A Fast WiFi Direct Device Discovery with Caching Discovery Schedules for Seamless N-Screen Services

Bong-Jin Oh
ETRI
Daejeon, Korea
bjoh@etri.re.kr

Sunggeun Jin
Daegu University
Gyeongsan, Korea
sgjin@daegu.ac.kr

***Abstract*— We propose a novel device discovery scheme in order to accelerate device discovery procedures in the Wi-Fi direct networks. In the proposed scheme, neighboring devices capable of caching the scheduling information are utilized to provide the cached scheduling information while a device in their vicinity is in search of another device(s).**

Keywords - WiFi direct, device discovery, seamless services.

I. INTRODUCTION

Thanks to the proliferation of smart-phones, we are experiencing totally novel services which we have never seen before. Indeed, smart-phones are expected to fulfill more promising services by accommodating bleeding-edge technologies. Particularly, the N-Screen services providing multimedia services with multiple screens are attractive ones among the feasible services which the smart-phones can provide now.

Typically, the smart-phone screens are small enough to be held by a hand. In contrast, the TV screen sizes become wider while their prices are plunging more sharply. Obviously, such trend stimulates the desire to watch multimedia contents on a well-mounted wide screen, and hence the desire could be a sufficient reason why the multimedia contents on a smart-phone should be transferred and displayed on a wider TV screen.

Meanwhile, we can perceive fast evolution of the 802.11 Wireless Local Area Networks (WLANs). Recent advent of the 802.11ac makes it possible that smart-phones are serviced with high data rates of more than a few Gbps. Such high rates are sufficient to provide satisfactory High Fidelity (HD) multimedia services. Besides, the emerging Wi-Fi direct mounted on top of the 802.11 WLAN protocol stack enables smart-phones to transfer directly HD multimedia data to their connected devices without passing through an Access Point (AP) [1].

In fact, it is already realized that recent smart-phones can be used for the N-Screen services with the help of proper peripherals equipped with the WiFi interface. For instance, the Google announces the Chromecast for wireless services. The Chromecast establishes wireless connections between smart devices including smart-phones and a HDMI-enabled display such as TVs and monitors. The Chromecast receives multimedia streaming data through WiFi interface and forwards it to its connected display via the High-Definition Multimedia Interface (HDMI) while a smart-phone sends the user-selected multimedia contents to the Chromecast. In this way, a smart-phone user can enjoy the multimedia services on a wider screen [2]. The Digital Living Network Alliance (DLNA) technology is another way to enjoy N-Screen services. The smart-phones can display their own multimedia contents on a wide screen via DLNA connections. Recently, more and more Smart TVs and monitors tend to adopt the DLNA standard for better contents sharing services.[3].

As we can see from the examples of the Chromecast and DLNA technology, it is necessary to manage wireless connections and data transfers between smart-phones and displays for proper N-Screen services. The Wi-Fi direct is designed for this purpose. Various N-Screen related technologies are already built on the WiFi direct technology. It is expected that more sorts of the N-Screen services are realized by adopting the Wi-Fi direct in near future.

A device with Wi-Fi direct should conduct a device discovery procedure first during its initial connection establishment. The device discovery takes two or more seconds to be completed which may incur service disruptions while a smart-phone tries to change displaying screen [4]. Let us imagine the following service scenario. When a user watching a movie on a smart-phone wishes to transfer the movie to a nearby Smart TV. The user obviously wants to keep watching the movie without any disruptions. Then, the smart-phone tries to find a near Smart TV in order to migrate the streaming service display on the smart-phone's screen to a newly found Smart TV immediately after it detects user's intention. Given this case, it is strongly recommended that a device discovery time should be reduced as possible as it can since it heavily affects service disruption times.

Therefore, we design a fast device discovery scheme to reduce the service disruptions when displaying screen is changed for N-Screen services. We consider neighboring devices in the vicinity of the smart-phone providing an N-Screen service for a user since we are in a huge trend that many consumer electronics, which could be potentially neighboring devices, such as TVs, phones, audio components, and even refrigerators are equipped with Wi-Fi interface. Those consumer electronics may give a help to the device conducting device discovery procedures although they are not capable of displaying the multimedia

 ISBN: 978-1-61208-363-6

contents. In the proposed scheme, those neighboring devices cache the information regarding device discovery schedules for the devices conducting WiFi direct device discovery procedures. Then, they will inform another devices trying device discovery procedures of the cached scheduling information if needed.

This paper is organized as follows: Firstly, Section II describes background on the N-Screen services. Then we explain the proposed Wi-Fi device discovery scheme in Section III. In Section IV, we discuss expected performance evaluations of the proposed scheme since this work is still in progress. Finally, Section V concludes the paper.

II. BACKGROUND

Figure 1 shows a Remote User Interface (RUI) framework designed by the ETRI [5], [6].

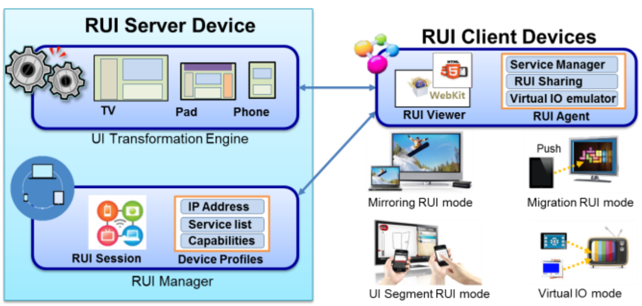

Figure 1. The structure and functionalities of ETRI's scalable RUI framework.

The RUI framework supports home networked smart devices to collaborate each other for the RUI services using migratable UIs. The UIs of RUI based application are implemented with the HTML5 language. They are composed of several UI segments, which can be moved to other devices according to user's requests. The core functionalities of the ETRI's RUI framework are summarized as follows:

- **UI transformation**: The UI transformation engine optimizes the UIs of RUI applications according to the device profiles including the information about resolution, screen size etc.
- **Mirroring RUI**: Users can share local UIs with multiple devices using duplicated UI documents. The duplicated UIs are synchronized according to invoked user events.
- **Migrating RUI**: When a user wants to display local UI on bigger screen than a local device, the user can push the local UI to a remote device. The local device can be used as input devices using I/O emulators.
- **Collaborative RUI**: A user can pull UI segments from a remote device to control remote RUI based services using their sub-UI displayed on the local screen.

Virtual I/O: A user can use local devices as virtual input devices to control remote devices or RUI services instead of a physical remote controller. The I/O emulators are embedded in the devices as Web applications composed of physical input device-like UI and functionalities. The emulators send invoked user events to a remote device through RUI framework, and the transmitted events are controlled by remote services.

The RUI framework is designed to work in the IP-based home networks. Therefore, all devices working on the same RUI framework should be connected to the same AP. However, the devices may not work correctly due to packet losses or delays while users move around at home. For example, an input event invoked by a virtual mouse often arrives too late to control remote services during the recovery procedures for packet losses caused by user's movement. In order to solve the problems, we are trying to expedite the operations regarding the RUI framework by revising the Wi-Fi direct protocols.

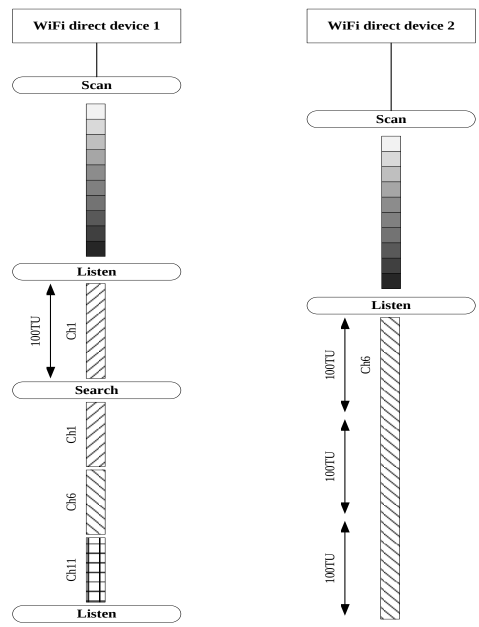

Figure 2. An Example of Wi-Fi Device Discovery Procedure.

The Wi-Fi direct announced by the Wi-Fi alliance is a protocol specification facilitating wireless Device-to-Device (D2D) communications in the 802.11 WLANs. It works on the top of Medium Access Control (MAC) layer of the 802.11 WLANs. Once a device with Wi-Fi direct is turned on, it begins to search another Wi-Fi direct device in order to establish a connection. For this purpose, the devices should follow device discovery procedure.

Figure 2 illustrates an example of the device discovery procedure. In this figure, *device-1* is first turned on, and then conducts the 802.11 active scanning. The active scanning is a typical 802.11 procedure to find new APs. An active scanning device sends *Probe Request* messages and waits for *Probe Response* messages for a while in each available channel in turn. Empirical measurements show that active scanning may take several hundred milliseconds so that extensive studies have been carried out to accelerate the active scanning procedure [7]-[14].

If the *device-1* fails to find a new AP with the active scanning, it enters device discovery procedure. Prior to the device discovery entrance, the device selects one channel as a home channel among three social channels, of which numbers are 1, 6, and 11. The device discovery procedure consists of two phases. The first phase is *Listen* phase. In *Listen* phase, Wi-Fi direct device camping on its home channel stays listening to *Probe Request* messages which another Wi-Fi direct device may transmit. The listening time can be arbitrarily chosen from one of three options, 100 Time Units (TUs, 1 TU is 1.024 milliseconds), 200 TUs, and 300 TUs.

If none of devices are found during the *Listen* phase, *device-1* begins *search* phase. In *search* phase, *device-1*sends *Probe Request* messages and waits for *Probe Response* messages. It repeats this procedure in each social channel in turn. In case when it cannot find a new device with this message exchanges, it tries *Listen* phase again. The device with device discovery procedure conducts this procedure until it finds a new device. Once a device succeeds in finding a new device, they form a network. When a Wi-Fi direct network is created, a group owner elected by comparing devices' intent value takes the role of an AP.

III. PROPOSED DEVICE DISCOVERY

A. Operation Scenario for N-Screen Services

Figure 3 shows an example for our scenario regarding the operation while a user with a smartphone moves from the TV at location A toward the TV at location B. For simplicity, we call the TV at location A and B as TV-A and TV-B, respectively. In this scenario, the smartphone is initially connected with the TV-A via Wi-Fi direct. It sends multimedia traffic for the streaming service so that the user with the smartphone can watch the TV enjoying multimedia service. Then, the user moves toward the TV-B while he/she is still watching the TV-A. During user's movement, the Wi-Fi direct connection may be lost due to weak wireless signal strength, and then the smartphone should search a new TV for continuous multimedia service.

Figure 3. An Exemplary Service Scenario.

There can be one of two possible cases to cope with this situation. (1) The service on the TV-A seamless migrates to the TV-B when the smartphone successfully finds the TV-B. The smartphone should control overall procedure for the seamless migration. (2) Otherwise, the service should be provided in the smartphone temporarily until the smartphone finds a new TV. If it succeeds in finding a new TV, the service is switched to the newly found TV, e.g., the TV-B in this service scenario. If not, the service continues to be serviced in the smartphone.

In either case, the smartphone should try to find a new TV for continuous multimedia service so that the device discovery procedure should be done as early as possible. However, as reported in [4], the Wi-Fi direct device discovery may take longer than two or three seconds to be completed. Such a long device discovery procedure may disrupt the multimedia service thus deteriorating service quality seriously. For this reason, we propose more efficient device discovery procedure depending on the existence of neighboring devices capable of taking over ongoing services in the subsequent subsections.

B. Device Discovery with Neighboring Devices

We assume that the TV-B does not have any connection with other device. Therefore, the TV-B periodically conducts device discovery procedure.

Typically, the device conducting Wi-Fi direct device discovery broadcasts *Probe Request* messages and thereafter waits for *Probe Response* messages for a while in a social channel. Neighboring devices receiving the *Probe Request* messages reply with *Probe Response* message when they can provide the services which the searching device wants to be served.

Additionally, we add scheduling information to the *Probe Request* message. The scheduling information includes the time when the device revisits to the social channel in near future. Neighboring devices should cache the information even though the neighboring devices are not what the device conducting device discovery wants to find. It implies that the device caching the scheduling information does not need to reply if the Probe Request messages are not relevant to its capability. The cached information is conveyed to another device if what the device is searching matches the cached information. Then, the device can utilize the information to find a new device in its vicinity.

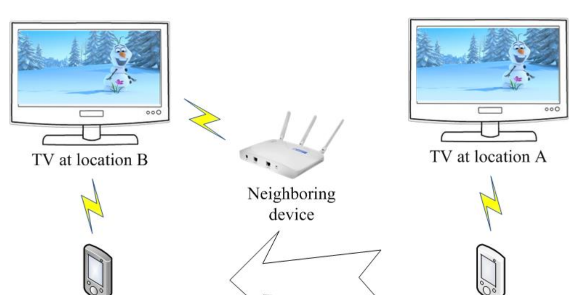

Figure 4. An Example for the Proposed Device Discovery with the Help of a Neighboring Device.

Figure 4 shows an example for this proposed procedure. In this figure, the AP neighboring to TV-B caches the scheduling information of the TV-B. The smartphone begins to move toward TV-B. However, it cannot find the

TV-B since the TV-B is far from it. Instead, the smartphone can obtain the scheduling information from the AP in advance. Therefore, the smartphone can wait for the TV-B in the social channel by referring to the scheduling information in the AP's vicinity. For better cache hit ratio, the AP can estimate the smartphone's movement so that the AP sends replies for only case when the smartphone approaches to it.

C. Device Discovery without Neighboring Devices

It is possible that there are none of neighboring devices, which can help the smartphone. In order to resolve the situation, we consider two candidates as follows:

(1) As explained earlier, each device has its own home channel. From this fact, we design the device to have its own time schedule, in which the device should stay definitely at the channel. The time schedule should repeat so that another device can estimate the future schedule with the current schedule. In the service scenario shown in Fig. 3, a smartphone may happen to pass by the TV-B and the TV-A in order. Therefore, the smartphone can cache the scheduling information of the TV-B, and then transfer it to the TV-A. In this way, the scheduling information may propagate to the TV-A. Later, a new smartphone may approach to the TV-A. Then, the TV-A gives the scheduling information of the TV-B to the smartphone. For this reason, the smartphone can utilize the information to predict future scheduling information and try to find a new device, i.e., TV-B with the scheduling information.

(2) We consider globally predefined time schedule. By using this schedule, devices should stay at a globally defined home channel for a given period. However, this scheme need to be polished more since it has many issues to be solved such as global synchronization, device discovery latency etc. Consequently, two proposed schemes still need to be refined for better efficiency.

IV. Expected Performance Evaluation

The proposed device discovery procedure is under development with the Odroid XU boards [15]. The boards are equipped with the Wi-Fi module manufactured by Realtek [16]. For our service scenario, we assume that TVs and smartphones run in the Android environments. Therefore, we utilize the Wi-Fi direct source code in the Android devices for the practical system developments in both sides of the TV as well as the smartphone.

We plan to conduct experiments with complete implementation of the proposed schemes. For our experiments, we will employ several neighboring devices and measure the lost packets during the proposed device discovery procedure and the existing device discovery procedure, respectively. After that, we will show the comparison results.

V. Conclusion

We propose a feasible device discovery scheme accelerating Wi-Fi direct device discovery procedure. The proposed scheme will be evaluated with practical implementations. We expect that our work is certainly helpful for the work regarding N-Screen services since these services become prevalent with Wi-Fi direct in near future.

Acknowledgments

This work was supported by the IT R&D program of MKE/KEIT, [I2905-13-1018, Smart TV platform built using a multi-device collaboration gateway].

References

[1] WiFi Peer-to-Peer Technical Specification Version 1.1, October 2010.

[2] https://www.google.co.kr/intl/en/chrome/devices/chromecast/apps.html.2014.05.30

[3] www.dlna.org. 2014.05.30

[4] S. Jin and B. Oh "An Analytical Study on the WiFi-Direct Device Discovery," Proc. ICCCS'13, November 2013, pp. 136-137.

[5] Y. Bae, B. Oh, and J. Park, "Adaptive Transformation for A Scalable User Interface Framework Supporting Multi-Screen Services," Proc. CUTE'13, Dec. 2013, pp.425-432.

[6] B. Oh and J. Park, "A Remote User Interface Framework for Collaborative Services Using Interconnected Smart Devices," Proc. ICACT2014, Feb. 2014, pp. 630-634.

[7] S. Kim, S. Choi, S. Park, J. Lee, and S. Kim, "An Empirical Measurements-based Analysis of Public WLAN Handoff Operations," Proc. WILLOPAN'06, Jan. 2006, pp. 1-6.

[8] A. Mishra, M. Shin, and W. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," ACM SIGCOMM CCR, vol. 33, no. 2, pp. 93-102, Apr. 2003.

[9] S. Jin, M. Choi, L. Wang, and S. Choi, "Fast Scanning Schemes for IEEE 802.11 WLANs in Virtual AP Environments," Elsevier ComNet, vol. 55, no. 10, pp. 2520-2533, July 2011

[10] S. Jin, Munhwan C., and S. Choi, "Multiple WNIC-Based Handoff in IEEE 802.11 WLANs," IEEE CL, vol. 13, no. 10, pp. 752-754, Oct. 2009.

[11] S. Jin and S. Choi, "A Seamless Handoff with Multiple Radios in IEEE 802.11 WLANs," IEEE TVT, vol. 63, no. 3, pp. 1408-1418, 13 Mar. 2014.

[12] K. Ramachandran, S. Rangarajan, and C. J. Lin, "Make-before-break Mac Layer Handoff in 802.11 Wireless Networks," Proc. IEEE ICC'06, June 2006, pp. 4818-4823.

[13] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, "Proactive Scan: Fast Handoff with Smart Triggers for 802.11 Wireless LAN," Proc. IEEE INFOCOM'07, May 2007, pp. 749-757.

[14] Y. S. Chen, M. Chuang, and C. Chen, "Deucescan: Deuce-based Fast Handover Scheme in IEEE 802.11 Wireless Networks," IEEE TVT, vol. 57, no. 2, pp. 1126-1141, Mar. 2008.

[15] http://www.hardkernel.com/main/products/prdt_info.php?g_code=G137510300620. 2014.05.30

[16] http://www.hardkernel.com/main/products/prdt_info.php?g_code=G137447734369. 2014.05.30