this.agentBridge.messageReceived(msg, wifiP2pDevice,
messageReceivedTime, true);
        }
        break;
    }
```

f.) Code snippet of the DNS TXT listener to handle the broadcast type messages decoded from a peer.

```
case  broadCastMessage:
{
    int indexOf = ((String)record.get("oriUID")).indexOf("@");
    String oriUIDMac =
((String)record.get("oriUID")).substring(0,indexOf);
    if(oriUIDMac.equals(deviceMac.getMacAddr()))
    {
        this.agentBridge.clearService((String)record.get("oriUID"));
    }
    else
if(!messageUIDCollected.containsValue((String)record.get("oriUID")))
    {
        messageUIDCollected.put((String) record.get("uid"), (String)
record.get("oriUID"));
        boolean done = this.agentBridge.BCmessageReceived(msg,
wifiP2pDevice, messageReceivedTime,false);
    }
    else
    {
        this.agentBridge.clearService((String)record.get("oriUID"));
        boolean done = this.agentBridge.BCmessageReceived(msg,
wifiP2pDevice, messageReceivedTime,true);
    }
    break;
}
```

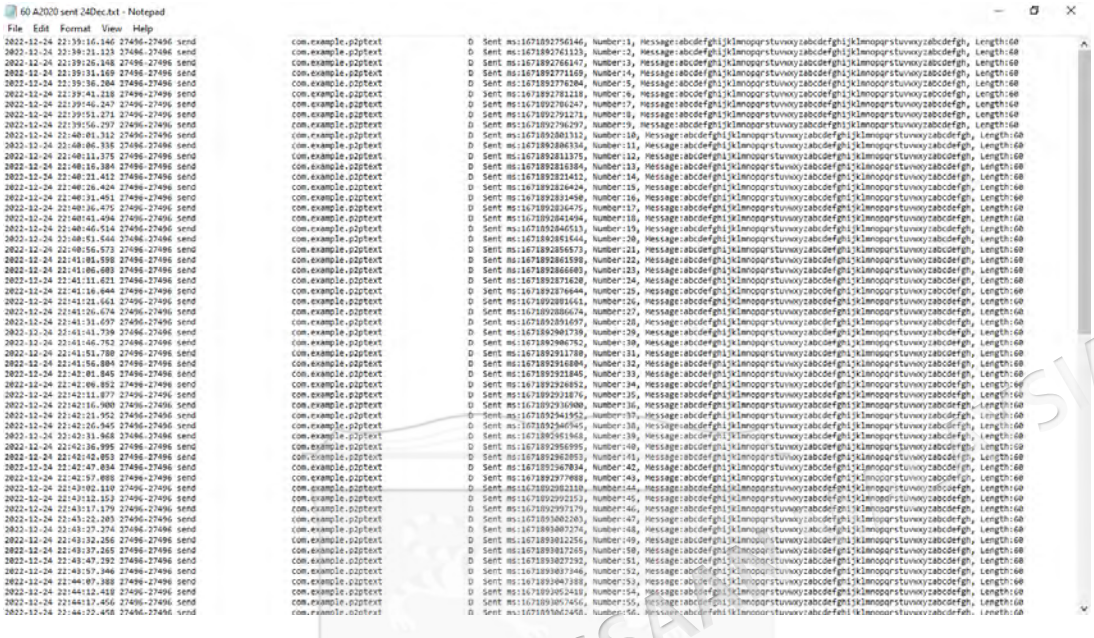
g.) Code snippet of the DNS TXT listener to handle the acknowledgement type messages decoded from a peer.

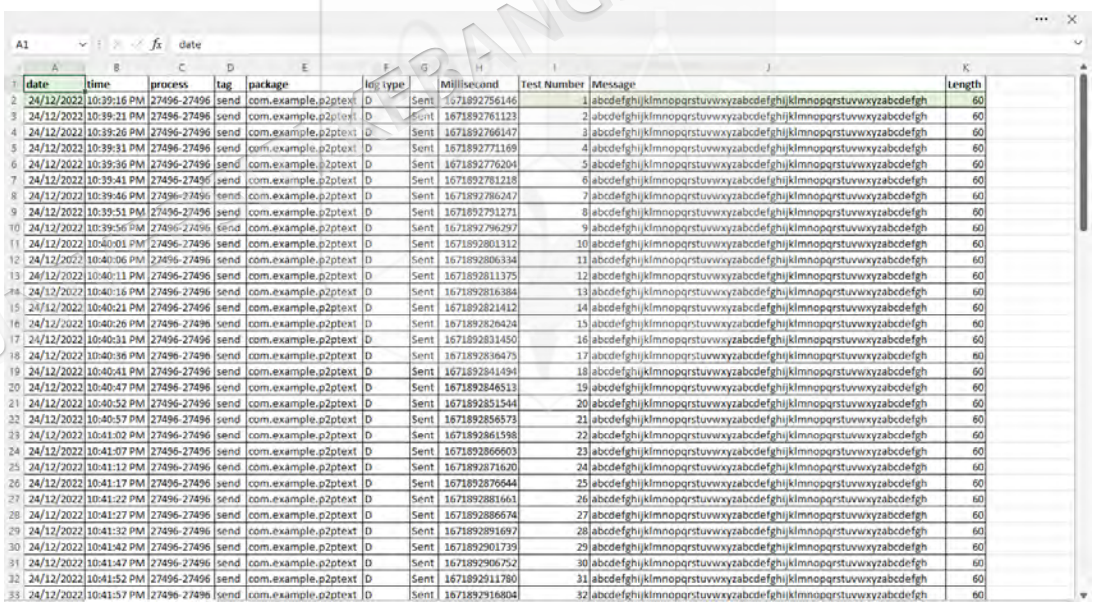
```
case ackMessage: {
    int indexOf = ((String)record.get("oriUID")).indexOf("@");
    String oriUIDMac =
((String)record.get("oriUID")).substring(0,indexOf);
    if(!oriUIDMac.equals(deviceMac.getMacAddr()))
    {
        //ackMessage to be ignored
    }
    else if(oriUIDMac.equals(deviceMac.getMacAddr()))
    {

if(!ackMessageUIDCollected.containsKey((String)record.get("uid")))
        {
            ackMessageUIDCollected.put((String) record.get("uid"),
(String) record.get("oriUID"));
            this.agentBridge.ackmessageReceived(msg, wifiP2pDevice,
messageReceivedTime);
        }
        else
        {
            Log.d(TAG, "ack processed before, ignore");
        }
    }
    break;
}
```

# APPENDIX E

## SAMPLE SENDER LOG OF THE DATA EXCHANGE TEST CASE RECORDED BY ANDROID LOGCAT, BEFORE AND AFTER BEING TABULATED

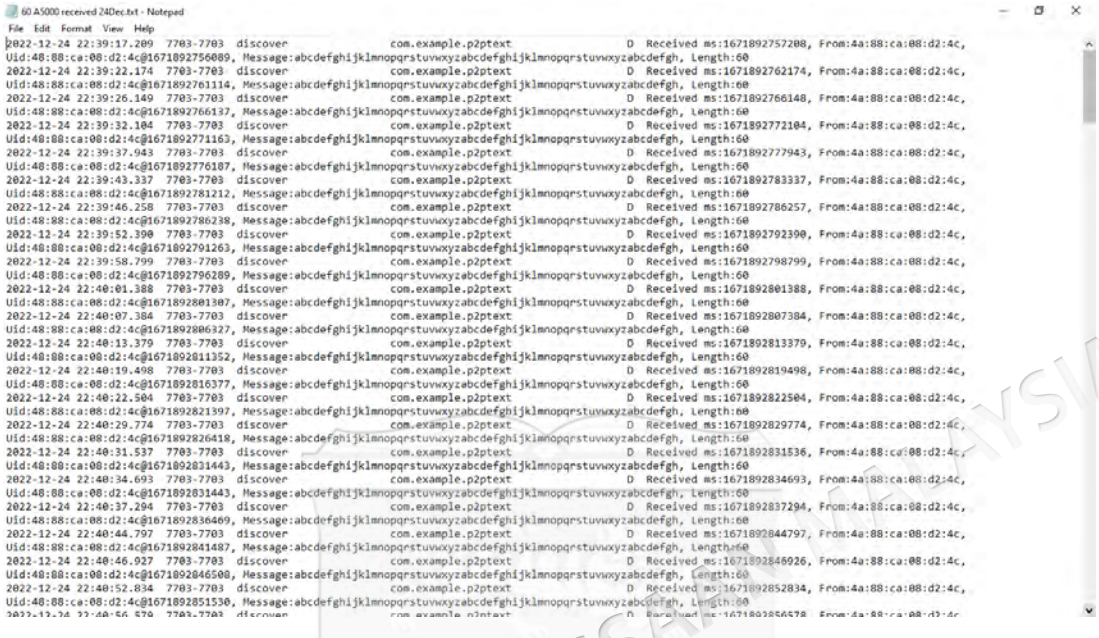


| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | date | time | process | tag | package | log type | | Millisecond | Test Number | Message | Length |
| 2 | 24/12/2022 | 10:39:16 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892756146 | 1 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 3 | 24/12/2022 | 10:39:21 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892761123 | 2 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 4 | 24/12/2022 | 10:39:26 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892766147 | 3 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 5 | 24/12/2022 | 10:39:31 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892771169 | 4 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 6 | 24/12/2022 | 10:39:36 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892776204 | 5 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 7 | 24/12/2022 | 10:39:41 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892781218 | 6 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 8 | 24/12/2022 | 10:39:46 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892786247 | 7 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 9 | 24/12/2022 | 10:39:51 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892791271 | 8 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 10 | 24/12/2022 | 10:39:56 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892796297 | 9 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 11 | 24/12/2022 | 10:40:01 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892801312 | 10 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 12 | 24/12/2022 | 10:40:06 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892806334 | 11 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 13 | 24/12/2022 | 10:40:11 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892811375 | 12 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 14 | 24/12/2022 | 10:40:16 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892816384 | 13 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 15 | 24/12/2022 | 10:40:21 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892821412 | 14 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 16 | 24/12/2022 | 10:40:26 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892826424 | 15 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 17 | 24/12/2022 | 10:40:31 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892831450 | 16 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 18 | 24/12/2022 | 10:40:36 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892836475 | 17 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 19 | 24/12/2022 | 10:40:41 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892841494 | 18 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 20 | 24/12/2022 | 10:40:47 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892846513 | 19 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 21 | 24/12/2022 | 10:40:52 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892851544 | 20 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 22 | 24/12/2022 | 10:40:57 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892856573 | 21 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 23 | 24/12/2022 | 10:41:02 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892861598 | 22 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 24 | 24/12/2022 | 10:41:07 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892866603 | 23 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 25 | 24/12/2022 | 10:41:12 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892871620 | 24 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 26 | 24/12/2022 | 10:41:17 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892876644 | 25 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 27 | 24/12/2022 | 10:41:22 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892881661 | 26 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 28 | 24/12/2022 | 10:41:27 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892886674 | 27 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 29 | 24/12/2022 | 10:41:32 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892891697 | 28 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 30 | 24/12/2022 | 10:41:42 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892901739 | 29 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 31 | 24/12/2022 | 10:41:47 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892906752 | 30 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 32 | 24/12/2022 | 10:41:52 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892911780 | 31 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |
| 33 | 24/12/2022 | 10:41:57 PM | 27496-27496 | send | com.example.p2ptext | D | Sent | 1671892916804 | 32 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 |

# APPENDIX F

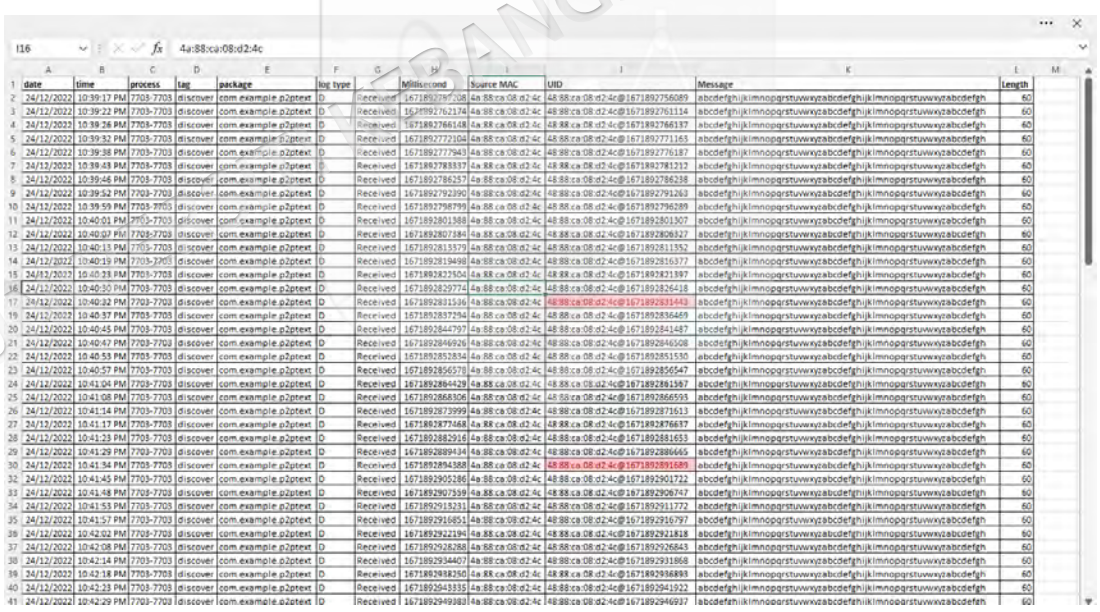
## SAMPLE RECEIVER LOG OF THE DATA EXCHANGE TEST CASE RECORDED BY ANDROID LOGCAT, BEFORE AND AFTER BEING TABULATED



60 A5000 received 24Dec.txt - Notepad

File Edit Format View Help

```
2022-12-24 22:39:17.209  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892757208, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892756089, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:22.174  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892762174, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892761114, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:26.149  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892766148, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892766137, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:32.104  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892772104, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892771163, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:37.943  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892777943, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892776107, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:43.337  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892783337, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892781212, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:46.258  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892786257, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892786238, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:52.390  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892792390, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892791263, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:39:58.799  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892798799, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892796289, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:01.388  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892801388, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892801307, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:07.384  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892807384, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892806327, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:13.379  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892813379, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892811352, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:19.498  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892819498, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892816377, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:22.504  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892822504, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892821397, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:29.774  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892829774, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892826418, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:31.537  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892831536, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892831443, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:34.693  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892834693, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892831443, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:37.294  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892837294, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892836469, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:44.797  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892844797, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892841487, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:46.927  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892846926, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892846508, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:52.834  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892852834, From:4a:88:ca:08:d2:4c,
Uid:48:88:ca:08:d2:4c@1671892851530, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2022-12-24 22:40:56.579  7703-7703  discover            com.example.p2ptext             D  Received ms:1671892856578, From:4a:88:ca:08:d2:4c,
```



I16 4a:88:ca:08:d2:4c

| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | date | time | process | tag | package | log type | | Millisecond | Source MAC | UID | Message | Length | |
| 2 | 24/12/2022 | 10:39:17 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892757208 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892756089 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 3 | 24/12/2022 | 10:39:22 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892762174 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892761114 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 4 | 24/12/2022 | 10:39:26 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892766148 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892766137 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 5 | 24/12/2022 | 10:39:32 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892772104 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892771163 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 6 | 24/12/2022 | 10:39:38 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892777943 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892776107 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 7 | 24/12/2022 | 10:39:43 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892783337 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892781212 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 8 | 24/12/2022 | 10:39:46 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892786257 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892786238 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 9 | 24/12/2022 | 10:39:52 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892792390 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892791263 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 10 | 24/12/2022 | 10:39:59 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892798799 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892796289 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 11 | 24/12/2022 | 10:40:01 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892801388 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892801307 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 12 | 24/12/2022 | 10:40:07 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892807384 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892806327 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 13 | 24/12/2022 | 10:40:13 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892813379 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892811352 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 14 | 24/12/2022 | 10:40:19 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892819498 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892816377 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 15 | 24/12/2022 | 10:40:23 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892822504 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892821397 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 16 | 24/12/2022 | 10:40:30 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892829774 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892826418 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 17 | 24/12/2022 | 10:40:32 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892831536 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892831443 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 18 | 24/12/2022 | 10:40:37 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892837294 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892836469 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 19 | 24/12/2022 | 10:40:45 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892844797 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892841487 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 20 | 24/12/2022 | 10:40:47 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892846926 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892846508 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 21 | 24/12/2022 | 10:40:53 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892852834 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892851530 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 22 | 24/12/2022 | 10:40:57 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892856578 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892856547 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 23 | 24/12/2022 | 10:41:04 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892864429 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892861587 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 24 | 24/12/2022 | 10:41:08 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892868306 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892866595 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 25 | 24/12/2022 | 10:41:14 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892873999 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892871613 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 26 | 24/12/2022 | 10:41:17 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892877468 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892876617 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 27 | 24/12/2022 | 10:41:23 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892882916 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892881653 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 28 | 24/12/2022 | 10:41:29 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892889434 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892886665 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 29 | 24/12/2022 | 10:41:34 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892894388 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892891689 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 30 | 24/12/2022 | 10:41:43 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892905186 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892901722 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 31 | 24/12/2022 | 10:41:48 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892907339 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892906747 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 32 | 24/12/2022 | 10:41:53 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892915231 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892911772 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 33 | 24/12/2022 | 10:41:57 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892916851 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892916797 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 34 | 24/12/2022 | 10:42:02 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892922194 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892921818 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 35 | 24/12/2022 | 10:42:08 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892928288 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892926843 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 36 | 24/12/2022 | 10:42:14 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892934407 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892931868 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 37 | 24/12/2022 | 10:42:18 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892938250 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892936893 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 38 | 24/12/2022 | 10:42:23 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892943335 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892941922 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |
| 39 | 24/12/2022 | 10:42:29 PM | 7703-7703 | discover | com.example.p2ptext | D | Received | 1671892949383 | 4a:88:ca:08:d2:4c | 48:88:ca:08:d2:4c@1671892946937 | abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh | 60 | |

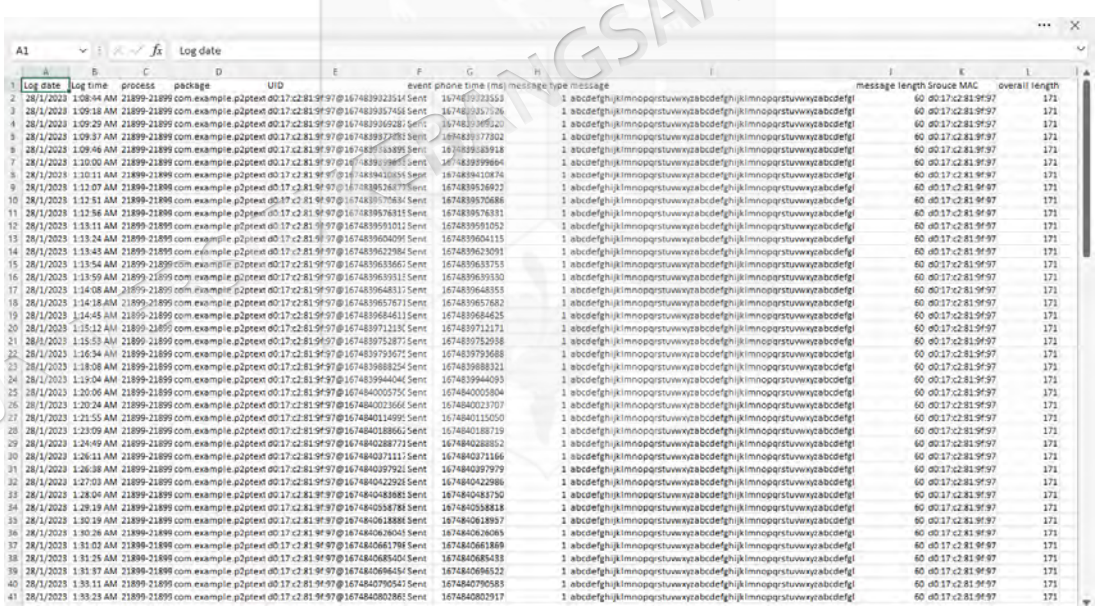
# APPENDIX G

## SAMPLE SENDER LOG OF THE FORWARDING TEST CASE RECORDED BY ANDROID LOGCAT, BEFORE AND AFTER BEING TABULATED

60Asus send.txt - Notepad

File Edit Format View Help

```
28/1/2023       1:08:44 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839323514 Sent    1674839323553   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:09:18 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839357458 Sent    1674839357526   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:09:29 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839369287 Sent    1674839369320   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:09:37 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839377283 Sent    1674839377302   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:09:46 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839385899 Sent    1674839385918   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:10:00 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839399633 Sent    1674839399664   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:10:11 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839410859 Sent    1674839410874   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:12:07 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839526877 Sent    1674839526922   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:12:51 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839570634 Sent    1674839570686   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:12:56 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839576315 Sent    1674839576331   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:13:11 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839591012 Sent    1674839591052   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:13:24 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839604099 Sent    1674839604115   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:13:43 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839622984 Sent    1674839623091   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:13:54 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839633667 Sent    1674839633753   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:13:59 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839639313 Sent    1674839639330   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:14:08 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839648317 Sent    1674839648353   1
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh    60      d0:17:c2:81:9f:97       171
28/1/2023       1:14:18 AM      21899-21899     com.example.p2ptext     d0:17:c2:81:9f:97@1674839657671 Sent    1674839657682   1
```



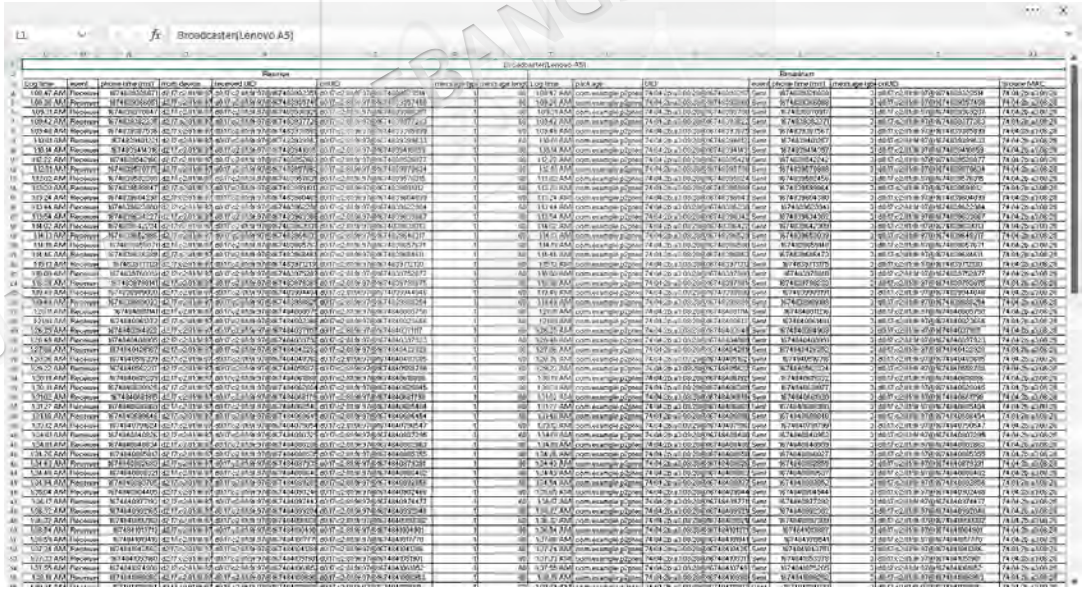
# APPENDIX H

## SAMPLE FORWARDER LOG OF THE FORWARDING TEST CASE RECORDED BY ANDROID LOGCAT, BEFORE AND AFTER BEING TABULATED

60 lenovo5 BC.txt - Notepad

File Edit Format View Help

```
"2023-01-28 01:08:46.671 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839326671,
from:d2:17:c2:81:9f:97, Uid=oriUID:d0:17:c2:81:9f:97@1674839323514, oriUID:d0:17:c2:81:9f:97@1674839323514,
TTL:3, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60"                    "2023-
01-28 01:08:46.671 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839326671,
from:d2:17:c2:81:9f:97, Uid=oriUID:d0:17:c2:81:9f:97@1674839323514, oriUID:d0:17:c2:81:9f:97@1674839323514,
TTL:3, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60" "2023-01-28
01:08:46.671 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839326671, from:d2:17:c2:81:9f:97,
Uid=oriUID:d0:17:c2:81:9f:97@1674839323514, oriUID:d0:17:c2:81:9f:97@1674839323514, TTL:3,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60"        "2023-01-28
01:08:46.839 send         com...le.p2ptext  D  74:04:2b:a3:08:28@1674839326710, Sent Time:1674839326838,
Number:2, Message:outgoing message at 2023.01.28 AD at 01:08:46 MYT, type: 3,
message:d0:17:c2:81:9f:97@1674839323514, length: 31, from: 74:04:2b:a3:08:28TTL:2
                          , Length:142"
"2023-01-28 01:08:46.839 send         com...le.p2ptext  D  74:04:2b:a3:08:28@1674839326710, Sent
Time:1674839326838, Number:2, Message:outgoing message at 2023.01.28 AD at 01:08:46 MYT, type: 3,
message:d0:17:c2:81:9f:97@1674839323514, length: 31, from: 74:04:2b:a3:08:28TTL:2"                        "2023-
01-28 01:08:46.839 send         com...le.p2ptext  D  74:04:2b:a3:08:28@1674839326710, Sent Time:1674839326838,
Number:2, Message:outgoing message at 2023.01.28 AD at 01:08:46 MYT, type: 3,
message:d0:17:c2:81:9f:97@1674839323514, length: 31, from: 74:04:2b:a3:08:28TTL:2
                          , Length:142"
"2023-01-28 01:09:26.051 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839366051,
from:d2:17:c2:81:9f:97, Uid=oriUID:d0:17:c2:81:9f:97@1674839357458, oriUID:d0:17:c2:81:9f:97@1674839357458,
TTL:3, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60"                    "2023-
01-28 01:09:26.051 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839366051,
from:d2:17:c2:81:9f:97, Uid=oriUID:d0:17:c2:81:9f:97@1674839357458, oriUID:d0:17:c2:81:9f:97@1674839357458,
TTL:3, Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60" "2023-01-28
01:09:26.051 discover     com...le.p2ptext  D  SenderMessage Received ms:1674839366051, from:d2:17:c2:81:9f:97,
Uid=oriUID:d0:17:c2:81:9f:97@1674839357458, oriUID:d0:17:c2:81:9f:97@1674839357458, TTL:3,
```



# APPENDIX I

## SAMPLE RECEIVER LOG OF THE FORWARDING TEST CASE RECORDED BY ANDROID LOGCAT, BEFORE AND AFTER BEING TABULATED

60 lenovo2 Discover.txt - Notepad

File Edit Format View Help

```
2023-01-28 01:08:48.585 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839328585, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839326710, oriUID:d0:17:c2:81:9f:97@1674839323514, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:09:27.383 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839367383, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839366080, oriUID:d0:17:c2:81:9f:97@1674839357458, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:09:33.774 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839373774, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839370887, oriUID:d0:17:c2:81:9f:97@1674839369287, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:09:42.474 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839382474, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839382237, oriUID:d0:17:c2:81:9f:97@1674839377283, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:09:48.504 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839388504, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839387541, oriUID:d0:17:c2:81:9f:97@1674839385899, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:10:02.579 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839402579, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839401225, oriUID:d0:17:c2:81:9f:97@1674839399633, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:10:16.017 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839416017, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839414317, oriUID:d0:17:c2:81:9f:97@1674839410859, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:12:25.893 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839545893, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839542167, oriUID:d0:17:c2:81:9f:97@1674839526877, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:12:53.465 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839573465, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839570818, oriUID:d0:17:c2:81:9f:97@1674839570634, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:13:03.776 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839583776, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839582404, oriUID:d0:17:c2:81:9f:97@1674839576315, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
2023-01-28 01:13:20.826 discover     com...le.p2ptext  D  BroadcastMessage Received ms:1674839600826, from:76:04:2b:a3:08:28,
Uid:74:04:2b:a3:08:28@1674839599908, oriUID:d0:17:c2:81:9f:97@1674839591012, TTL:2,
Message:abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefgh, Length:60
```