Master's Thesis

Wi-Fi Direct Services

Jimmy Tieu
Sihan Ye

Department of Electrical and Information Technology,
Faculty of Engineering, LTH, Lund University, June 2014.

Wi-Fi Direct Services

Design and Investigation of an Emerging Wireless Technology

Jimmy Tieu
ada09jti@student.lu.se

Sihan Ye
ael09sye@student.lu.se

Department of Electrical and Information Technology
Lund University

Advisor: Maria Kihl, EIT

June 18, 2014

Printed in Sweden
E-huset, Lund, 2014

Abstract

Over the last two decades, the use of wireless technologies has been extensively increased and has revolutionized the way electronic devices interact and communicate. Wi-Fi Direct is considered one of very few wireless technologies that are robust enough to offer the combination of high throughput and wide network range. As it is a peer-to-peer solution, it not only diverges from the traditional Wi-Fi infrastructure we know but also has distinct dissimilarities with Ad hoc peer-to-peer networks. Despite the high number of devices supporting Wi-Fi Direct, the use of the technology has been limited due to lack of interoperable services supporting the technology. As a solution to this problem, Wi-Fi Alliance has been developing the Wi-Fi Direct Services (WFDS) standard with clearly defined service guidelines to guarantee interoperability across different brand devices.

This master's thesis will present an investigation of WFDS and the underlying technology. A prototype which runs on Sony Mobile Communications' code base was developed to demonstrate the features of WFDS and to build a platform on which any service can be extended. The prototype was later used as a tool in the performance evaluation to analyze the advantages and limitations of Wi-Fi Direct. Comparisons were made with similar existing technologies and solutions, and the industrial relevance of WFDS was examined.

Through our research we concluded that the launch of WFDS will greatly enhance the use of Wi-Fi Direct. In certain field areas it will be able to replace Bluetooth as a more robust alternative, and can advantegously be combined with NFC to increase the user experience of services.

Acknowledgements

We would like to express our gratitude to our supervisor Kristoffer Karlsson at Sony Mobile Communications AB for his guidance and helpful advices throughout this journey. We would also like to thank our advisor Maria Kihl at EIT for her valuable input and feedback on this project. We are also grateful to Carl Gustavsson and Hares Mawlayi for showing interest in us and giving us the opportunity to carry out this thesis project at Sony Mobile Communications AB. Finally, we would like to express our sincerest gratitude to our families and friends for their love, support and encouragement.

Table of Contents

List of Figures

List of Tables

List Of Abbreviations

ADB Android Debug Bridge
ASP Application Service Platform
AP Access Point
API Application Programming Interface
BLE Bluetooth Low Energy
DLNA Digital Living Network Alliance
GO Group Owner
HTTP Hypertext Transfer Protocol
IPP Internet Printing Protocol
IEEE Institute of Electrical and Electronics Engineering
MIME Multipurpose Internet Mail Extensions
P2P Peer-to-peer
SIG (Bluetooth) Special Interest Group
Soft AP Software enabled Access Point
SoMC Sony Mobile Communications
QoS Quality of Service
STA Station
TCP Transmission Control Protocol
UDP User Datagram Protocol
UI User Interface
UPnP Universal Plug and Play™
WFDS Wi-Fi Direct Service
WLAN Wireless Local Area Network
WPS Wi-Fi Protected Setup

Chapter 1
Introduction

1.1 Background

Traditional Wi-Fi is a wireless technology based on IEEE 802.11 standards, developed for the purpose of allowing multiple network client devices to connect with other network devices using a Wi-Fi Access Point (AP). All communication between client-client is handled through the AP.

Wi-Fi Direct, also known as Wi-Fi peer-to-peer (P2P), is a technology which enables Wi-Fi devices to connect directly without the need to go through a physical AP. Devices supporting Wi-Fi Direct are not even required to belong to the same Wi-Fi network or have access to the Internet, the devices simply establish a P2P Group in order to communicate peer-to-peer, shown in figure 1.1. A device that belongs to a P2P Group is referred to as P2P Group Owner (GO) or P2P Client.

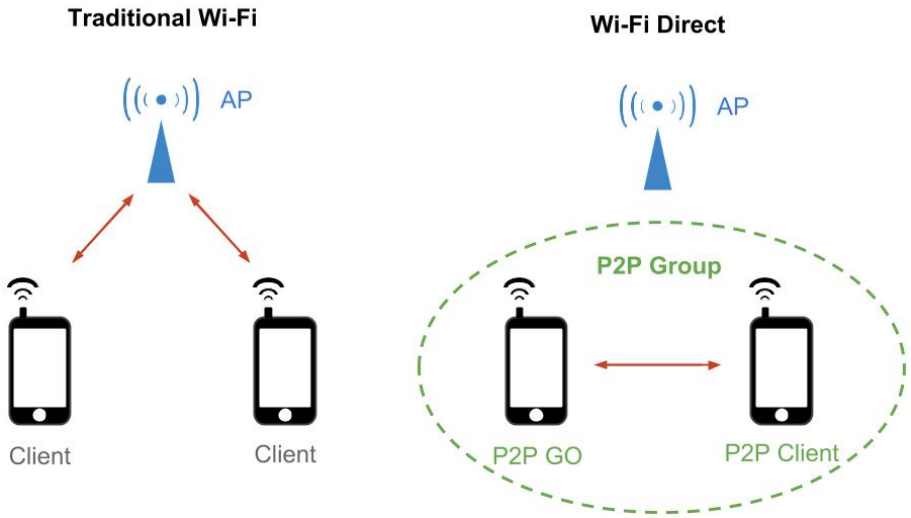

Figure 1.1: Traditional Wi-Fi network compared with Wi-Fi Direct.

Wi-Fi Direct Services (WFDS) is a new standard specification defined by Wi-Fi Alliance, built on the Wi-Fi Direct technology. There are four major standardized services defined by Wi-Fi Alliance; Send, Play, Display and Print. WFDS enables devices to advertise and seek for a specific service and retrieve service information from a remote device prior to establishing a connection.

The services are supported by an underlying logical entity Application Service Platform (ASP), independent from applications running on top of it. In addition to the defined services, the ASP also offers the same support for service which are not defined by Wi-Fi Alliance. A WFDS framework component named Enable Interface allows third party applications to interact with the ASP and act as an ASP service [1].

1.2 Project Scope

Despite the high number of devices supporting Wi-Fi Direct since its launch four years ago, the use of this technology has never really taken off. The major reason is the lack of standardized services on top of Wi-Fi Direct, which results in simple use cases such as transferring a file completely dependent on the application managing the transfer. Existing implementations of these type of services today are often brand specific with very limited interoperability.

The WFDS standard was developed as a solution to this problem, offering clearly defined services. This makes interoperability possible between different brand devices running different applications, as long as they support the WFDS standard. The certification of WFDS hasn't officially been launched yet (planned in mid 2014), and previous research within this area is therefore very limited.

Sony Mobile Communications (SoMC) believes that WFDS will become a very useful technology in mobile phones, but there is no existing support for WFDS in SoMC's current code base. The goal of this thesis project is to investigate in Wi-Fi Direct and WFDS from a mobile device perspective.

1.2.1 Method

A large part of this project is dedicated for development of a WFDS prototype for Android devices, covering the following components:

- Application Service Platform including the ASP Coordination Protocol.
- Send Service with support for both "Transmitter" and "Receiver" role.
- Application for presenting and testing the functionalities of underlying building blocks.

The purpose of the prototype is to demonstrate the functionalities of WFDS and build a platform on which any service can be extended. The only reference used are the specifications from Wi-Fi Alliance, the implementation is independent from any other reference source code. The prototype will also be used as a tool for evaluation and performance analysis of WFDS later in this thesis, where the following problem statements will be addressed:

- What are the characteristics of Wi-Fi Direct, and how does it differ from traditional Wi-Fi in WLAN and Ad hoc networks?
- How does WFDS compare with similar technologies and solutions that achieve the same tasks?
- When is it beneficial to use WFDS and how can SoMC apply WFDS use cases to their mobile devices? When is it beneficial to use WFDS and how can it be applied to SoMC mobile devices and other Sony products? When is it beneficial to use WFDS and how can Sony apply WFDS use cases to their mobile deivces and other products?

Due to limitations in battery capacity in mobile devices, efficiency in terms of power consumption and throughput will be of particular interest during the investigation and comparison of respective wireless technology.

1.2.2 Limitation

In order to avoid making changes in SoMC's current code base and risk affecting other parts of their code, implementation was done on existing APIs to avoid making changes in the existing Frameworks Base. Some minor details in the specification were overlooked in the implementation because they were not possible to implement without changing the Frameworks Base. Since only the Send Service was decided to be implemented for demonstration of the ASP functionality (due to time limitation), later discussions about WFDS will somewhat base on the Send functionality.

The investigation scope of other wireless technologies will mainly focus on those commonly used in mobile devices. Wireless technologies that are not intended for the mobile market will not be mentioned in this thesis.

1.3 Thesis Outline

Chapter 2 presents relevant work done by Wi-Fi Alliance, Bluetooth Special Interest Group, and Near Field Communication Forum within the field of wireless technologies. The chapter contains a brief description of the different technologies' characteristics, and where Wi-Fi Direct positions itself.

Chapter 3 introduces the fundamentals of Wi-Fi Direct and WFDS. Chapter 3.1-3.3 presents the functionalities of Wi-Fi Direct and the connection setup process. Chapter 3.4-3.7 describes the additional framework components defined in WFDS.

Chapter 4 describes the design and architecture of the prototype and the environment for the development. This chapter presents a detailed description of the implementation and design choices made, together with a description of the Send Service characteristics.

Chapter 5 presents the tools and environment used in the throughput, discovery time and power consumption tests. The results from the performance evaluations are presented and analyzed.

Chapter 6 presents an analysis of the differences between Wi-Fi Direct and other wireless technologies as well as comparing WFDS with existing solutions with similar funtionality. A proposition for how WFDS can be integrated into Sony products is presented.

Chapter 7 summarizes the key elements from previous chapters and presents our conclusions.

Chapter 2
Related Work

Over the last two decades, the growing availability of wireless communication have steadily been increasing. With the constantly shrinking size of electronic components supporting these technologies, the chipsets used today are so small that they have promoted completely new and revolutionary ways of wireless interaction between devices. In our modern world we no longer only *want* our electronic devices to be able to connect without any physical attachment, we simply *expect* them to offer this without compromises.

There are a number of different wireless technologies available on the market today, with ranges stretching from a few centimeters up to hundreds of meters. Each of them offer an unique set throughput, range, security, and cost etc., which make them suitable for very different fields of application.

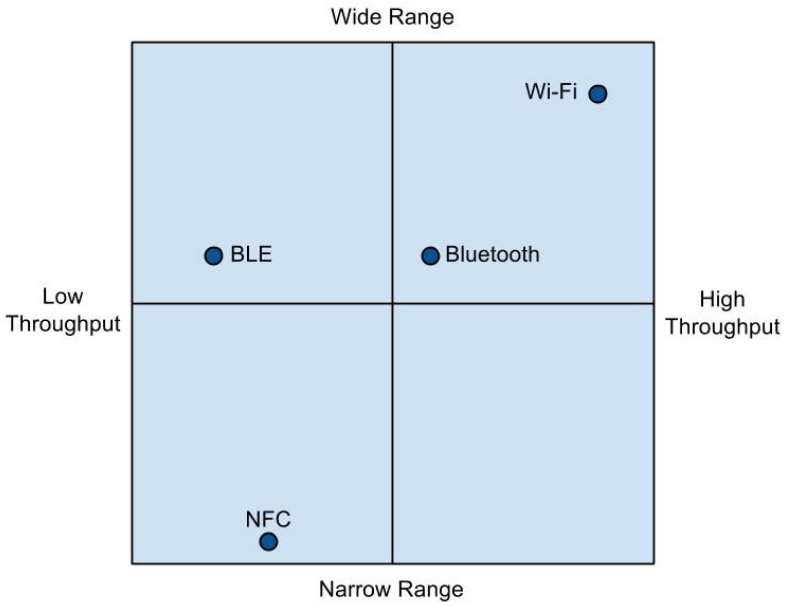

Figure 2.1: Wireless technologies used in mobile devices.

When it comes to mobile devices, there are three wireless technologies that are more frequently used than others, shown in figure 2.1. Wireless Fidelity (Wi-Fi), Bluetooth, and Near Field Communication (NFC) are today widely supported across nearly all mobile devices. These technologies are therefore highly relevant to have in mind when investigating in WFDS.

2.1 IEEE 802.11 WLAN

Wi-Fi Alliance, formerly known as WECA, is a global non-profit organization that owns the Wi-Fi trademark, founded in 1999. The goal of their work is to attain a worldwide standard of Wi-Fi technology and interoperability through certifications and guidelines. Since their establishment they have certified over 18 000 products. Wi-Fi Alliance defines Wi-Fi as any Wireless Local Area Network (WLAN) device that implements the 802.11 standards by the Institute of Electrical and Electronics Engineers' (IEEE) [2]. In recent years, Wi-Fi Alliance has been extending Wi-Fi beyond the WLAN applications it was originally intended for toward peer-to-peer solutions.

A WLAN is a wireless network of devices that communicates with Radio Frequency signals. Almost all modern WLANs devices based on the IEEE 802.11 standard are Wi-Fi certified, which is the reason why the term Wi-Fi has become synonymous with IEEE 802.11 WLAN. All devices within a WLAN fall into two categories, Access Points and Stations (also known as clients in Wi-Fi):

- Station (STA)

  A station is a device that supports IEEE 802.11, used in WLAN Infrastructure and Ad hoc mode. A station can be a mobile, laptop or a desktop computer.

- Access Point (AP)

  The AP is the central device in WLAN Infrastructure mode. All stations in the star topology network are connected to the AP, which is responsible of transporting data between them. An AP can also act as a bridging device for its stations to other networks, for example when a router allows its clients' wireless access the Internet. By acting as a bridging device, the AP then needs to translate wireless IEEE 802.11 frames to wired Ethernet frames and vice versa. A typical 802.11n AP can offer a speed up to 300-400 Mbit/s within a range up to 300 m [3].

IEEE 802.11 WLAN has two basic operation modes; Infrastructure mode and Ad hoc mode [4]. However, most IEEE 802.11 WLAN networks today are using the infrastructure mode. When talking about traditional Wi-Fi, it is usually understated that it refers to the Infrastructure mode.

2.1.1 Infrastructure Mode

Infrastructure mode is a star topology network that consists of one central AP device and multiple station devices connected to it. Although 802.11 WLAN

provides a very high throughput, the power consumption is a tradeoff. The AP is required to constantly broadcast frames to announce its presence and maintain the link to each station even though there is no actual transmission. The power consumption varies depending of the throughput capability of the AP device, but a typical AP consumes 150 mA with a supply voltage 15V [5]. The AP functionality is therefore more suitable for devices constantly connected to a power source.

2.1.2 Ad Hoc Mode

Ad hoc is a decentralized network type which consists of only stations. Since there is no AP handling the network control, the control is distributed between the stations within the network. This means that the network does not rely on preexisting infrastructure around an AP. Instead, every stations holds a routing table, a list of the routes to a particular network destination. Data between two end-points are forwarded from station to station according to the routing table on each device.

A station is connected to all the reachable stations by so called "links", shown in figure 2.2. Since a link can be connected or disconnected at anytime, stations must be able to handle a dynamic restructuring of the network. To discover if a link or a station has been added or removed, the stations typically announce its presence (beaconing) and listen for announcement broadcasts from its neighbors. When the network structure is changed, the routing tables of affected stations are updated [6].

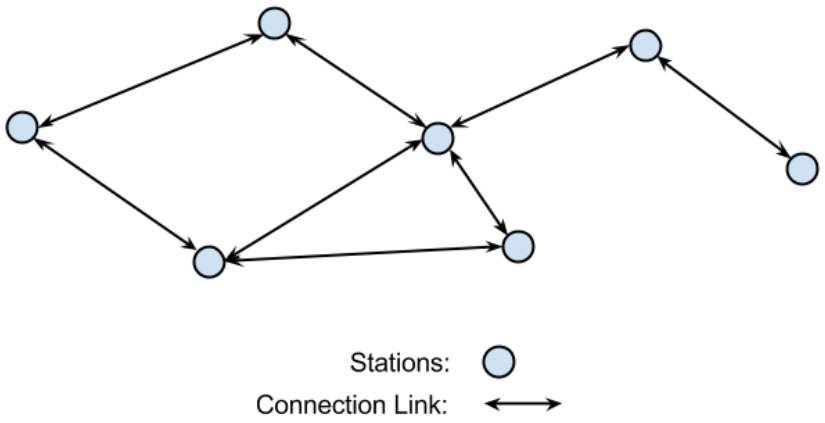

Figure 2.2: Ad hoc Network structure.

The advantage of Ad hoc mode is that it is a peer-to-peer solution that offers flexibility and mobility. Compared with the Infrastructure mode, Ad hoc is not limited to a physical point since the location of each station is not limited to the range of the AP. But the throughput level in is also lower, Wi-Fi devices in Ad hoc

mode have a maximum data transfer rate of 11 Mbit/s. The power consumption for each station in Ad hoc is also larger than stations in Infrastructure mode, consuming 156 mA in idle mode [7].

2.2 Bluetooth

Bluetooth SIG is non-profit standard organization established 1998. Bluetooth SIG owns the Bluetooth trademark, oversees the development of Bluetooth standards, and is responsible for licensing Bluetooth trademarks to companies that are implementing Bluetooth technology in their products. Today, more than 20,000 companies are members in the Bluetooth SIG [8].

Bluetooth is a short-range communication technology that offers a direct connection between devices. Bluetooth uses adaptive frequency-hopping scheme in the 2.4 GHz band, which means it hops from frequency to frequency in a pseudo-random sequence.

The network topology of Bluetooth is 1:N, where one Master device can connect to one or many Slave devices. The formed network is called a Piconet, where the device with Master role is responsible of the network control [6].

2.2.1 Bluetooth Classic & Bluetooth Low Energy

The Bluetooth technology can be divided into two categories: Bluetooth Classic and Bluetooth Low Energy. These two aren't compatible, but a device can implement them both. The main differences between these two types are their different power consumption, throughput and range. Bluetooth Classic is mainly designed for continuously streaming of data during a connection, but even when there is no data to be transmitted, the connection is still maintained. Due to this, Bluetooth Classic is not ideal for some use cases that only require episodic or periodic transfer of smaller amounts of data.

In 2010 SIG developed a new Bluetooth revision; Bluetooth Low Energy (BLE). The main advantage of BLE is the low power consumption it offers which makes it possible to power a small Bluetooth device with a tiny coin cell battery for a longer period of time. Table 2.1 shows a comparison between the range, average power consumption and throughput at baseband between Bluetooth Classic and BLE.

	Bluetooth Classic[9][10]	Bluetooth Low Energy[11] [12]
Range	$<$ 10 m (Class II)	$<$ 100 m
Power Consumption	37.7 mA	0.024 mA
Throughput	Up to 2.1 Mbit/s	Up to 1 Mbit/s

Table 2.1: Bluetooth Classic and Bluetooth Low Energy specification.

A device using BLE with a battery capacity of 230 mAh can maintain a connection time corresponding to more than a year (assuming average consumption

0.024 mA) [11]. As shown in the table above, BLE is optimal when maintaining a longer Bluetooth connection is the goal. But for a constant streaming of data, using Bluetooth Classic is still more efficient due to the higher throughput.

2.3 Near Field Communication

The NFC is a technology developed through a collaboration with Sony and Philips in late 2002. The NFC Forum is a non-profit industry association formed in 2004, with the mission is to promote and develop interoperability standards for the NFC technology [13].

NFC is a wireless technology used for short-ranged communication between devices. The transmission range is only approximately a few centimeters, and communication is immediately initiated when two NFC devices are hold next to each other, a motion called "One-Touch" or "tapping". NFC operates in the 13.56 MHz band, the same frequency as Radio-Frequency Identification (RFID), and supports a maximum throughput of 424 kbit/s [14].

There are two kinds of NFC modes, active and passive. The active ones are often found in mobile device, and are able to write and read data with a power consumption at 15 mA[14]. The passive ones, which known as NFC Tags, are unpowered and can only be used to store data. A typical integration is access cards, where the NFC tag contains information about access information. Lately, integrated NFC have become a popular feature in many consumer electronic devices and a majority of all new mobile devices now offer NFC support. In many use cases, NFC is combined with another wireless technology, for example Bluetooth. NFC Handover allows two devices to establish the Bluetooth connection simply by holding the devices together. This allows user to skip the step where devices searches the nearby area for compatible Bluetooth devices [15].

2.4 Where Wi-Fi Direct comes in

Wi-Fi Direct is a peer-to-peer solution developed by Wi-Fi Alliance that supports the IEEE 802.11 a/b/g/n/ standards [16]. The topology in a P2P Group is 1:N, where one GO can connect to multiple clients. It is similar to the WLAN topology but without the client-client communication through the GO. Wi-Fi Direct is also quite different from Ad hoc although they both are peer-to-peer solutions, because their network structures are completely different.

The Wi-Fi Direct certification program was launched in October 2010 by Wi-Fi Alliance. The purpose of the certification is to ensure that all Wi-Fi Direct devices meet the same requirements of interoperability, security and reliability. Wi-Fi Direct is backwards compatible with Wi-Fi certified devices. Since Wi-Fi Direct is basically using the same IEEE 802.11 standards as traditional Wi-Fi, it can operate at distances up to 200 m and reach data transfer speeds up to 250 Mbit/s [17].

As of today there are close to 5000 Wi-Fi Direct certified consumer electronic products, ranging from smartphones and tablets to televisions and gaming devices [18]. Wi-Fi Direct has been a mandatory feature for all new Android devices

starting with version 4.0x Ice Cream Sandwich. The first smartphone to pass certification was Samsung Galaxy S.

Chapter 3
Wi-Fi Direct Services

3.1 Wi-Fi Direct

Devices using Wi-Fi Direct, officially named "P2P devices", interacts with other devices by establishing P2P Groups. One device in a P2P Group will assume the role of P2P GO and the other P2P devices in the group are referred to as P2P Clients.

3.1.1 P2P Device

A P2P device supports the role of both P2P GO and P2P Client. The roles are dynamic and all P2P devices are required to implement both roles.

It is possible for a P2P device to operate as a client in a WLAN Infrastructure network and to have the role of GO in a P2P Group simultaneously. This mode of operation is called concurrent mode, the device alternates between roles by time-sharing the Wi-Fi interface.

P2P GO

The P2P GO role is equivalent to the function of a traditional AP in WLAN networks. The GO announces itself to other P2P devices and functions as a Software enabled Access Point (Soft AP) for the P2P Clients in its P2P Group.

Only the GO is allowed to cross-connect its P2P Clients to an external network. This type of connection is done at the OSI Network layer [19]. The GO will then act as a DHCP server which provides its P2P Clients with IP addresses.

P2P Client

The P2P Clients function like traditional Wi-Fi clients, connected to the P2P GO. A P2P Client is only intended to communicate with the GO in the P2P Group.

3.2 Device Discovery and Group Formation

There are three ways in which two devices can establish a P2P Group; Standard, Autonomous or Persistent formation. All three procedures starts with a traditional

Wi-Fi (IEEE 802.11) active scan, but differs somewhat in the later phases [20].

Standard	Autonomous	Persistent
1. Discovery	1. Discovery	1. Discovery
Scan Phase	Scan Phase	Scan Phase
Find Phase		
2. GO Negotiation		2. Invitation
3. WPS Provisioning	2. WPS Provisioning	3. WPS Provisioning
Phase 1	Phase 1	
Phase 2	Phase 2	Phase 2
4. Operational Phase	3. Operational Phase	4. Operational Phase

Table 3.1: P2P discovery and P2P Group formation procedures.

- Standard formation

  The standard formation is used when two unacquainted P2P devices establish a P2P Group. This procedure is typically performed when two devices establish a group for the first time and need to negotiate who will take on the GO role. In addition to the active scan, the discovery phase also includes a find phase where Probe Requests and Probe Responses are used to detect other devices.

- Autonomous formation

  The autonomous formation is used when a P2P device already has decided to become GO and autonomously create its own group. Other devices can discover the established group and join as clients, and thereby skip the GO Negotiation phase. The discovery phase is also simplified, as the device establishing the group does not need to go through the find phase.

- Persistent formation

  When P2P devices during the group formation uses a flag to declare the group as persistent, the devices can remember the network credentials and assigned P2P roles in the P2P Group. Next time the devices wish to connect, one of them sends an invitation request to quickly re-establish the persistent group with their original roles.

  If the devices previously were part of a persistent group but the device receiving the invitation request no longer remembers the persistent group, the invitation fails and group formation will continue as a standard formation from the GO Negotiation phase [21].

3.2.1 Device Discovery

Scan Phase

A P2P device starts the discovery phase by performing a traditional Wi-Fi scan through all supported channels to find and collect information about surrounding P2P Groups and legacy networks. By using Wi-Fi scan, the P2P device locates

the best potential channel for later establishment of a P2P Group. During this phase the P2P device shall not reply to any Probe Request [22].

Find Phase

Once the Scan phase ends the P2P device proceeds to the Find Phase. The aim is to quickly ensure that two P2P devices in Device Discovery are located in the same channel to exchange device information and determine if a connection should be attempted. This is achieved by letting the P2P devices alter between the two states Listen and Search.

Listen state is when the P2P device waits for a Probe Request on one fixed channel, and Search state is when the P2P device sends Probe Requests to a fixed set of channels. If a suitable P2P device receive a Probe Request, it could respond with a Probe Response to proceed to the next phase.

The probability for two devices to find each other depends on the number of channels and the time they spend in each of the two states, which are randomly generated within a boundary. This is optimized by choosing the set of channels recommended for Wi-Fi Direct, also known as the three Social Channels 1, 6 and 11 in the 2.4 GHz band, shown in figure 3.1 [22].

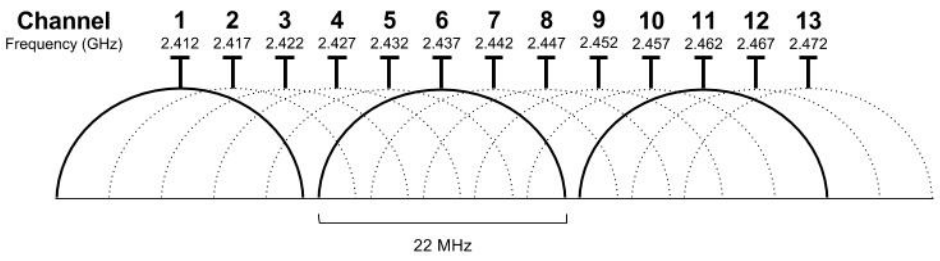

Figure 3.1: Channels in the 2.4 GHz band.

3.2.2 Group Owner Negotiation

When the device information has been exchanged the GO negotiation takes place. This is performed by a three way frame exchange between the two P2P devices (GO Negotiation/Response/Confirmation). The GO Intent value, embedded in the first frame, is sent by both P2P devices. The device with the highest value set will receive the GO role, shown in figure 3.2.

In case the GO Intent is equal on both devices, a "Tie Breaker Bit" randomly set to 0 or 1 set in the first frame determines which device will become GO[22].

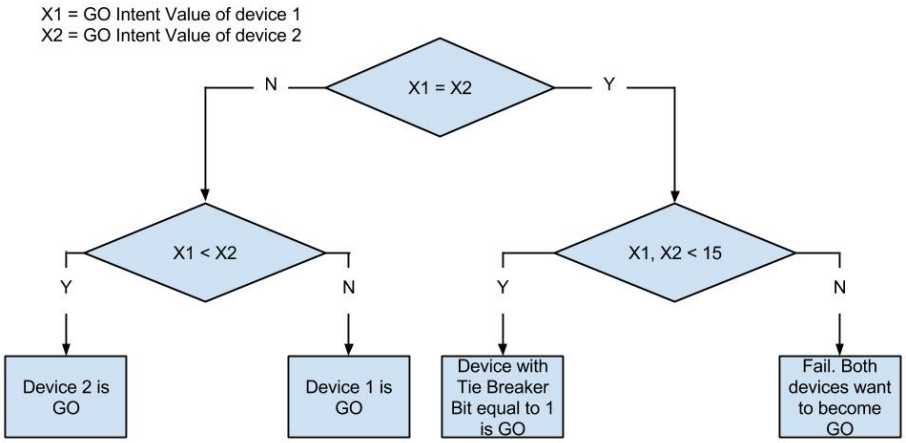

Figure 3.2: Group Owner Negotiation Flow Diagram [1].

3.2.3 WPS Provisioning

When respective roles have been negotiated the authentication part takes place. Wi-Fi Direct devices uses the secure protocol, Wi-Fi Protected Setup (WPS), to set up and support the connection. By adapting the WPS protocol the P2P Group is required to implement two parts, Internal Registrar and Enrollee (P2P GO and P2P Client respectively) [22].

Phase 1

In the first phase the Internal registrar generates and issues the network credentials to the Enrollee by exchanging frames as shown in the schematic above.

Phase 2

During the second phase the Enrollee reconnects to the Internal Registrar using its new authentication credentials.

3.2.4 P2P Invitation

The invitation is an optional procedure which is used during re-invoke of a persistent group, in which two P2P devices (one of them GO) previously have been provisioned. An invitation request frame can be sent from either the client or GO, and the persistent group is re-invoked once it receives a successful invitation response from the requested device.

However, a successful invitation procedure can only be performed if one of the devices had the GO role, and the invited device still remembers the P2P Group. If these two requirements are not fulfilled, the invitation procedure will transition into a standard group formation procedure.

Another type of use of P2P invitation is when either the P2P GO or Client in an operational group wishes to invite another P2P device to join as a client [22].

3.3 Service Discovery

Service discovery is an optional frame exchange which can be performed by a P2P device to any other discovered P2P device to find a list of all services it offers and determine compatibility information on the services. This can typically be performed following a successful Device Discovery procedure, which allows the service seeking device to collect specific information about the found service before deciding to form a P2P Group with the remote device or not [22].

The Service Discovery protocol is extensible which enables higher layer service advertisement protocols to be used, such as Bonjour and UPnP. The Service Discovery Query and Service Discovery Response uses the Generic Advertisement Service Protocol (GAS), an OSI Link Layer protocol [19].

Although the general paradigm of Wi-Fi-Direct is to establish and maintain relationships as Peer Devices, there is typically a Service Advertiser and a Service Seeker role in setting up a connection. The relationship between these roles will be described more in detail in the next section.

3.4 Wi-Fi Direct Services

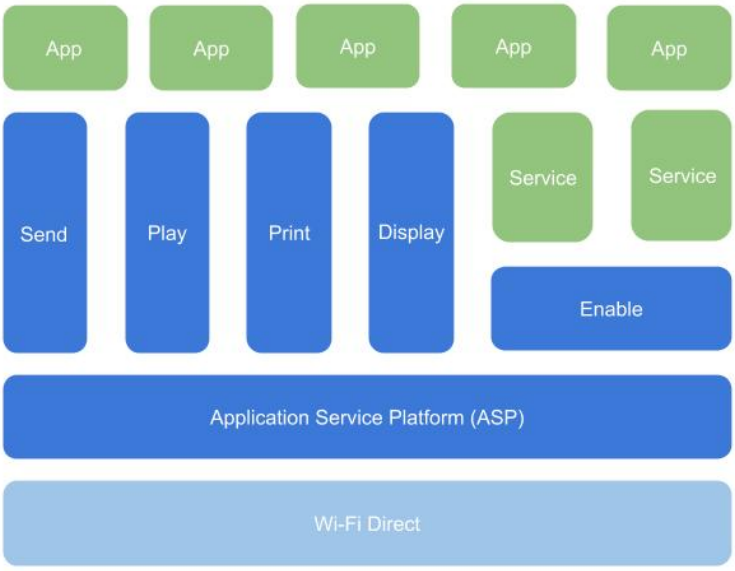

Figure 3.3: WFDS Framework Components [1].

The WFDS framework consists of components covered in blue in figure 3.3, the dark blue parts being new additions defined by WFDS. The framework includes a certified Wi-Fi Direct component, Application Service Platform, Enable APIs and Wi-Fi Alliance defined services [1].

3.4.1 Services

The services make interoperability possible across Wi-Fi Direct capable devices, allowing different brand devices running different applications to interoperate as long as they support the same WFDS. The following four services are defined by Wi-Fi Alliance.

Send Service

The Send service enables the transferring of bulk data from a Send Transmitter to a Send Receiver. The Data Plane which provides the transport path uses Hypertext Transfer Protocol (HTTP) for managing the data transfer.

The Send Transmitter utilizes the HTTP Client functionality to send files to the Send Receiver, which utilizes as a HTTP Server.

Play Service

The Play service enables a Digital Living Network Alliance (DLNA) connection, offering the user a control interface to play encoded media through other P2P devices. DLNA is an organization responsible forfwhich defining guidelines for interoperable sharing of digital media between multimedia devices.

The Play Transmitter utilizes the DLNA Push Controller functionality to stream encoded video, audio and images to a Play Receiver, which utilizes the DLNA Media Render functionality. [23]

Display Service

The Display service leverages Wi-Fi Miracast Certification with the addition of ASP, to setup and manage multiple services operating at the same time. This makes WFDS Display devices not only compatible with other WFDS Display devices, but also with Miracast Devices without ASP support (i.e Miracast Devices which do not support WFDS).

The Display Transmitter utilizes the Wi-Fi Display Source functionality and supports a user interface to display media contents on a Display Receiver. The Display Receiver utilizes the Wi-FI Display Sink functionality to displays the media content and has the ability to accept or reject an incoming connection request.

Print Service

The Print service specification is based on the Internet Printing Protocol (IPP), a standard network protocol for remote printing. IPP is implemented using HTTP as transfer protocol.

The Print Transmitter utilizes the IPP Client functionality, which discovers and starts a Print Session with the Print Receiver. The Print Receiver utilizes the IPP Server functionality, which receives data from the Transmitter and performs print jobs.

3.5 Application Service Platform

The Application Service Platform (ASP) is a logical entity which implement necessary functions needed by services. The ASP is responsible of coordinating the advertisement and discovery of services, ASP-Sessions management, connection topology management, and security.

3.5.1 Service Seeker and Service Advertiser

In order to allow other P2P devices to find the services which a P2P device offers, these services must first be advertised. Each service is advertised and represented by its service name defined by Wi-Fi Alliance. The Transmitter entity (TX) and Receiver entity (RX) in each service have different service names and are advertised separately.

A device which seeks for an advertised service on a remote device takes the role of Service Seeker. A device which advertises a service to make it known for potential Service Seekers takes the role of Service Advertiser. One device may have multiple Service Advertisers and Service Seekers roles simultaneously for different service entities.

3.5.2 ASP Session

An ASP-Session is the logical link between the Service Seeker and the Service Advertiser on separate devices. An ASP can set up multiple ASP-Sessions simultaneously, each ASP-Session is identified by a session identifier.

The Service Advertiser-Service Seeker relationship for each given ASP-Session is fixed and independent on the underlying P2P GO/Client roles and the overlying Transmitter/Receiver roles. Two communicating ASPs in an ASP-Session are termed as ASP-Peers of each other.

3.5.3 ASP Coordination Protocol

The ASP coordination protocol is used by ASP-Peers to exchange messages during the establishment and removal of an ASP-Session. ASP coordination messages utilizes User Datagram Protocol (UDP), and the messages are single datagrams (HEX byte streams).

Each command sent from an ASP-Peer contains information from its Service to the corresponding service on remote ASP-Peer. The remote ASP-Peer responds with an Acknowledgement message after each command message.

Figure 3.4 illustrates the operation sequence when two separate service-sessions are established between device A and device B.

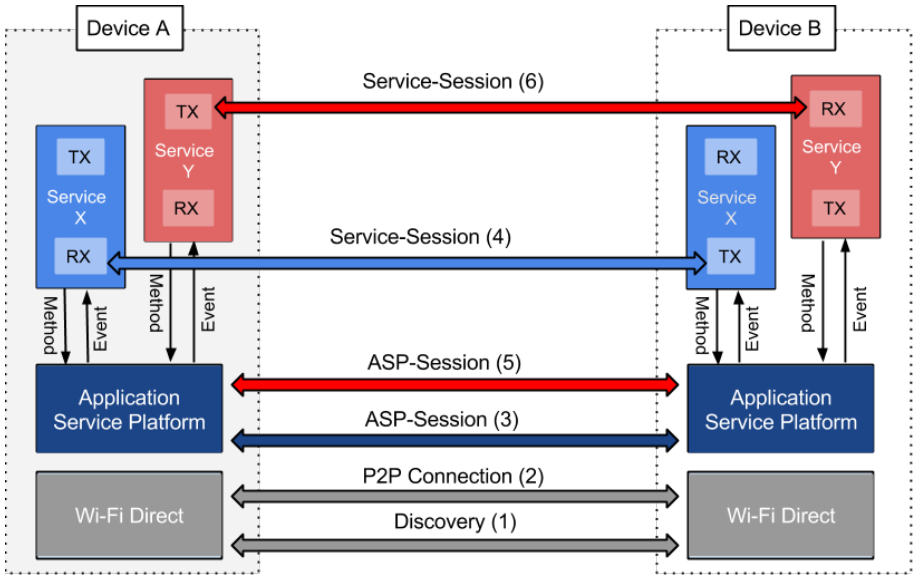

Figure 3.4: WFDS Flow of Operations.

3.6 Service Session

A service is divided into a Transmitter entity and a Receiver entity, and each P2P device can either implement one or both of these roles. However, a connection can only be made between the Service's Transmitter on one device to the corresponding Service's Receiver on another device.

A Service-Session is used for the transferring of data between two separate services. For example, when a Send Service Transmitter is connected to a Send Service Receiver, a Send-Session is established between the Transmitter's HTTP Client and the Receiver's HTTP Server.

3.7 Services and ASP Interface Primitives

The ASP and services communicate through an interface described in terms of Method and Event primitives. Methods and Events are asynchronous which means that one action does not necessarily follow the occurrence of another.

Methods are actions initiated by a service, to send information to the ASP through Method parameters. One example is AdvertiseService(), which triggers service advertisement by the ASP. Another example is SeekService(), which requests the ASP to search for WFDS services. A method call always returns a value back to the service immediately, which for instance indicates if the advertisement or search was successfully initiated.

Events are one-way actions which provide information from the ASP to a service about information obtained over a network. One example is the SearchRe-

sult() Event, which calls the service whenever the ASP has found a service name that matches the search criteria.

Chapter 4
Design and Implementation

4.1 Environment

The Android application runs on a SoMC code base, and it is written in the programming language Java. All code was developed in SoMC's own development environment SEMC Eclipse r22.3, a customized workspace for SEMC used for development of applications.

The implementation is using a SoMC modified Software Development Kit (SDK). SDK is a programming package that enables programmers to develop application for a specific software framework, software package, hardware platform etc.. The SDK includes Application Programming Interfaces (API) used for communication between software components.

The tool Gerrit was used for revision control (version and source). Gerrit is a web-based software code review tool that enables a more centralized use of Git. Gerrit allows multiple authorized users to submit changes to the master Git repository, users can also review modification on their source code.

4.2 Scope

The scope of the design and implementation for this prototype is shown in figure 4.1. The blocks encircled by the dotted area show the different components that are covered in this implementation including; Wi-Fi Direct, Application Service Platform, Send Service, and an Application layer for the User Interface and File Transfer. The colored blocks represent the components which didn't exist and were implemented from scratch for this prototype. The implementation is based on SoMC SDK, which has support for Wi-Fi Direct and some basic APIs for service discovery and adding local services.

4.3 Technical definitions

In order to help the reader to narrate through the implementation and design choices later in the chapter, this section presents a brief technical description about the fundamentals of asynchronous programming.

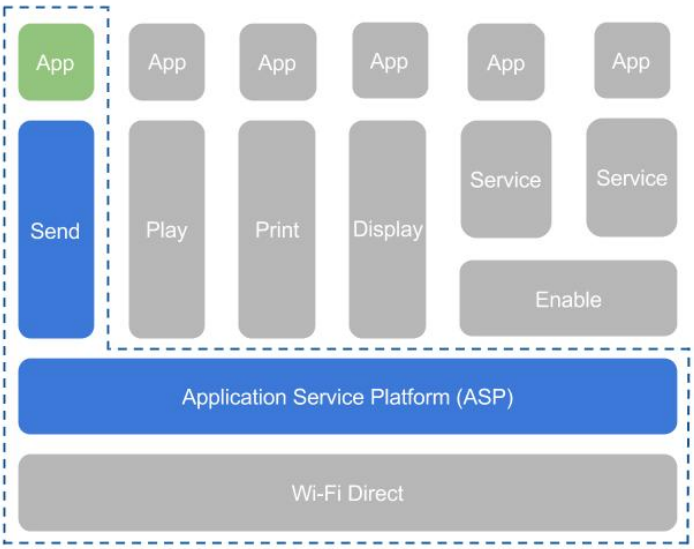

Figure 4.1: Scope of the Implementation.

4.3.1 Asynchronous Programming

The application needs to make regular calls to various time-consuming functions in the API, functions which are required to be executed simultaneously without blocking other parts of the application. In order to allow multiple tasks being performed side-by-side, a support for multi-execution was required (Asynchronous Programming). Without Asynchronous Programming, only a single task can be executed at once which results in blocking the application until the task is finished.

However, Asynchronous Programming is quite complex and not completely trouble-free. Multiple threads increases the risk of Race Conditions, which means that two or more threads tries to access and change the same shared data simultaneously. Therefore the sequence of execution and awareness to the application's constantly changing state must be taken into consideration. In this implementation, we have chosen to widely implement Handlers with the advantage of being able to hold queues and avoid race conditions.

AsyncTask

A large part of the application includes network functions, and a suitable choice was to use AsyncTasks due to its ability to perform background operation and publish result directly to the UI, without having to manipulate threads or handlers.

Handlers

With Handlers the different components of the application is able to communicate simultaneously without blocking the application. Each handler has their own

queue of tasks to be performed. When a task is placed in the queue, the task will be performed on the background thread in incoming order.

Java Service

The main advantage of Java Services is their ability to perform longer-running operations without having to supply functionality to other applications to use. Normally a Java Service is required to spawn a new thread if CPU intense operation is required. However, in our implementation, the Java Service will not run heavy calculation and is therefore running on the main thread.

4.4 Architectural Overview

The architecture of the prototype is illustrated in figure 4.2, describing the design of the implementation. The ASP is divided into two classes.

The prototype consists of four main components; User Interface (UI), Service, Application Service Platform (ASP) and File Transfer, which will be described more in detail in the following sections.

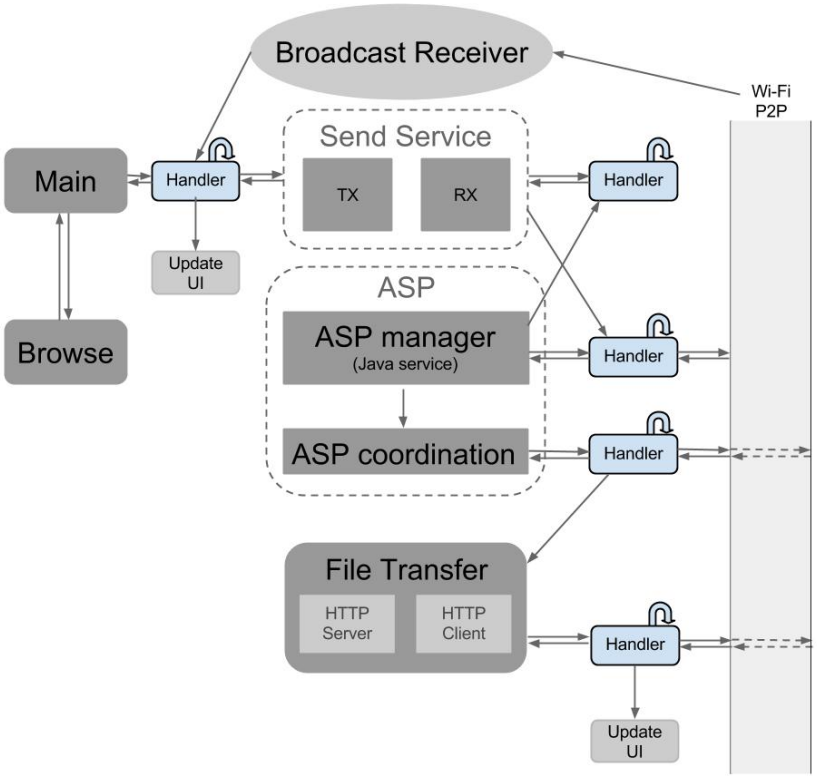

Figure 4.2: Architectural overview of implementation.

4.5 User Interface

An Application UI is implemented in this prototype to provide the user with information and the option to interact with the software. The UI consists of the three different Android Activities, each with a specific purpose. Android activities can be described as single presentation layers of an Android application.

Main Activity is the initial activity shown when the application is started. The main activity allows users to start adverting and seek for the specific service the device supports. The found services and devices are displayed in separate lists. Browse Activity allows the users to browse the content of current device and select files to send. File Transfer Activity displays real time information about the overall file transfer.

A very important requirement when it comes to user interfaces is reactiveness of the interaction with the user. By excluding network operation in the UI threads and using handlers correctly, heavy operations are avoided and the load on the UI threads are minimum.

4.5.1 Main Activity

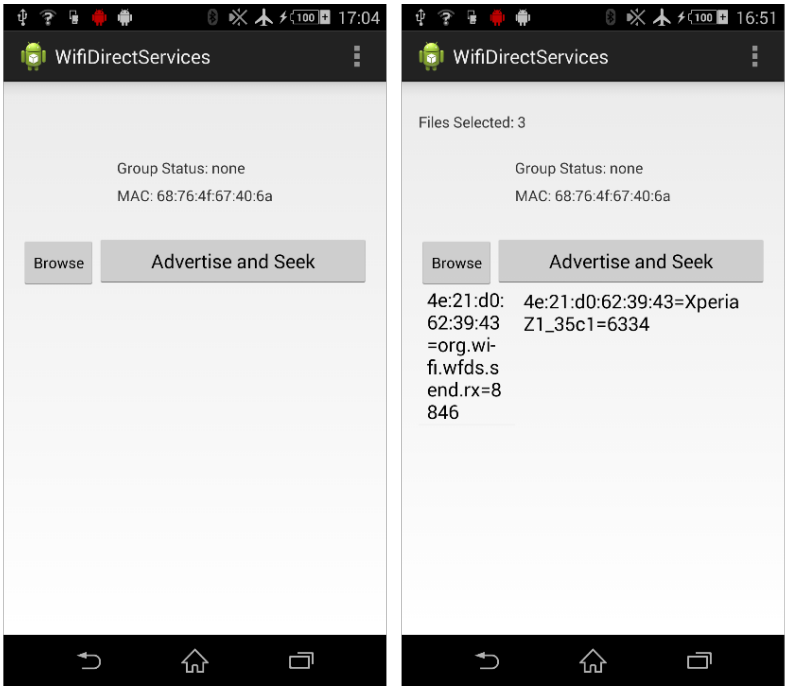

Figure 4.3: Left: Main Activity in its initial state. Right: Main Activity after files have been selected, advertisement and discovery have been initiated and a service has been discovered.

Figure 4.3 shows Main Activity in its initial state to the left and after service discovery to the right. The Main Activity contains a text field which shows information about the local device; MAC address, current P2P role (if any), and number of selected files retrieved from Browse Activity. The Main Activity displays two buttons, "Browse" and "Advertise and Seek". The first button switches the UI to the Browse Activity, the second triggers an advertisement of the local service. If a device is found it will appear in the right list under the buttons, if a service is found on the device it will appear in the left list.

4.5.2 Browse Activity

Figure 4.4: A screenshot of Browse Activity, showing the files in the internal storage.

Figure 4.4 shows the files in a device's internal storage that are displayed in the Browse Activity. The Browse Activity allows the user to navigate through the internal storage of the device and select files to be sent. The Browse Activity should only be used by devices that intend to become Send Transmitter and send the selected files in the coming Send-Session. When a file has been chosen, a global variable IS_TX is set to inform that this device now is the Send Transmitter. The Browse Activity automatically switches back to the Main Activity after a file is selected.

At least one file must be selected in the Browse Activity before the user is allowed to connect to a found Send Receiver entity in the list shown in Main

Activity. This is because during the establishment of the ASP-Session later, the Transmitter needs to send information to the Receiver about the files (before the file transfer). When a file is chosen, the path of the file is added to a global list.

4.5.3 File Transfer Activity

After an ASP-Session has been setup between the two devices, the Main Activity automatically switches to the File Transfer Activity. A Send button is displayed on the Transmitter device, which will initiate the file transfer when clicked. Two progress bars are also displayed in the File Transfer Activity, which will be updated during the file transfer to show how much is completed of the file transfer. The first progress bar visualizes the transfer status of the file that is currently being sent, and the second visualizes the overall transfer status for all files to be sent. If only one file is to be transferred, the two progress bars will be identical.

The total file size and number of files to be transferred are used to calculate the size of the progress bar. This file information is extracted from the REQUEST_SESSION message sent during the ASP-Session setup. Once the file transfer is completed, the average transfer speed and some additional information about the file transfer is displayed.

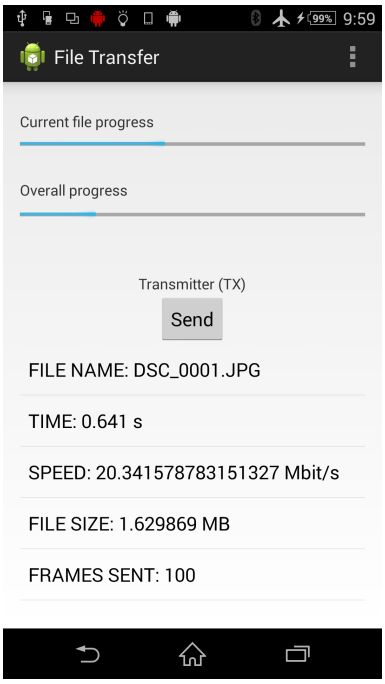

Figure 4.5: A screenshot of File Transfer Activity on the Transmitter device.

Figure 4.5 shows when one out of three selected files has been sent. After each file has been sent, information about the transferred file will be displayed under

the "Send" button.

4.6 Broadcast Receiver

Broadcast Receiver is a component that responds to system-wide announcements on a device. Many broadcasts originate for the system itself, for example when the battery level is low or the screen is turned off. This information is announcement through so called broadcast intents.

By implementing a Broadcast Receiver, these intents can be captured and followed up by initiating an action. In this implementation, the important intents to capture were the ones regarding status changes of Wi-Fi Direct. Therefore, an intent filter is initiated and added to the Broadcast Receiver to filter out the relevant incoming intents. The following code snippet below shows how the application initiates the Broadcast receiver, and adds the intent filter with specific Wi-Fi Direct actions.

```
/** Creating Broadcast Receiver and adding IntentFilter */
private void setUpWifiDirectBroadcastReceiver() {
    mFilter = new IntentFilter();
    mFilter.addAction(WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION);
    mFilter.addAction(WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION);
        .
        .
        .
    mReceiver = new WifiDirectBroadcastReceiver(mManager, mChannel,
        myUIHandler, this);
    registerReceiver(mReceiver, mFilter);}
}
```

When a broadcast intent is received, Broadcast Receiver automatically calls an "onReceive" method which extracts information about the intent. The information will then be used to determine what process should be triggered.

WIFI_P2P_PEERS_CHANGED_ACTION is an intent that is received when the available peer list has changed i.e. new devices with Wi-Fi capability has been found. When the application received this kind of events, it triggers a process that sends a request, through handlers, to update the device list in the Main Activity.

WIFI_P2P_CONNECTION_CHANGED_ACTION is an intent that is received when the connection of a P2P device has changed, for example when the device has joined or abandoned a P2P Group. If the device joined a P2P Group, information about the device's role in the P2P Group and the GO's IP address is extracted from the intent. This information is stored, and the Main Activity is updated with the new P2P role of the device (the "Group Status" field).

4.7 Send Service

The Send Service's Transmitter entity and Receiver entity are implemented separately. Each entity is responsible of initiating the advertisement of itself and the search of the opposite entity on other devices.

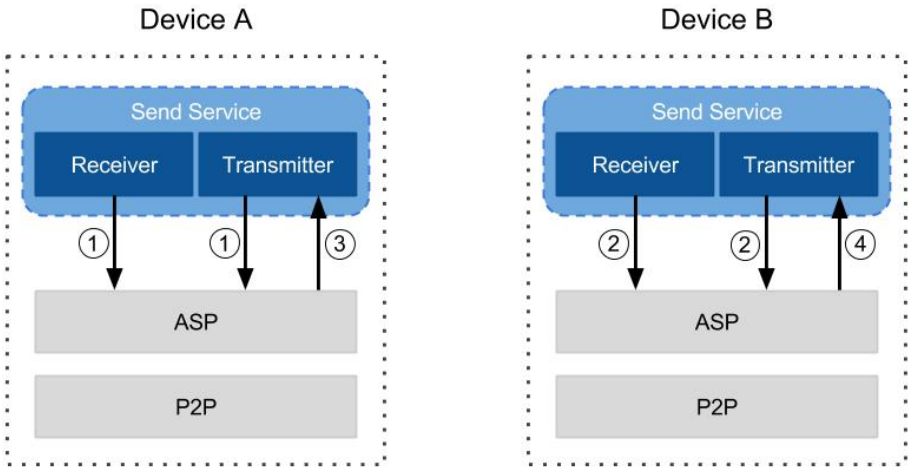

Figure 4.6: Service Discovery Sequence Diagram.

Figure 4.6 shows the sequence diagram of a typical service discovery between two devices with the following steps:

1. Receiver entity on device A advertises itself with the service name "org.wi-fi.wfds.send.rx" . The Transmitter entity initiates a search for Receivers with the service name "org.wi-fi.wfds.send.rx".

2. Receiver entity on device B advertises itself with the service name "org.wi-fi.wfds.send.rx" . The Transmitter entity initiates a search for Receiver with the service name "org.wi-fi.wfds.send.rx".

3. Device A found the Send Receiver on device B. The service name matches the search criteria, so the ASP forwards the result to the Transmitter entity.

4. Device B found the Send Receiver on device A. The service name matches the search criteria, so the ASP forwards the result to the Transmitter entity.

Whenever the ASP forwards a search result to the Transmitter entity, the list of found services in the Main Activity is updated. If device A wants to send files to device B, the user clicks on the Receiver entity in the Main Activity list to initiate a connection. This action will call the ConnectSessions() method, telling the ASP that the Send Transmitter is ready to connect with the selected entity.

4.8 Application Service Platform

The ASP is roughly divided into two types of operation, network operations and internal operations. Therefore, the implementation of the ASP is also divided into two classes; ASP Manager and ASP Coordination. The ASP Manager has the responsibility of managing the internal calls within a P2P device, while the ASP Coordination manages the external communication, shown in figure 4.7.

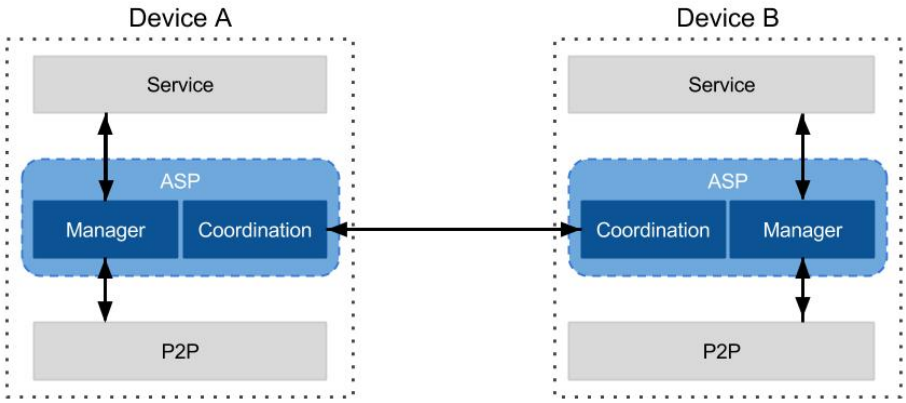

Figure 4.7: The ASP Coordination and ASP Manager operations.

The different functionalities of the two ASP classes are explained below:

ASP Manager - Handles the communication between ASP and a service through Events and Methods. It also manages the communication between ASP and the P2P layer, for example by forwarding a service's advertisement and search requests.

ASP Coordination - Handles the communication with the ASP on a remote devices to setup and remove ASP-Sessions. The implementation of the entire ASP Coordination Protocol is also located in this class, along with the encoding and decoding of ASP coordination messages.

4.8.1 ASP Manager

It is necessary that the ASP Manager is always available without being blocked while the application is running. The ASP Manager is implemented as a Java Service, which runs on the UI thread but does not necessarily needs to communicate with the UI. When an ASP Manager is initiated, it starts up a separate thread for the Handler to avoid CPU-intensive work on the UI Thread.

ASP Communication with Service and P2P Layer

The ASP Manager handles of all ASP communication with the overlying Services and the underlying P2P component. It is responsible for forwarding requests from

service entities down to the P2P layer and vice versa. Figure 4.8 describes a typical call sequence for the ASP Manager during service advertisement and discovery.

In this implementation, the Bonjour protocol was selected as Service Discovery Protocol for the advertisement and discovery of Send Service entities. Bonjour is a standardized implementation of the Zero-Configuration Networking (Zeroconf), developed by Apple [28].

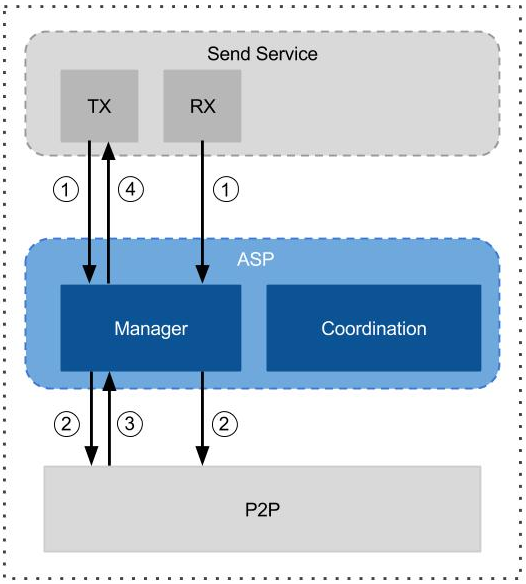

Figure 4.8: Sequence of Internal Calls for Advertisement and Discovery.

Figure 4.8 illustrates a typical internal call sequence between the layers when service advertisement and discovery is initiated. Each step in the figure is explained below:

1. The Transmitter entity calls the SeekService() Method and the ASP Manager immediately returns a generated search ID back to the Transmitter. The Receiver Entity calls the AdvertiseService() Method and the ASP Manager immediately returns a generated advertisement ID back to the Receiver.
2. The ASP Manager creates a service request with the search criterias from the Transmitter. Service discovery is initiated in the P2P layer, which listens for Bonjour Service Information responses that matches the search criteria.

   The ASP Manager creates an instance of a Bonjour Service Information Object for the Receiver. The Receiver is registered as a local service in the P2P layer, which initiates an service advertisement.
3. The P2P framework automatically respond to service discoveries on other devices and send the information back to the ASP Manager.

4. The ASP Manager generates a SearchResult() Event and the discovered service is displayed in the Main Activity.

Initiating the ASP Coordination

When the ASP Manager receives a ConnectSessions() method call from the Send Transmitter entity, it makes a call to the underlying P2P component to initiate a P2P connection with the remote device.

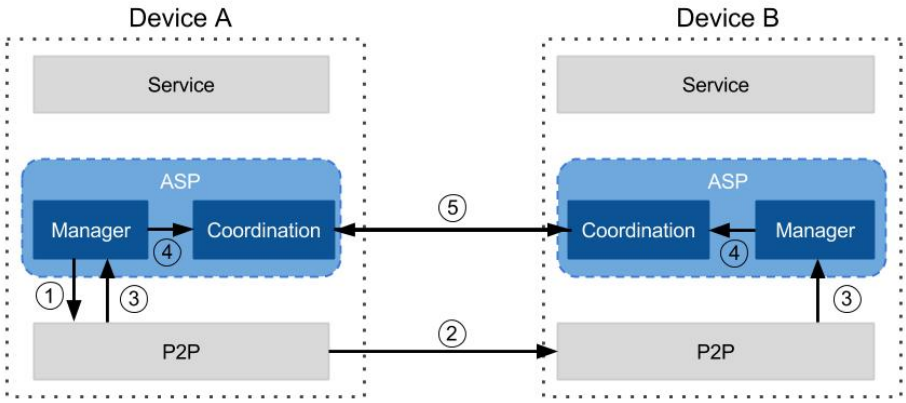

Figure 4.9: Setup of ASP Coordination.

Figure 4.9 illustrates the sequence diagram when device A initiates a connection with device B. Each step in the figure is explained in the list below:

1. User on device A chooses a service to connect to on device B. The ASP Manager on device A tells its underlying P2P component to connect to device B.
2. Device A initiates a P2P connection with device B forms a P2P Group.
3. The two devices have been assigned the GO or P2P Client role in the new P2P Group. The ASP on respective device is informed that the group formation is completed through a broadcast intent.
4. ASP Manager on respective device initiates the ASP Coordination, setting up an UDP socket.
5. ASP Coordination on device A and ASP Coordination on device B communicate through the UDP socket.

4.8.2 ASP Coordination

The ASP Coordination is responsible for the setup and removal of an ASP-Session. When an ASP Coordination has been initiated, it checks if the device is assigned the GO or P2P Client role. Two communicating ASP Coordinations uses an

UDP socket to exchange information during the setup and removal of an ASP-Session. ASP Coordination uses an AsyncTask thread because it requires network operations.

The destination of an UDP datagram, is determined by the IP address of the destination. In current Wi-Fi P2P standard API, only the P2P Client has access to the GO's IP address and not vice versa. Since the ASP coordination protocol requires a two-way communication, the GO needs to know the IP address of the P2P Client, which means the P2P Client therefore needs to be the one who sends the first message.

When setting up an ASP-Session, the first ASP Protocol message is always sent from the Service Seeker (which in this case is the Send Transmitter). But since the Service Advertiser/Seeker roles are independent of the P2P Client/GO roles, it could mean that GO is required to send the first VERSION message to the P2P Client (to which the GO does not have the IP address). This was solved by adding a datagram which the P2P Client sends to the GO, before any actual ASP Coordination Protocol messages are being transmitted. This additional datagram informs the GO of the P2P Client's IP address allowing a two-way communication where the GO can send datagrams back.

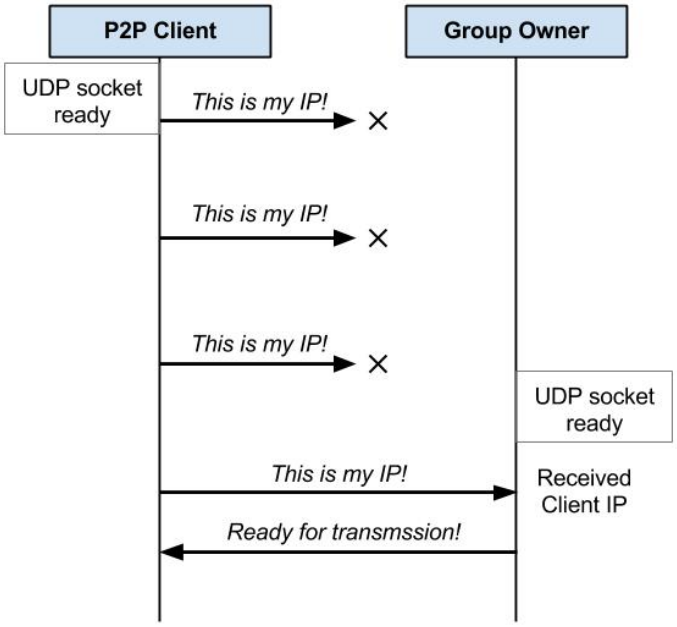

Figure 4.10: Retransmission of first UDP datagram.

The setup of a UDP socket on respective device is initiated only after a broadcast intent is received, informing which P2P role a device has in the established P2P Group. However, there is no way for either device to know if the remote device's UPD socket has been setup or not. If the P2P Client's socket is setup and immediately sends the first datagram before the GO has setup its socket, there's

a high risk that the datagram containing the IP address of the P2P Client is lost. The solution was to let the P2P Client repeatedly resend this datagram with with short intervals until the server responds, informing that it is ready for the ASP Coordination Protocol messages. Figure 4.10 illustrates how the retransmission of the first UDP datagram continues until the GO responds to the P2P Client.

Once the UDP socket on both devices are ready for transmission, the ASP Coordination on both devices check their variable IS_TX (set in Browse Activity) to see who is sending the first message. The Service Seeker, in this case the Transmitter, always starts sending the first VERSION message.

The UDP protocol does not provide any guarantee for the delivery of datagrams. When an ASP Coordination Protocol message has been sent, the sender cannot know if the datagram has reached its destination or not, or even if the message arrive in the correct order. UDP supports neither acknowledgement, retransmission, nor timeout of messages, which makes it an unreliable transmission protocol.

To overcome this, the ASP Coordination Protocol specifies its own transmission control mechanism in order to make the data exchange more reliable. This mechanism verifies that each received message is the expected one with correct sequence number, containing valid information. Each valid Command message is responded with an Acknowledgement message. In figure 4.11, the Command messages are highlighted with bold font and the Acknowledgement messages with italic.

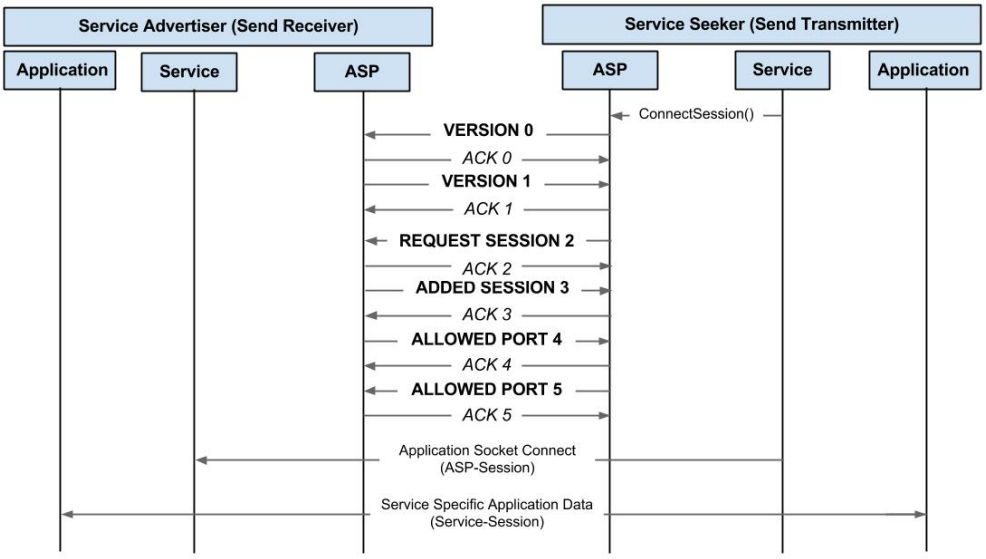

Figure 4.11: Message Sequence Diagram for ASP-Session Setup.

When the sequence 5 message has been sent from both devices, the ASP Coordination on respective device tells the ASP Manager to initiate the File Transfer.

ASP Coordination Protocol

ASP Coordination Protocol messages are single datagrams sent as byte streams. The information in each message needs to be encoded to a byte stream before transmission and decoded back once received on the other side. All messages starts with an Opcode that is unique for each message type, followed by a sequence number. The payload size and information fields are unique for every message type. The message format for ASP Coordination Protocol messages is shown in figure 4.12.

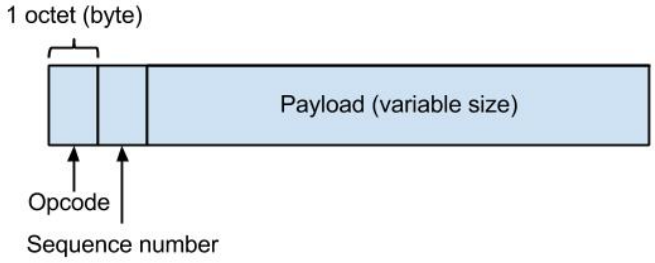

Figure 4.12: Message format of ASP Coordination Protocol.

When a message is received as a byte stream, the ASP Coordination entity first checks the sequence number in the second byte. If it's correct, the opcode in the first byte is used to determine the message type and if it corresponds to the expected message type for the specific sequence number. Since the payload is unique for every message type, the opcode is used to determine how information is to be extracted and decoded from which fields in the payload. The functions of the different message types is briefly explained below in chronological order:

- VERSION, Opcode 5 - Information about which version of the WFDS Technical Specification the device support. This message is sent from both Transmitter and Receiver.
- REQUEST_SESSION, Opcode 0 - Transmitter sends information about the file(s) to be sent and specifies the advertisement id of the service entity it wish to connect to. Message contains information about the number of files, total size and file name(s).
- ADDED_SESSION, Opcode 1 - Receiver informs that it is has confirmed the ASP-Session about to be setup.
- ALLOWED_PORT, Opcode 4 - Information about which port will be used for the Send-Session. This message is sent from both Transmitter and Receiver.
- ACK, Opcode 254 - Acknowledgement message informing that previous command was received and that the information was valid.

4.9 File Transfer

After the ASP-Session has been setup, the ASP Manager initiates the File Transfer to setup the Send-Session. Similar to the ASP Coordination, the File Transfer require network operations and therefore implements an AsyncTask thread.

The Send Service is required to use the HTTP protocol for the Send-Session Data Plane. HTTP is an OSI Application Layer protocol that encapsulates the Transmission Control Protocol (TCP), which means the HTTP messages need to be transported through a TCP socket. TCP is a connection-oriented protcol in the OSI Transport Layer [19]. Unlike UDP, it handles all transmission failures automatically and guarantees that transmission data arrives correctly [29].

4.9.1 HTTP Client and Server

The Send Transmitter utilizes the HTTP Client role. The selected file to be sent will be converted into a byte stream, divided and allocated into fixed frames. The HTTP Client sends the file data by using a PUT Request message to the HTTP Server.

The Send Receiver utilizes the HTTP Server role. When all frames in a file has been received, the HTTP Server reassembles the files from the byte streams and stores it in the internal storage. The Send Receiver sends a HTTP Response back to the HTTP Client to inform that all frames containing the file have arrived successfully.

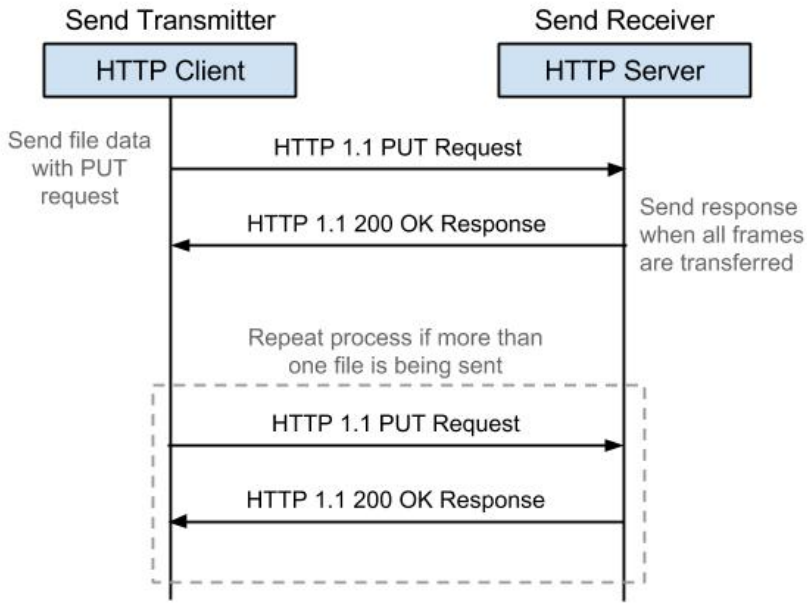

Figure 4.13: The HTTP Send Session between Send Transmitter and Send Receiver.

If more than one file is to be sent, this procedure is repeated for each file, illustrated in figure 4.13. The HTTP Client sends a new HTTP Request message containing the next file, and the Server responds accordingly when the file has arrived successfully.

4.9.2 HTTP Request and Response message

All HTTP messages are divided into a message header and a message body. The header always starts with a single request or status line, followed by header fields and field values separated by a colon (":"). The Message Body is used to carry the entity-body, which in this case is the file itself in bytes. The Message Header and the Message Body are separated with a blank line [30].

Once the processing of a file into bytes streams is done, the HTTP Client sends a PUT Request message to the HTTP Server. The message header contains the destination and information about the file, and the message body contains the file itself (in bytes).

HTTP PUT Request

```
PUT 100ANDRO/DSC_0001.JPG HTTP/1.1
Host: 192.168.49.1:8081
Content-Type: image/jpeg
Content-Length: 1629869

<! ------- Actual File Goes Here ------- !>
```

The request line starts with the PUT method token, followed by the URI of the file (container structure and file name) and the HTTP protocol version. The three following lines are all header fields. The Host field contains information about the destination, which includes the IP address and port of the Send Receiver device. Content-Type represents the Multipurpose Internet Mail Extensions (MIME) type of the file, which is determined by the file name [31]. The Content-length field indicates the length of the message body in bytes. If a HTTP message is divided into multiple frames that are transferred separately, the HTTP Server uses the Content-length to determine where the message body containing the file data ends.

HTTP OK Response

```
HTTP/1.1 200 OK
Content-Length: 12

PUT Complete
```

When the HTTP Server has received the amount of data that is specified in the Content-Length field of the request header, it shall respond with a HTTP Response message back to the HTTP Client. The response line starts with the HTTP protocol version followed by a status code and reason phrase. If the request

succeeded, the status code is set to 200 and reason phase to OK. The header field contains the Content-Length indicating text length in the message body, which in this case is a "PUT Complete" text string.

Chapter 5
Experiment and Result

5.1 Environment and Tools

The experiments we performed include measurements for throughput, discovery time and power consumption. The prototype required support for Wi-Fi Direct and the most suitable choice on SoMC hardware was the Sony Xperia Z1. The device holds a Qualcomm MSM8974 board with support for Wi-Fi 802.11a/b/g/n/ac with a maximum throughput of 325 Mbit/s, specifications shown in table 5.1 [24].

Model	Sony Xperia Z1 C6903
Operating system	Google Android 4.2.2 (Jelly Bean)
Board	Qualcomm MSM8974 Snapdragon 800
Processor	2.2 GHz Quad Core Krait 400
GPU	Aderno 330

Table 5.1: Sony Xperia Z1 Specifications.

The tools used for the performance measurements are described below:

- Anritsu Shield Box - A shield box allowing over-the-air testing of wireless devices and data cards. The internal wide bandwidth antenna enables testing of LTE, W-CDMA/UMTS, CDMA2000, GSM , WLAN, and other wireless devices [25].
- Agilent 34410A - A digital multimeter which can be used for current and voltage measurements.
- LabVIEW - A graphical programming platform tool which can be used to automate measurement processes, in this case used for power measurements [26].
- iPerf - A network testing tool to measure TCP and UDP bandwidth performance [27].

All measurements were done in an in-door office environment with open landscape, but throughput and discovery time measurements were performed inside a shield box to avoid interference from other wireless devices in the office landscape. The Wi-Fi Direct throughput was measured using iPerf software on two devices,

approximately 10 cm apart. The transmitting device was connected to Android Debug Bridge (ADB) where data was monitored and captured. The captured data was later exported for plotting.

The power consumption measurements were done on a single device, by connecting a digital multimeter directly to the device's battery. The device was set to Airplane Mode with only Wi-Fi turned on, and no additional applications were running in the background. The measurements were performed and displayed using LabVIEW software.

5.2 Throughput

The throughput measurements were performed using the iPerf software to test the TCP throughput capacity. Each measurement was performed several times to give a more accurate value of the average throughput. Sample rate for each measurement was 1x per second. Figure 5.1 shows the result of the throughput measurement for 1 minute, where the Transmitter is GO and Receiver is P2P Client.

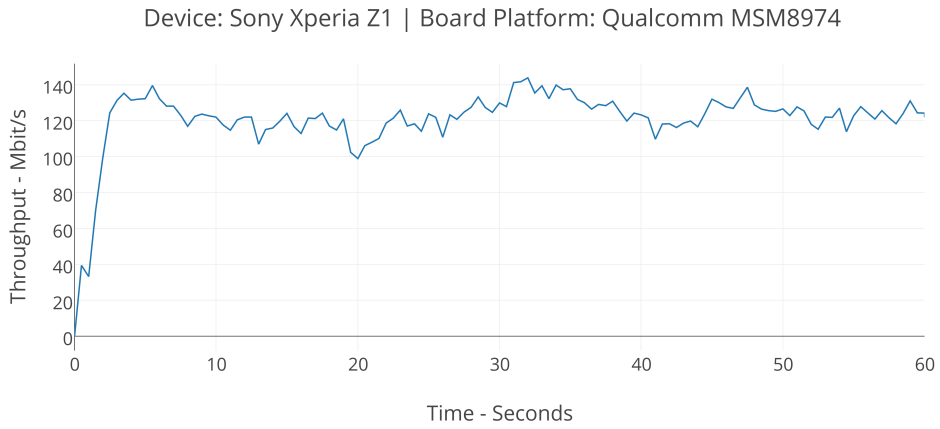

Figure 5.1: Throughput measurement during 1 min, GO to P2P Client.

Figure 5.2 shows the result of the throughput measurement for 5 minutes, where the Transmitter is GO and Receiver is P2P Client.

Figure 5.3 shows the result of the throughput measurement for 30 minutes, where the the Transmitter is P2P Client and Receiver is GO.

To put in perspective how much data that is actually transmitted in the throughput measurements, we use media contents to illustrate this in our analysis. The values chosen to represent the media contents are standard sizes of a picture or video taken with a Xperia Z1 camera with the highest resolution:

- Picture = 4.6 MB (taken with 20,7 MP, 5248 x 3936 pixel)

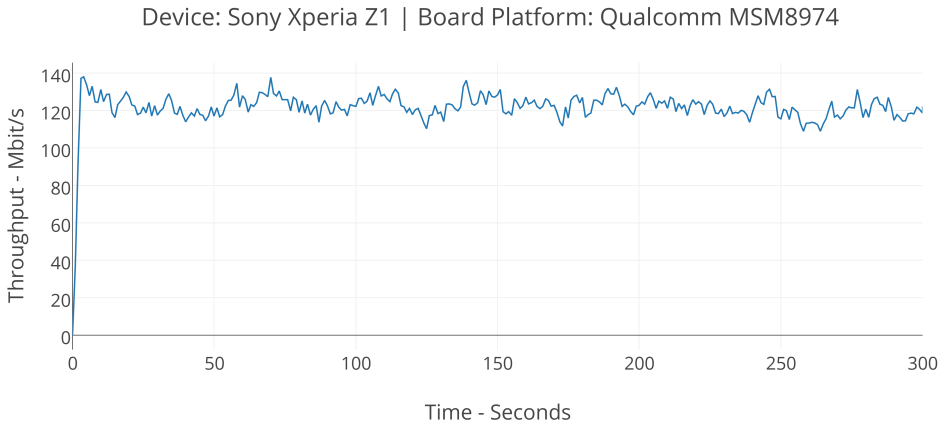

Figure 5.2: Throughput measurement during 5 minutes, GO to P2P Client.

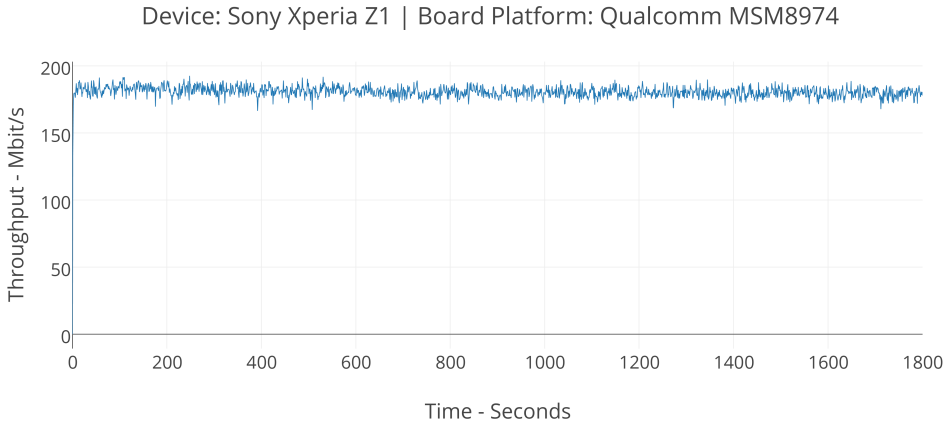

Figure 5.3: Throughput measurement during 30 minutes, P2P Client to GO.

- Video = 110 MB/min (Full HD, 1920 x 1080 pixel)

The results from figure 5.1 and 5.2 shows that the average throughput from a GO to P2P Client is around 125 Mbit/s. With this average transfer speed, a device can transfer up to 935 MB during one minute, which is equivalent to 2000 pictures or 8,5 min of Full HD video in mp4 format. During five minutes, this speed would correspond to 4,35 GB data transfer, which is equivalent to 945 pictures or 40 min of Full HD video. Figure 5.3 shows that the average throughput from a P2P Client to GO is around 180 Mbit/s, which is slightly higher than from a GO to P2P Client. The amount of data that can be transferred with this speed during

30 min is 29,9 GB, which is equivalent to 6522 pictures or 272 min (4,5 h) of Full HD video. Some dips caused by internal operations can be observed in the figures, which could be a result of the Wi-Fi scan being performed in the background.

From these results we can draw the conclusion that Wi-Fi Direct, with the measured average transfer speed above 100 Mbit/s, is a very good alternative for transferring media contents. Whether it is between mobile devices or from a camera to a tablet, Wi-Fi Direct Send service allows a very fast file transfer of high resolution media contents (such as 4K video).

5.3 Discovery Time

The large time variations for device- and service discovery was known before the measurements were performed. Therefore, the measurements were performed 100 times each to give an overview of the time variation for discovery (in seconds). Figure 5.4 shows the distribution of time variation in device discovery for a specific device.

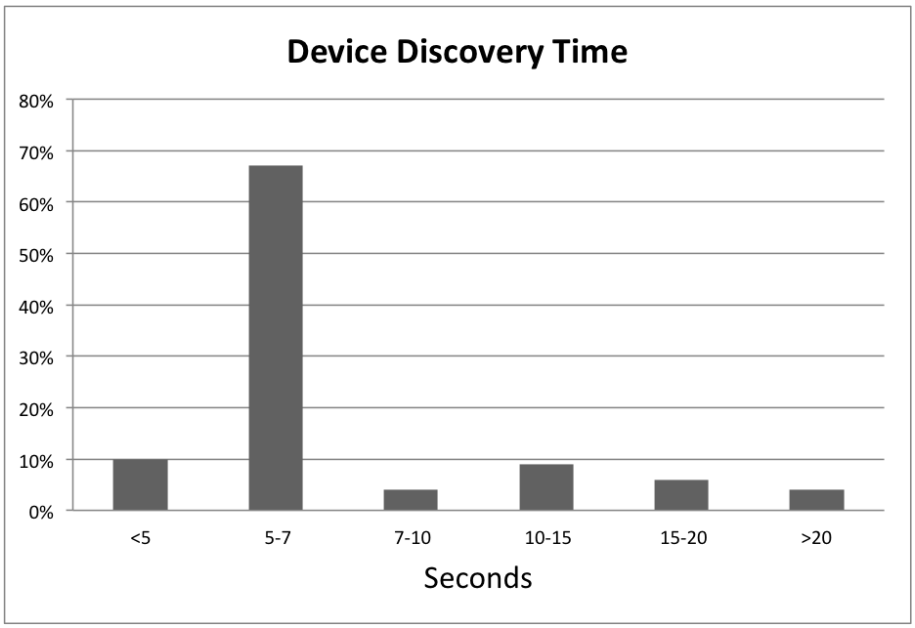

Figure 5.4: Device Discovery time for a specific service.

Figure 5.5 shows the distribution of time variation in service discovery for a Send Transmitter.

Since a service discovery can only be performed on devices that already have been found, it explains why the service discovery time is longer than device discovery. As seen in the results from the previous chapter, the discovery time of devices and services can vary quite a lot from time to time, but is generally performed

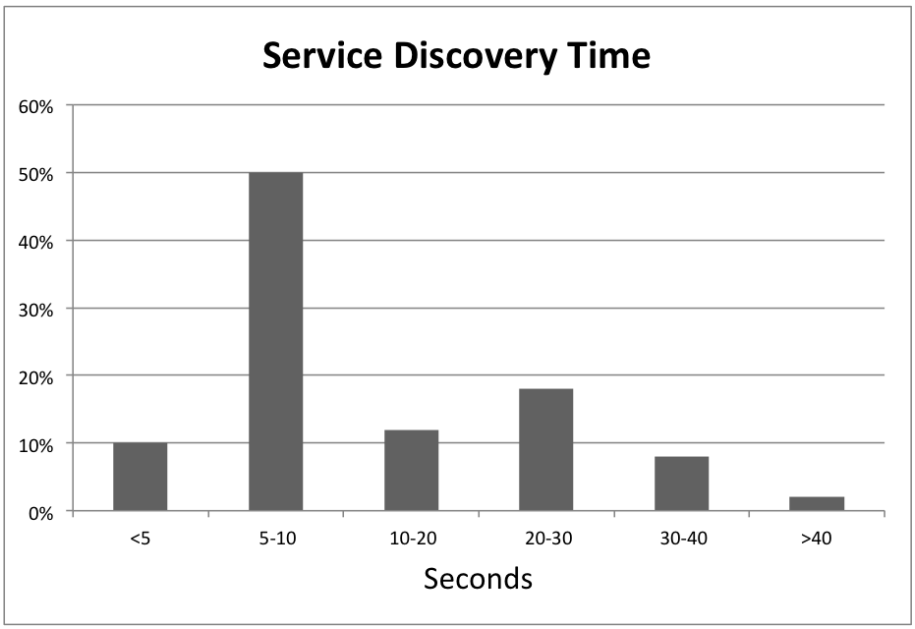

Figure 5.5: Service Discovery time for a name-specific service.

relatively fast. Figure 5.4 shows that approximately 75% of the times, a specific devices was discovered under 7 seconds. Figure 5.5 shows that approximately 50% of the times, a name-specific service was found under 10 seconds.

This time variation is mainly a result of the way the search mechanism operates in the Find phase, the device is required to switch between three different channels and alternate between listening and searching. Although it might not seem like the most efficient implementation, in reality it would be hard to narrow down the discovery time noticeably.

5.4 Power Consumption

Each measurement was performed during 5 minutes in idle mode, both with and without the application running in the background. The sample rate was chosen 100x per second, i.e. every 10 ms, in order to capture all the significant power changes. Figure 5.6 shows the power consumption for a device in idle mode when it has the Client role in a P2P Group. Figure 5.7 shows the power consumption in idle mode when the device has the Client role with the application running in the background.

Figure 5.8 shows the power consumption for a device in idle mode when it has the GO role in a P2P Group. Figure 5.9 shows the power consumption in idle mode when the device has the GO role with the application running in the background.

The results from the power consumption measurements shows that a GO device

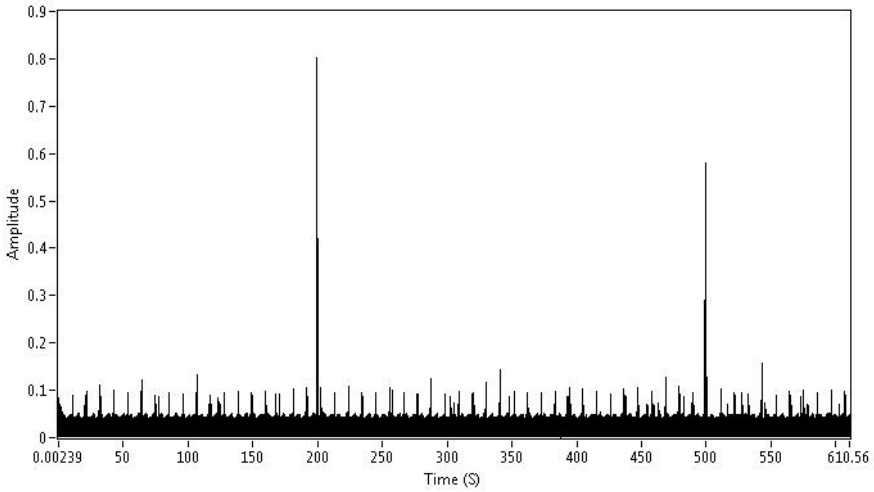

Figure 5.6: Power consumption for P2P Client.

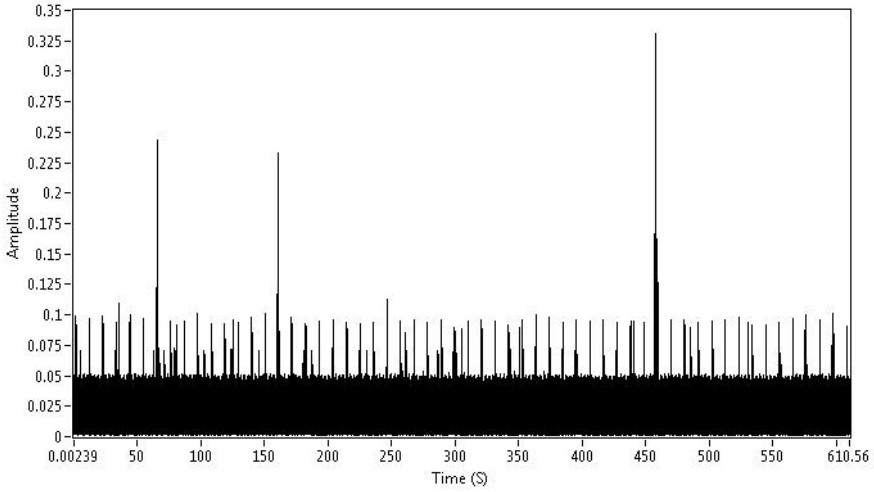

Figure 5.7: Power consumption for P2P Client with application.

consumes significantly more power than a P2P Client device, which is because of the Soft AP role a GO has. As a Soft AP, the GO has to announce its presence by broadcasting beacon frames which are captured by the P2P Clients, which explains the periodic peaks in the figures.

It is also shown that the GO is consuming more power with the application running, compared to the P2P Client which doesn't consume noticeably more power with the application. We believe the reason has to do with GO automatically setting up an open TCP Server socket that is ready for transmission. In the

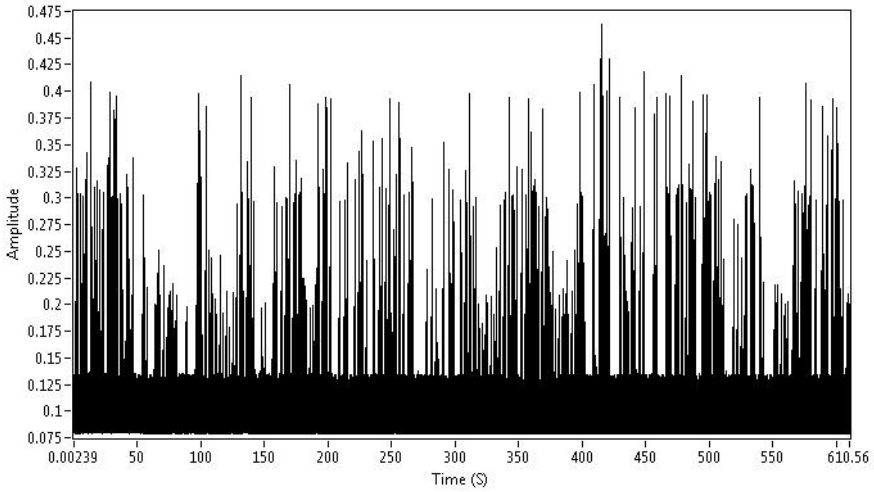

Figure 5.8: Power consumption for GO.

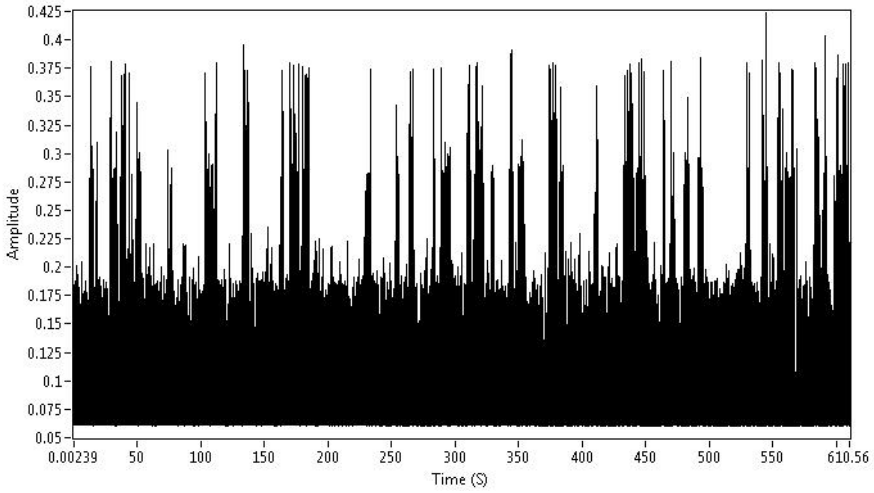

Figure 5.9: Power consumption for GO with application.

measurement with the application, no transmission was actually performed in the idle mode, but the TCP Service socket was constantly open and ready for transmission.

Chapter 6
Discussion

6.1 Comparison between Wi-Fi Direct and similar wireless technologies

Table 6.1 gives an overview of the differences between Wi-Fi Direct and similar wireless technologies. A comparison in terms of range, throughput and power consumption is presented.

	Wi-Fi Direct	Infrast-ructure*	Ad hoc*	Bluetooth	BLE	NFC
Range	H	H	H	M	M	L
Throughput	H	H	M	L	L	L
Power Con-sumption	M-H	M	H	L	L	L

Table 6.1: Comparison in performance between wireless technologies. L=low, M=medium, H=high.

6.1.1 A comparison with Wi-Fi

Wireless Ad hoc networks were until the introduction of Wi-Fi Direct the only peer-to-peer solution within the Wi-Fi technology. However, it never reached any particular success because of its lack of security, high power consumption and relatively lower throughput. In Wi-Fi Alliance terms, Ad hoc refers to an aging Wi-Fi device-to-device transfer method that was painful to set up with a maximum throughput limit at 11 Mbit/s.

In wireless Ad hoc networks, the control is distributed over all the devices within the network, which means that every device needs to forward traffic unrelated to its own use. Due to the lack of a central unit, Ad hoc networks suffer from security and Quality and Service (QoS) issues. The QoS measures the quality of the network service such as; error rates, throughput, transmission delay and availability. An example of a security threat could be a malicious device connected

to the network can easily capture bypassing data, which will violate the privacy. From a power consumption perspective, each device in a wireless Ad hoc network consumes more power compared to Wi-Fi Direct due to the routing requirements (forwarding data and constantly updating their routing table). Every device is also required to constantly announce its presence and detect possible changes in network structure. This results in a heavy network load that limits the throughput within the network. In a Wi-Fi Direct group, only the GO is responsible for controlling and managing the traffic in its network. This means that each P2P Client doesn't have any responsibility toward the other P2P Clients, it only needs to care about the communication between itself and the GO. Wi-Fi Direct also promises regular Wi-Fi speeds of up to 250 Mbps and some QoS mechanisms, combined with a much easier to set up and use than Ad hoc. Since the range of an Ad hoc network is determined by how the stations are placed, in theory it means that the range of the network is not limited. An Ad hoc network is therefor more flexible than a Wi-Fi Direct group in these aspects, which in some cases could be an advantage. However, it does require that stations in the network do not relocate and disappear out of range, something that is hard to guarantee.

Compared to the WLAN Infrastructure, Wi-Fi Direct's group topology can seem quite similar to the infrastructure star topology at first. However the communication within a P2P Group and an WLAN infrastructure network are very different. P2P Clients in Wi-Fi Direct are only intended to communicate with the GO, which means that there is no communication between the P2P Clients passing through the GO. In a WLAN infrastructure on the other hand, the main purpose of the AP is to forward data between the clients in its network. Wi-Fi Direct is simply not intended to be used as a permanent network, but more as a temporary network that can be setup anywhere at any time, allowing direct and flexible communication. A Wi-Fi Direct device also does not require any special hardware compared to a traditional Wi-Fi AP device.

6.1.2 A comparison with Bluetooth

The functionalities WFDS offer are in many ways very similar to the functionalities offered by the Bluetooth technology. Even the network structure in a Wi-Fi Direct P2P Group is similar to a Bluetooth Piconet. Basically all modern mobile devices today have support for Bluetooth, and the technology is already a so wellknown industry standard. By using Bluetooth, it basically guarantees interoperability with almost any devices. Until now, Bluetooth has been the first hand choice for many use scenarios where users want to transfer a file to a peer device or connect to a wireless speaker.

One main advantage with Bluetooth is the lower power consumption it offers in comparison with Wi-Fi and Wi-Fi Direct, especially since the introduction of BLE. This has made the technology very popular among various gadgets on the market today that need to communicate with other devices. Many of these connections are intended to last for a longer period of time, but do not necessarily require a large data transfer. For example when a wireless keyboard or mouse is connected to a computer, or a smartwatch that needs to stay connected with a mobile device for a longer period. Therefore Bluetooth is ideal in these type of use cases, since

the gadgets generally hold a very small battery with less than 200 mAh and need to limit the power consumption. Using Wi-Fi Direct in these types of gadgets would not be very suitable since it would consume more power (GO in particular).

But the technology also has its weaknesses, and one of them is the limited data transfer speed. Bluetooth Classic only supports a theoretical throughput of up to 2.1 Mbit/s at baseband level while BLE only supports up to 1 Mbit/s. Although Bluetooth promises simultaneous connections with multiple devices, the time-division makes the actual throughput to be lowered for each additional device. In comparison with the measured throughput values that Wi-Fi Direct offers, it's obvious that Wi-Fi Direct is much more suited for use cases involving larger amount of data transfer. The difference in range between Wi-Fi Direct and Bluetooth is also important to mind. The theoretical range is up to 10 m for class 2 Bluetooth Classic devices and up to 100 m for BLE devices [32].

6.1.3 A comparison with NFC

The NFC technology is significantly different from many other wireless technologies due to its very limited range. The use cases NFC offers are significantly different from Wi-Fi Direct. NFC is only intended for transferring of small amount of data during a very short period of time, due to the fact that the devices cannot be separated as a result of the short range. Wi-Fi Direct on the other hand is intended for transferring large amount of data over a larger distance, which makes the use cases for the two technologies very different. However, the short range NFC offers do provide additional security, which makes the NFC Handover feature attractive to combine with other wireless technologies.

Combining Wi-Fi Direct with NFC for WPS

In April 2014, NFC Handover was added as a setup method in WPS. The combination of NFC with Bluetooth is already well established, where NFC's "one-touch" motion is used for quick pairing of Bluetooth devices. But until now, using PIN code and Push-button method has been the only available alternatives for WPS.

Since WPS is used in both Wi-Fi and Wi-Fi Direct, this new addition means that a standardized combination of NFC and Wi-Fi Direct is made possible. By tapping two Wi-Fi Direct devices together after service discovery, NFC automatically configures the group establishemnt, allowing the devices to connect immediately with minimum effort.

6.2 Existing solutions with WFDS functionality

Today there are a number of popular wireless solutions on the market that has similar functionality to WFDS. They basically achieve the same tasks as the services in WFDS is intended for, and many of them are also based on the Wi-Fi Direct technology.

Table 6.2 presents existing peer-to-peer solutions from competitors that corresponds services defined in WFDS. Many of these solutions are to large extent brand specific which limits the interoperability between devices.

	Send	Play	Display	Print
Android Beam [33] (Android)	BT+NFC			
S BEAM [34] (Samsung)	WFD+NFC			
SuperBeam [35] (Third Party App)	WFD(+NFC)			
Wi-Fi Shoot! [36] (Third Party App)	WFD			
Airplay [37] (iOS)		Proprietary protocol	Proprietary protocol	
Allshare [38] (Samsung)		DLNA	Miracast	
Screen Mirroring [39] (Sony)			Miracast	
SmartShare [40] (LG)			Miracast	
Samsung Print [41] (Samsung)				WFD+NFC

Table 6.2: Existing solutions that correspond to the services in WFDS.

Android Beam is a feature that was introduced in Android version 4.0x Ice Cream Sandwich that allows smaller amount of data to be transferred through NFC, e.g. contacts or web bookmarks. From version 4.1x Jelly Bean, Android Beam uses NFC handover to establish a Bluetooth (BT) connection for the file transfer. As the name indicates, Android Beam is limited to Android devices that has NFC support. SBEAM is limited to Samsung devices only, and uses NFC to establish a Wi-Fi Direct (WFD) connection for file transferring. SuperBeam and Wi-Fi Shoot! are both third party applications that enables file transfer using Wi-Fi Direct.

Airplay offers wireless media streaming from iOS devices and uses a proprietary protocol developed by Apple. Allshare Play is limited to Samsung devices, and enables media streaming through a DLNA connection.

Airplay Mirroring allows wireless display mirroring from iOS devices to an

Apple TV. Apple TV is not a TV but a digital media receiver that can be connected to a compliant TV that displays the content. Allshare Cast from Samsung, Screen Mirroring from Sony, and Smartshare from LG are basically implementations of Miracast under different names.

Samsung Mobile Print uses Wi-Fi Direct for wireless printing from a Samsung or iOS device. It also offer NFC for setting up the Wi-Fi Direct connection.

6.3 WFDS use cases for Sony products

Since Sony offers a large variety of different consumer electronic products, it would be advantageous to integrate the services Send, Play, Display and Print into existing products. Many of these products already have support for Wi-Fi, and would therefor not require any additional hardware for Wi-Fi Direct.

As there are significant differences in functionality between a Transmitter (TX) and a Receiver (RX) in every service, some products are only suitable to implement one of the roles. For example, the only device that should be able to implement the Print Receiver entity is a wireless printer, while a number of different products can implement the Print Transmitter entity. Table 6.3 presents a proposition for how the services entities in WFDS can be integrated into Sony's existing products.

	Send		**Play**		**Display**		**Print**	
	TX	RX	TX	RX	TX	RX	TX	RX
Mobile device	**x**	**x**	**x**		**x**		**x**	
Tablet	**x**	**x**	**x**		**x**	**x**	**x**	
Laptop	**x**	**x**	**x**		**x**		**x**	
Camera	**x**		**x**				**x**	
TV				**x**		**x**		
Printer								**x**
Portable HDD		**x**						

Table 6.3: WFDS use cases in Sony's existing products.

Chapter 7
Conclusion

In this master's thesis we have presented an in-depth analysis in the Wi-Fi Direct technology. A prototype for WFDS was developed in form of an Android application based on the specifications from Wi-Fi Alliance, which offers peer devices to wirelessly exchange file data. To avoid making changes in Sony Mobile Communications' current code base, some minor adjustments/modifications were necessary in the implementation of Application Service Platform.

The main characteristics of Wi-Fi Direct is that it is a peer-to-peer solution that does not limit the connection to a physical location. Wi-Fi Direct differs from Wi-Fi networks in both infrastructure and Ad hoc mode, although they're all under the same Wi-Fi family. It was not developed with the intention of replacing traditional Wi-Fi but to extend it, making it easier to share content among Wi-Fi devices by offering both flexibility and mobility.

The combination of high throughput, range, and mobility that Wi-Fi Direct offers makes it an unique peer-to-peer technology. When comparing with other wireless technologies that achieve the same tasks, we concluded that Bluetooth is the main competitor in the same field of application. Since Wi-Fi Direct is in many ways more robust compared with Bluetooth, we believe that when the certification of WFDS starts it will enhance the use of Wi-Fi Direct in these fields. NFC on the other hand is more likely to be used in combination with Wi-Fi Direct to increase the user experience. We believe that the launch of WFDS will enhance the use of Wi-Fi Direct. There already exists popular third party applications and brand specific solutions on the market that offers the same functionalities, which reflects that there is a demand for these type of services. The certification will extend the compatibility of Wi-Fi Direct use between different brand devices, simplifying many existing use cases and even create new ones.

Future Work

In our prototype, the services were chosen to be utilized through an android application. Future work should include integrating WFDS directly into the relevant default applications in the mobile devices. For example, the Send service should be integrated into the Album and Music application, allowing the user to initiate a file transfer when browsing through the media files.

Our performance results shows that it is not uncommon that service discovery time varies very much and can sometimes take up to 30 seconds to find the right

service. From a user's point of view, if the actual file transfer only takes two seconds to perform it would probably feel unacceptable that the service discovery might take multiple times longer.

Although it's not possible to use NFC Handover to connect corresponding services on two devices, NFC could still be used to minimize the service discovery time. By using NFC's "one-touch" motion to exchange service information, a user can immediately see which services the remote device supports and connect to the desired one. We estimate that this procedure should take approximately three seconds to perform, which guarantees a much quicker connection setup.

References

[1] Wi-Fi Alliance®. Technical Committee (2012). Wi-Fi Direct Services Task Group. *Wi-Fi Direct Services Draft Technical Specification Version 0.5*.

[2] Wi-Fi Alliance. *Who are We*. `http://www.wi-fi.org/who-we-are` [2014-05-17]

[3] Thornycroft, P. (2013). *Designed for Speed: Network Infrastructure in an 802.11n World*.

[4] Lee, B.G., Cho, S. (2008). *Broadband Wireless Access and Local Networks: Mobile WiMax and WiFi*.

[5] Sikdar, B. (2010). *Environmental Impact of IEEE 802.11 Access Points: A Case Study*.

[6] Frodigh, M., Johansson, P., Larsson, P. (2000). *Wireless Ad hoc networking - The art of networking without a network*.

[7] Feeney, L.M., Nilsson, M. (2001). *Investigating the Energy Consumption of a Wireless Network Interface in an Ad hoc Networking Environment* .

[8] Bluetooth Special Interest Group. *About the Bluetooth SIG*. `https://www.bluetooth.org/en-us/members/about-sig` [2014-05-22]

[9] Bluetooth SIG. (2014). *A Look at the Basics of Bluetooth Technology*. `http://www.bluetooth.com/Pages/Basics.aspx` [2014-04-16]

[10] Linsky, J. (2001).*Bluetooth and Power Consumption: Issues and Answers*.

[11] Kamath, S., Lindh, J. (2012) *Measuring Bluetooth Low Energy Power Consumption*.

[12] Bluetooth SIG. (2013) *Specification of the Bluetooth System ver 4.1*.

[13] NFC Forum. *Our Mission & Goals*. `http://nfc-forum.org/about-us/the-nfc-forum/` [2014-05-11]

[14] Ahson, S.A., Ilyas, M. (2011). *Near Field Communications Handbook*.

[15] NFC Forum. (2010). *Connection Handover, Technical Specification*

[16] Wi-Fi Alliance®. *FAQ*. `http://www.wi-fi.org/knowledge-center/faq/does-wi-fi-direct-work-on-80211-abgn` [2014-05-14]

[17] Wi-Fi Alliance®. *Discover Wi-Fi Direct Services.* http://www.wi-fi.org/discover-wi-fi/wi-fi-direct [2014-04-24]

[18] Wi-Fi Alliance®. *Wi-Fi Direct Certified Products.* http://www.wi-fi.org/certified-products-results?capabilities%5B50%5D=50 [2014-06-09]

[19] Forouzan, B.A. (2003). *Data Communications and Networking.*

[20] Camps-Mur, D. (2013). NEC Network Laboratories. *Device-To-Device Communications With WiFi Direct: Overview and Experimentation.*

[21] Hughes Systique™. (2013). *Wi-Fi Direct® White Paper.*

[22] Wi-Fi Alliance®. Technical Committee, P2P Task Group. (2011). *Wi-Fi Peer-to-Peer (P2P) Technical Specification Version 1.2.*

[23] Ranjani, H., Radhika, K., Sampath, V. (2014) *WFDS - A new Emerging technology over Wi-Fi Direct*

[24] Sony Mobile. (2013). *Sony Xperia™, Z1 C6903 White Paper* .

[25] Anritsu Shield Box. http://www.mcs-testequipment.com/product/anritsu-ma8120e-shield-box/#sthash.kCgHQhai.dpuf [2014-05-25]

[26] National Instruments Laboratory Virtual Instrumentation Engineering Workbench. http://www.ni.com/labview/ [2014-05-25]

[27] iPerf. *Network performance measurement.* http://iperf.fr [2014-05-25]

[28] Apple. *Bonjour for Developers.* https://developer.apple.com/bonjour/ [2014-05-28]

[29] Blank, A.G. (2004). *TCP/IP Foundations.*

[30] Network Working Group. (1999). *Hypertext Transfer Protocol – HTTP/1.1.*

[31] Network Working Group - Internet Engineering Task Force. (1996). *Multipurpose Internet Mail Extensions Part Two: Media Types.*

[32] Bluetooth SIG. *About Bluetooth Low Energy Technology.* http://www.bluetooth.com/Pages/low-energy-tech-info.aspx [2014-05-16]

[33] Android Beat. (2014). *NFC 101: What is Android Beam and how do you use it?.* http://www.androidbeat.com/2014/03/nfc-how-to-use-android-beam/ [2014-06-11]

[34] Samsung. (2013). *What is S Beam, and how do I use it on my Samsung Galaxy S III?.* http://www.samsung.com/us/support/supportOwnersHowToGuidePopup.do?howto_guide_seq=7042&prd_ia_cd=N0000003&map_seq=48157 [2014-06-11]

[35] SuperBeam. *Quick and Easy File Sharing.* http://superbe.am [2014-06-11]

[36] Nothing INC. *WiFi Shoot! WiFi Direct* https://play.google.com/store/apps/details?id=com.budius.WiFiShoot&hl=sv [2014-06-11]

[37] Apple. *Airplay.* http://www.apple.com/se/airplay/ [2014-06-11]

[38] Samsung. *Allshare*. http://content.samsung.com/us/contents/aboutn/allShareIntro.do [2014-06-11]

[39] Sony. *What kind of devices can be connected to use the mirroring function?*. https://us.en.kb.sony.com/app/answers/detail/a_id/43105/~/what-kind-of-devices-can-be-connected-to-use-the-mirroring-function%3F [2014-06-11]

[40] LG. *Smart Share*. http://www.lg.com/se/support/smart-share [2014-06-11]

[41] Samsung. *Smart Printing Solution*. http://www.samsung.com/global/smartprinting/# [2014-06-11]

LUND
UNIVERSITY

http://www.eit.lth.se