Wi-Fi helping out Bluetooth Smart for an improved home automation user experience

Jen Rossey, Ingrid Moerman, Piet Demeester, Jeroen Hoebeke
Ghent University – iMinds, Department of Information Technology (INTEC)
Technologiepark-Zwijnaarde 15, 9052 Ghent, Belgium
{firstname.lastname}@intec.ugent.be

***Abstract*— Home automation devices are becoming increasingly popular in the field of consumer electronics. Various appliances like thermostats, smoke detectors, intelligent lighting systems, etc., have appeared on the market to create a smart home. Vendors have the availability over multiple wireless technologies to connect their products to the smart home and communicate with the user. The most adopted technologies are the ones that can interface directly with a mobile device such as a smartphone or tablet, without the need for an additional gateway. Within this context, Wi-Fi and Bluetooth are the dominant technologies. In this paper we look at home automation devices that have chosen to solely support Bluetooth 4.0 as communication interface. We highlight the downsides of this technology in a home setting and try to mitigate this problem by exploiting the Wi-Fi capabilities of other devices, in particular smartphones. The proposed solution realizes a Wi-Fi bridge on the smartphone that is connected to the Bluetooth device. This enables other smartphone users to connect to the Bluetooth device over the Wi-Fi network, alleviating some of the downsides of the Bluetooth technology.**

Keywords*— *Building automation; Bluetooth Smart; Bluetooth Low Energy; Wi-Fi; Smartphone; Internet of Things

I. INTRODUCTION

Due to the advances in electronics, miniaturization, wireless communication, batteries, etc., more and more devices are being connected to the Internet, resulting in what is being called the Internet of Things (IoT). By 2020, a projected 30 billion devices will enter the IoT ecosystem by 2020 [1]. A major IoT application domain is home or building automation. Within this domain, the connected devices will enable you to control the lights, temperature, household appliances, window and door locks and security systems. With all these possibilities, your home will become a truly smart home.

Many of these devices will make use of wireless communication technologies in order to realize their interconnection to the Internet. A wide range of wireless communication technologies exist, such as IEEE 802.11, IEEE 802.15.4, Bluetooth, Z-Wave, etc. One of the key factors for vendors to select a particular wireless technology is the ability for users to directly interact with their connected devices from their smartphones and tablets, without having to purchase an additional gateway. Looking at the capabilities of current smartphones, this reduces the number of candidate technologies strongly, leaving only Wi-Fi and Bluetooth.

Bluetooth has always been a popular technology to be used in combination with mobile devices such as phones. In the past Bluetooth was mainly used for wireless transmission of audio and direct communication between phones and computers. With the introduction of the power-friendly version Bluetooth Smart or Bluetooth Low Energy (BLE) it has opened up a multitude of possibilities to create Internet of Things (IoT) devices that can directly connect to your smartphone [2] and that can run on batteries for several months or even years. In this paper we will focus on Bluetooth Smart devices and the implications of choosing Bluetooth Smart on the interactions with the user

The design of Bluetooth Smart has implications on the interactions with users, especially in settings where multiple users are present, such as a smart home. Only one user is able to connect to a device at the same time, complicating multi-user interactions. To alleviate this problem, we propose to exploit the availability of Wi-Fi connectivity on the users' devices, in order to facilitate multi-user multi-device interactions in a seamless way. To our knowledge, this is the first work that explores this possibility and presents concrete performance measurements. It shows that a combined solution, where Wi-Fi is helping out Bluetooth Smart can contribute to an enhanced user experience.

The remainder of this article is organized as follows. In Section 2, we first discuss some key features of Bluetooth Smart and explain their implications on the interactions with users. Next, in section 3, we present our approach to enable Wi-Fi assisted interactions from different smartphones with a single Bluetooth Smart device. In Section 4, we evaluate the performance of our design. In section 5 we look at related work. Finally, we conclude this paper in Section 6.

II. USER INTERACTIONS WITH BLUETOOTH SMART DEVICES

A. Bluetooth Smart

Bluetooth Smart or Bluetooth Low Energy is the power-friendly, low cost version of the well-known Bluetooth technology. Bluetooth Smart is used in power-constrained devices such as wireless sensors and controls. These kinds of devices have limited data transmission and communication happens infrequently. This is different from classic Bluetooth applications like audio streaming. Bluetooth Smart utilizes 40 channels of 2 MHz in the 2.4GHz ISM band, 37 channels are for data transfer and 3 channels are for advertising. Bluetooth smart uses TDMA and frequency hopping to limit interference with other wireless technologies.

There are two main network topologies that are used with Bluetooth Smart. The first one, shown in Figure 1, is the

978-1-5090-4361-3/16/$31.00 ©2016 European Union

broadcast topology where one device broadcasts advertisement messages to all the observers that are listening. In this topology, only one-way communication is possible from the broadcaster to the observer. This topology is used when applications use beacons for localization or advertisement purposes.

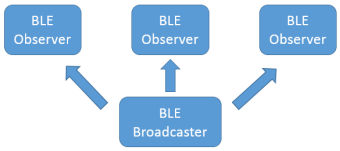

Figure 1: Broadcast topology

The second topology, shown in Figure 2, is the connections topology where one central device can connect to one or multiple peripheral device. With connections, there is two-way communication between the central and the peripheral device. A typical central device is a smartphone or tablet.

Figure 2: Connections topology

A smart home setting with peripheral devices can be seen in Figure 3. There are multiple users that have a smartphone (central) and there are multiple smart home devices (peripheral) like a thermostat, smart lightning, etc. Next to the Bluetooth devices, there is a Wi-Fi network where all the smartphones are connected to so they can access the Internet. This Wi-Fi network can also be used to establish communication between the smartphones.

Figure 3: Smart home with multiple users

B. *Bluetooth Smart problems*

One of the problems with the most widespread version of Bluetooth Smart (4.0) is that a peripheral device can only connect to one central at the same time. This means that only one user can connect to the smart home device at the same time, as shown in Figure 4. If a different user wants to connect to the device, it will not discover the device because it will not be sending advertisement beacons to make itself discoverable. This is not desirable as users will not be able to interact with the smart home device, while they think they are in range.

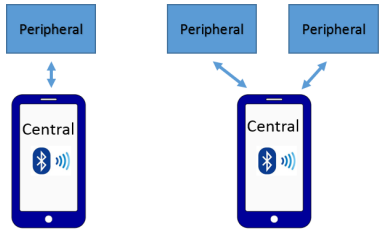

Figure 4: Peripheral can only connect to one central

III. Combining Bluetooth and Wi-Fi

The network context of this paper will be a smart home environment where there is one Wi-Fi network and multiple users that are connected to this network with their smartphone Figure 3. There is at least one Bluetooth enabled smart home device (heating system, lights, etc.).

A. *Connecting to a peripheral through the Wi-Fi network*

When no smartphone is connected to the smart home device (peripheral), the peripheral is broadcasting connectable advertisement packets. When a smartphone (central) receives one of these advertisement packets, it knows the peripheral is in range and is able to connect to it. When a peripheral is connected to a central, it stops sending these advertisement packets, as it can only connect to one central. If a second central wants to discover the peripheral, it will have no way of knowing that the peripheral is in range or that it is occupied by a different smartphone.

Figure 5 shows the general architecture of our solution, the client smartphone can communicate with the peripheral, while a different smartphone (bridge) already has an active Bluetooth Smart connection to the peripheral.

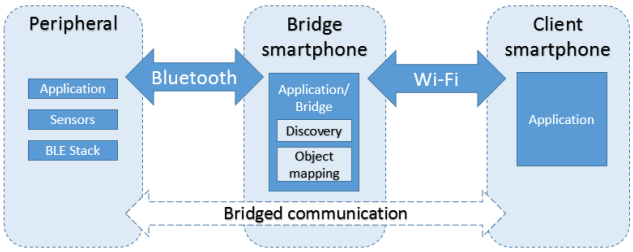

Figure 5: Interaction between peripheral, bridge and client smartphone

Using our proposed solution, the smartphone that connects to the peripheral will start a network service on the connected Wi-Fi network which tells other smartphones on the same network that it is currently connected to a specific peripheral. This network service is the Network Service Discovery (NSD) that is available on Android smartphones [3]. We focussed on android devices, but a similar multicast DNS service is also available on Apple devices with the Bonjour protocol [4]. Both of these services are based on the multicast DNS [5]. The name of the NSD service will be a service identifier, the hardware address and the device name of the connected peripheral.

When other smartphones scan for Bluetooth devices, they also check if there are any NSD services with the correct service identifier available on the network that show that a different smartphone is already connected to a peripheral. By checking the advertised name of the NSD service, the smartphone immediately knows the address and name of the peripheral. Depending on the type of smartphone application, this device can be shown to the user as available.

Figure 6 shows a discovered NSD service with the name of the service including the service identifier 'bt-wifi' to specify that this service shows the connected Bluetooth device, the hardware address and the name of the peripheral. It also shows the IP address of the smartphone that is connected to the peripheral and the port that is accepting incoming TCP connections.

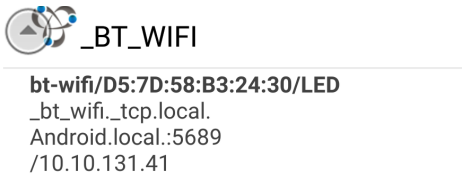

Figure 6: Discovered NSD service

In Figure 7 a screenshot is shown of an example application that lists all discovered Bluetooth devices. In Figure 8, the same application shows how it has discovered the same peripheral, but now through NSD.

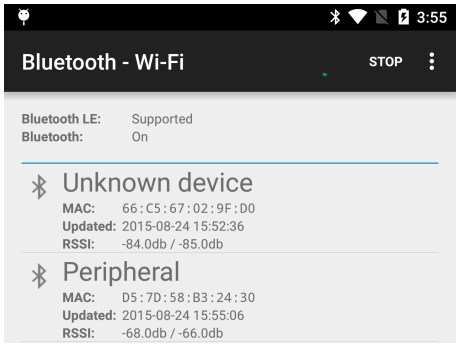

Figure 7: Discovered Bluetooth devices

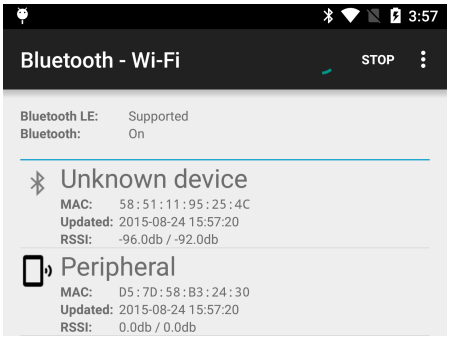

Figure 8: Discovered Peripheral through NSD

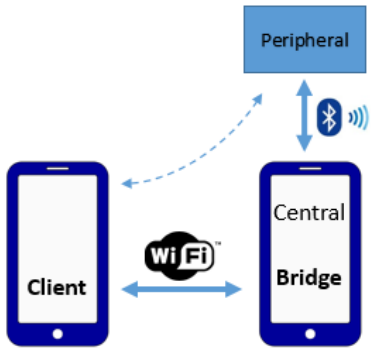

Figure 9: Connecting to peripheral through other smartphone

This way, a client smartphone can 'connect' to a peripheral that is already connected to a different smartphone. When a smartphone (client) wants to connect to a specific peripheral, and it finds a NSD service with the address of the peripheral, it can 'connect' to this peripheral using the connected smartphone as a bridge as illustrated in Figure 9.

B. Communicating to the peripheral

The connection between the client smartphone and the peripheral is not a Bluetooth connection but a TCP socket that is set up between the client and bridge smartphone over the Wi-Fi network. This connection is set up when the client smartphone wants to connect to the peripheral. When the bridge receives an incoming TCP connection from a client smartphone, it sends the list of resources that are supported on the peripheral device. The communication between the client and bridge smartphone is done using object serialization over the socket. These objects contain all the relevant data that is typically sent between the central and peripheral device.

When the client smartphone wants to query a specific resource, it sends this query to the bridge and the bridge then relays the request to the peripheral. When the bridge receives a reply from the peripheral, this is then sent back to the client. The upper layers of the client application have no idea that the smartphone is not directly connected to the peripheral, as it still has access to all the relevant data it would have access too if it was directly connected.

C. Disconnecting or losing the connection

There are multiple ways that the bridge smartphone can lose the connection to the peripheral. The bridge smartphone can turn off, manually disconnect, turn the Bluetooth off, go out of range, etc. When this happens, the client smartphone is no longer able to communicate with the peripheral through the bridge. The bridge notifies the client by sending a disconnect message to the client when it is aware that it lost the connection to the peripheral. It is however also possible that the bridge can't notify the client of a disconnection, this can happen if airplane mode is activated, if the bridge suddenly stops working, if the bridge goes out of range of the Wi-Fi network, etc. In this case the client has to automatically detect that the socket to the bridge is no longer working. This is done using heartbeat packets and is further explained in Section 4.C.

When the bridge smartphone can no longer function as a bridge for whatever reason, the client smartphone will try to connect to the peripheral directly and resume normal Bluetooth connectivity. If the client cannot find the peripheral, it will again search for NSD services because a different smartphone can already be connected to the peripheral and function as a new bridge. Finally, in case the client smartphone cannot reconnect to the peripheral, the user will be notified that the smartphone is out of range to the peripheral, or that the peripheral is offline.

IV. Evaluation

When a bridge is used to connect to a peripheral, there will be an added delay for the communication between the client and the peripheral. We tested the delay in different steps of the communication with a Google Nexus 6 running Android 5.1.1 that functions as the bridge and a OnePlus One running Android 5.0.2 as the client. The bridge is connected to a Nordic nRF51 development kit, shown in Figure 10, that runs as a peripheral device simulating a smart home lighting control system. The nRF51 has basic resources for the device name, serial number, etc. It also has a resource that enables to read and change the status of the LEDs and a generic resource that returns a big payload to test performance.

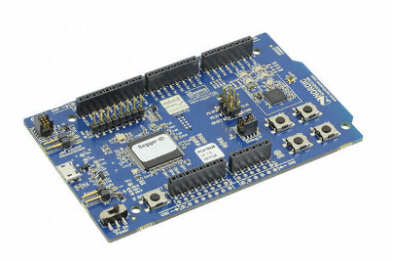

Figure 10: Nordic nRF51 DK as a smart home lighting peripheral

A. Discovery

The first step for the client smartphone to communicate with the peripheral while it is connected to a bridge smartphone is discovering the NSD service. We connected the bridge smartphone to the peripheral and let the client smartphone scan for NSD services. We measured the time it takes from beginning the scan to finding a NSD service. The results of this test can be found in Table 1.

Table 1: NSD discovery measurements

Number of tests	50
Mean discovery time	8.1ms
Standard deviation	5ms
Min discovery time	3ms
Max discovery time	19

After discovering a NSD service, the service name of the found service is compared to the bt_wifi service name, because other NSD services can also be active on the network. After the correct NSD service is discovered, the connection information for the service has to be resolved. The connection information contains the IP address of the bridge smartphone and the port that the bridge can receive incoming connections on.

We measured the time it takes to resolve the NSD service after discovery. The results can be found in Table 2.

Table 2: NSD resolve measurements

Number of tests	50
Mean resolve time	41ms
Standard deviation	82ms
Min resolve time	3ms
Max resolve time	355ms

If we add the time it takes to discover and resolve the correct NSD service, we have the total time before we can connect the client smartphone to the bridge smartphone. You can find the results of this in Table 3. The big difference between the minimum and the maximum resolve time is most likely caused by heavy usage of the used Wi-Fi network.

Table 3: Combined discovery and resolve measurements

Number of tests	50
Mean resolve time	49ms
Standard deviation	82ms
Min resolve time	7ms
Max resolve time	361ms

The average discovery time of a Bluetooth Smart peripheral varies based on the advertisement interval. A peripheral sends advertisement packets to let other devices discover the peripheral. This interval ranges from 20ms to 10.24s [6] and this affects the power consumption of the peripheral, the more radio activity on the device, the greater the consumption.

If the advertisement interval is set to the minimum (20ms), the discovery of a Bluetooth device takes on average half the time of the discovery of a NSD service. Most Bluetooth Smart applications don't use the minimum advertisement interval as this has a big impact on the power consumption. The difference between the NSD discovery time and the minimum Bluetooth Smart discovery time is negligible for a user interacting with an application on the client smartphone. If the peripheral is set with a very high interval (e.g. 10.24s), the discovery time with the NSD service is a lot faster.

B. Connection and communication

We define the setup of a connection when there is a TCP connection between the client and the bridge smartphone and when the bridge has sent the available GATT resources on the peripheral to the client. The time needed to setup such a connection between the client and bridge smartphone is very low as shown in Table 4.

Table 4: Client – Bridge connection time measurements

Number of tests	50
Mean connection time	15ms
Standard deviation	5ms
Min connection time	8ms
Max connection time	31ms

Table 5 shows the time for a smartphone to connect to a peripheral that has a connection advertisement interval of 50ms. It is very clear that a connection over Wi-Fi is a lot faster than over Bluetooth. When a client wants to connect to a peripheral

over a bridge smartphone, it is significantly faster than directly connecting to the peripheral.

Table 5: Peripheral connection time measurements

Number of tests	10
Mean connection time	2.83s
Standard deviation	0.2s
Min connection time	2.54s
Max connection time	3.16s

When a connection is made between the client and the bridge smartphone, the client can communicate with the peripheral. The extra delay introduced by the communication between the bridge and client is not noticeable for the end user. This extra delay is almost never longer than 30ms.

C. Handover

There are several scenarios where a handover is necessary, implying that he client smartphone has to terminate the connection to the bridge and has to start a direct Bluetooth connection to the peripheral. These scenarios can be split up in controlled and uncontrolled handovers. A controlled handover happens when the bridge notifies the client that is has disconnected from the peripheral, this way the client immediately knows that there is no longer an indirect connection with the peripheral and that he has to try to either connect directly over Bluetooth or via a different bridge smartphone that is now connected to the peripheral. In the following paragraph, we assume that the client is in range of the peripheral. If the client is not in range of the peripheral, a handover using a direct Bluetooth connection will not be possible.

The average time it takes the bridge to disconnect from the peripheral is 40ms. The bridge notifies the client right before it is going to disconnect from the peripheral, which takes less than 40ms. This can be done in parallel with the disconnection from the peripheral so we can ignore this. Table 5 shows the connection time for a smartphone to connect to a peripheral. When we combine the disconnection time of the bridge and the connection time of the client to the peripheral, we can conclude that the handover time will always be less than 3s.

There are multiple cases where the bridge will not notify the client when disconnecting from the peripheral (airplane mode, out of Wi-Fi range, etc.). When this happens, the client has to detect that the TCP connection with the bridge has been terminated. The TCP protocol does not support any fast means of detecting a broken connection. Therefore, we implemented a heartbeat packet that is sent every 100ms between the client and the bridge if there is no other communication. If the client does not receive a reply on the heartbeat packet after 250ms, it assumes the bridge has gone offline and will try to repair the connection to the peripheral by either directly connecting to the peripheral over Bluetooth or by finding a different NSD service representing the peripheral.

V. Related work

Other work also focusses on the problem of gateway devices and proposes an architecture that leverages the ubiquitous presence of Bluetooth Smart to connect IoT peripherals to the Internet [7]. In this architecture, a smartphone device functions as a bridge device between the peripheral and the Internet. The difference with our work is that we implemented a working communication and discovery solution. Another paper that bridges Bluetooth connectivity uses a bridge architecture to enable web applications that can use the Bluetooth communication module of the client device [8]. This way, web applications can manage the Smartphone's Bluetooth communication module and use information from nearby electronic devices. Our solution is different and focuses on other end users and smartphones that exploit Wi-Fi connectivity to access peripherals.

A lot of the related work has focused on the coexistence between Wi-Fi and Bluetooth and the impact on interference [9]. Next to this, there are commercial solutions available that create a 'Bluetooth to Wi-Fi' bridge service [10]. This is a different solution for the problems described in this paper. However, such a solution requires a dedicated gateway device, something we wanted to avoid.

There is also work that focusses on gateway architectures to enable communication between devices that use different communication protocols [11]. Such solutions do not implement an actual solution to communicate between Wi-Fi and Bluetooth devices and is again reliant on a gateway device. Other work is focussing on combining the strong points of Wi-Fi and Bluetooth. For instance, [12] uses the high-speed data transmission rate of Wi-Fi P2P and the low power consumption communication of Bluetooth, whereas [13] looks into using Bluetooth Smart to start an initial connection before using Wi-Fi P2P to have a fast data transmission. However, these approaches tackle a different problem.

VI. Conclusions

BLE is popular technology for connected IoT devices. Nevertheless, it has some drawbacks in settings where multiple users need to interact with the same BLE device. When one user is already connected to a BLE device, another user within the same range is not able to communicate with the very same device. In case the other user is out of range, no interaction with the device is possible at all. In this paper we have shown that the collaboration between different wireless technologies, in this case Wi-Fi and BLE, can mitigate the identified problems. We have presented a design where Wi-Fi helps out BLE, by acting as a bridging technology. Our experiments show that our approach is fast enough to allow for seamless communication between a peripheral device, and a user that connects to this device with his own smartphone through a bridge smartphone.

As such this paper illustrates that a converged approach, where higher-layer protocols are designed in such a way that they can operate beyond a single lower-level technology, can improve the way users interact with their IoT devices.

Acknowledgment

The research leading to these results has been carried out within the ITEA2 FUSE-IT project (13023) and has received funding from the agency for Innovation by Science and Technology (IWT).

REFERENCES

[1] ABI Research. (2013, May 09). *More Than 30 Billion Devices Will Wirelessly Connect to the Internet of Everything in 2020*. Retrieved from ABI Research: https://www.abiresearch.com/press/more-than-30-billion-devices-will-wirelessly-conne/

[2] Bluetooth SIG, Inc. (2015). *Bluetooth Smart Technology: Powering the Internet of Things*. Retrieved from http://www.bluetooth.com/Pages/Bluetooth-Smart.aspx

[3] Google. (n.d.). *Using Network Service Discovery*. Retrieved from Android Developers: http://developer.android.com/training/connect-devices-wirelessly/nsd.html

[4] Apple. (2015). *Bonjour for Developers*. Retrieved from https://developer.apple.com/bonjour/

[5] Cheshire, S., Krochmal, M., & Inc., A. (2013, February). *Multicast DNS*. Retrieved from The Internet Engineering Task Force : https://tools.ietf.org/html/rfc6762

[6] Townsend, K., Cufí, C., Akiba, & Davidson, R. (2014). *Getting started with Bluetooth Low Energy*. O'Reilly Media.

[7] Zachariah, T., Klugman, N., Campbell, B., Adkins, J., Jackson, N., & Dutta, P. (2015). The Internet of Things Has a Gateway Problem. *HotMobile '15 Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. Ann Arbor: ACM New York.

[8] Espada, J. P., Díaz, V. G., Crespo, R. G., Martínez, O. S., G-Bustelo, B. P., & Lovelle, J. M. (2015). Using extended web technologies to develop Bluetooth multi-platform. *Information Fusion*.

[9] Silva, S., Fernandes, T., & Moreira, A. V. (2014). Coexistence and Interference Tests on a Bluetooth. *Science and Information Conference* . London.

[10] bluvision. (2015). *BluFI Bluetooth-to-WiFi Sensor*. Retrieved from bluvision: http://bluvision.com/blufi-wifi-sensor

[11] Starsinic, M. (2010). System Architecture Challenges in the Home M2M. *InterDigital Communications*.

[12] Joh, H., & Ryoo, I. (2015). A hybrid Wi-Fi P2P with bluetooth low energy for optimizing. *Peer-to-Peer Networking and Applications*.

[13] Joh, H., Yang, I., & Ryoo, I. (2015). The internet of everything based on energy efficient P2P transmission technology with Bluetooth low energy. *Peer-to-Peer Networking and Applications*