Comparison of Energy Consumption in Wi-Fi and Bluetooth Communication: A Case Study on Context Aware Building

Guntur Dharma Putra

Abstract— Context awareness has been an interesting topic recently. Its ability to infer whether a person exists on a particular room or building is really important for smart building. The result shows that Bluetooth is 29.97% more energy efficient than WiFi.

Index Terms—Context-aware, smart building, wi-fi, bluetooth low energy

1 Introduction

Smart building has been an interesting topic of research recently. One portion of research in smart home is occupancy detection, which aims to detect whether a person is present in a particular location. Its importance to detect user presence is crucial in the building energy management, since the building can manage energy allocation efficiently regarding how many persons are present.

Several methods have been proposed to overcome the occupancy detection. One of them makes use of Bluetooth Low Energy (BLE) beacon, as the beacon is useful because it always transmitting a unique data packet that indicates certain location information. Assuming that the users always bring mobile phone with them, an application can be installed on the mobile phone to scan a particular beacon, so that the application knows where currently the user is, then the application sends the data to the central server. The data can be analyzed later on to detect user presence.

Normally, the application sends the data to the server through HTTP communication done via Wi-Fi connectivity, as the BLE is already used to sense the beacon. No BLE utilization for data transmission to the server has been found. In fact, BLE is obviously more energy efficient compared to WiFi, as BLE is designed to be implemented in devices coupled with limited source of energy, e.g., battery.

This study tries to investigate BLE utilization for transmitting the occupancy data to the server. This study measures and compares the energy consumption of the mobile phone when performing data transmission via WiFif and BLE. A tailored application is developed and several possible scenario is also taken into consideration, such as number of detected sensor and user location relative to the server or access point. The result of this study may be useful for the future decision whether BLE will be implemented instead of WiFi to transmit occupancy data to the server.

The rest of this report is structured as follows. Section 2 presents other related work to this study. Methodology is described in section 3, while the results and discussion is discussed in section 4. Lastly, a conclusion is drawn in section 5

2 Related Work

cite all the related work properly that you base your own work on

discuss why it is relevant and what is similar or different to your own work use images (from other papers) to illustrate the related work, credit the authors with a reference in the image caption

give details for each publication (authors, title, year, page numbers, publisher, publisher address (town); for articles volume and number and month; for things other than books, articles, or papers also the type of publication)

• *Guntur Dharma Putra is an MSc Student in Computing Science at the Universtiy of Groningen. E-mail: g.d.putra@student.rug.nl.*

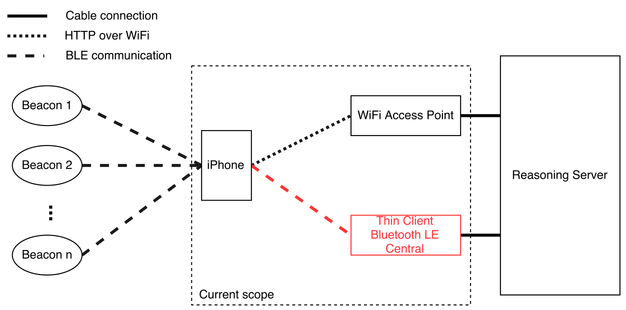


Fig. 1: System architecture overview.

3 Methodology

As a term project, this study was performed in three consecutive months. The main part of this study, the energy measurement, was carried out in iPhone 6, which runs iOS operating system. Additionally, Asus vivo mini PC, which runs Xubuntu as its operating system, was also utilized as a thin client that hosts the server application.

3.1 System Architecture

This study is based on a study from Azkario, which attempts to extract occupancy data in smart building. This system consist of BLE beacons, which are placed accordingly in each room, a user's mobile phone, and a sensing infrastructure, which consists of WiFi infrastructure and sensing server. Currently, the user's mobile phone uses HTTP over WiFi to send occupancy data to the server, while Bluetooth is only used in mobile phone to or from BLE beacons. The Bluetooth communication in this study is intended to replace the mobile phone to server that is currently implemented using HTTP over WiFi. Briefly, the system architecture is depicted in Figure 1.

As seen in Figure 1, the thin client (colored in red), which acts as BLE central, is added to the Azkario's architecture. Later, energy consumption between BLE and WiFi are measured and compared. Although the result may be obvious that BLE is more energy efficient than HTTP over WiFi communication, this study aims to figure out how efficient is BLE communication compared to HTTP over WiFi in this context.

This study only focuses on mobile phone (iPhone) to WiFi Access Point (for HTTP over WiFi) or Thin Client (for BLE communication) for energy consumption measurement, which is surrounded in dashed box. The sensing part of the whole architecture, which is from beacons to iPhone, is neglected. As a consequences, dummy data that imitates the real data from beacons is used, which is sent from the mobile phone in every second.

3.2 Measuring the Energy Consumption

Energy consumption measurement or tracing is the core of this study. It is done by using untethered energy logging in iOS, in which the

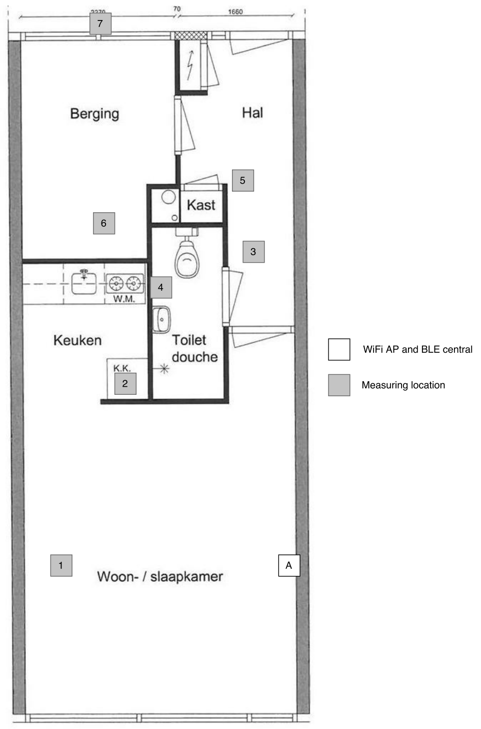


Fig. 2: iPhone placement map.

energy consumption data is logged internally in the device itself before imported to the computer by using cable connection. This method is selected because it has the flexibility over the other methods. Wireless logging, which does not involve internal logging in the device, is not used as it requires Bonjour enabled router that was unavailable.

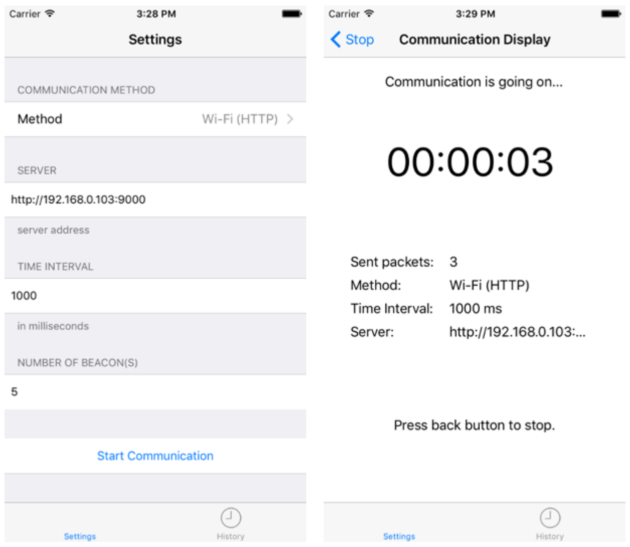
Apple's Instrument application in Mac OS X was used to import the logged energy measurement in iPhone, as done in [4]. It does not show the measurement result in standard energy measurement format, e.g., mAh, but it shows that in its own format, which is scaled from 0 to 20. Each increment in the scale costs an hour of battery life [5]. Thus, phone running at level 1/20 will have 20 hours of battery life, while phone running at level 20/20 will have only 1 hour of battery life.

Furthermore, export feature is limited in Apple's Instrument, i.e., it does not support energy consumption log exporting to other commonly used format, e.g., csv or xls file. However, manual copying and pasting on each row is still supported. An Apple script was used to automate the copying and pasting the energy log to Excel for further processing. However, it was also not stable. It encountered several errors during its runtime, which were caused by OS instability, and restarting the process was only way to overcome that.

3.2.1 Measurement Parameters

Some parameters were selected for measuring energy consumption, which are number of beacon and distance of communication. Three number of beacons are selected, which are 5, 10, and 20, because five beacons are the most common number of beacon sensed in real case, while 10 and 20 are the double size of it. Distance of communication is divided into two category, Line of Sight (LoS) and non-LoS. Each of category has three distance. The set up of the measurement experiment is shown in Figure 2.

The experiment was carried out in a Planetenlaan flat, Groningen, which has a living/bedroom, a kitchen, a toilet and shower room, and a storage room, as shown in Figure 2. The WiFi Access Point and the thin client is located in the right of the apartment next to the wall. There are seven positions of measurement in this case, in which even number indicates non-LoS, e.g., 2, 4, and 6, while odd number indicates LoS, e.g., 1, 3, and 5. The last position, 7, is added to measure the effect when the mobile phone is located outside the apartment. During the


Carrier 3:28 PM
Settings
COMMUNICATION METHOD
Method Wi-Fi (HTTP)
SERVER
http://192.168.0.103:9000
server address
TIME INTERVAL
1000
in milliseconds
NUMBER OF BEACON(S)
5
Start Communication

(a) Setting screen.

Carrier 3:29 PM
Stop Communication Display
Communication is going on...
00:00:03
Sent packets: 3
Method: Wi-Fi (HTTP)
Time Interval: 1000 ms
Server: http://192.168.0.103:...
Press back button to stop.

(b) Running screen.

Fig. 3: Screenshots example of the application showing a view for setting up the experiment (a) and a view showing the experiment progress (b).

experiment, energy consumption is recorded when the mobile phone is positioned to each of the locations. Each experiment is measured in 3 minutes time interval.

At the time of experiment, flight mode was turned on to hinder Mobile data connection that may possibly affect energy consumption. Background application may affect energy measuring result. Background App refresh in iOS is also disabled to prevent significant impact of background processes. When measuring WiFi communication, the Internet on the WiFi Access Point is disabled to prevent any running background application from fetching data from Internet through WiFi connection. Furthermore, only one means of communication is turned on during experiment, i.e., when recording HTTP over WiFi communication, WiFi is switched on and Bluetooth is switched off and vice versa.

3.3 Tailored Application Development

Unlike [2] that uses common tasks in mobile phone, such as Internet browsing and emailing, a tailored application that meets the requirements described in Section 3.2.1. This tailored application allows controlled environment that may focuses the energy measurement into several criteria, specifically for smart building application, with Azkario's architecture.

The application is written in Swift, which is the new general-purpose programming language developed by Apple Inc. Xcode, with Storyboard, is used to develop the application, with Git as the source code repository. A dummy occupancy data is sent to the server each time, as this study imitates the real implementation of user occupancy but not necessarily involves the sensing part. Each communication history is also saved using NSCoding in iOS for the sake of logging. The code is publicly stored in Github.com and is accessible at `https://github.com/gtrdp/cs-rug-internship`.

Figure 3 denotes two example of the application screenshots of the tailored application. As seen in Figure 3a, the user could setting what communication method, the destination server, time interval (if needed), and number of beacon. When the measurement experiment is running, a progress screen is shown to the user that indicates the number of sent packets and current experiment runtime, as denoted in Figure 3b.

3.3.1 WiFi Communication Scheme

One of the communication method in the application is the WiFi communication. In this study, Alamofire library[1] is used to handle the HTTP communication as it encapsulates HTTP communication for easier and elegant use. A JSON object is sent via HTTP POST method in each time interval, i.e., 1000 ms, that contains occupancy data, as shown in Listing 1. The dummy data is derived from the real implementation of Azkario's architecture.

Listing 1: Example of dummy data in JSON.

```
{
  "nearby_data":
    [
      {"data":{"proximity_zone":"NEAR",
               "proximity_distance
                   ":1.8456140098254021,
               "rssi":-81},
        "minor":1,
        "major":2222
      },
      {"data":{"proximity_zone":"FAR",
               "proximity_distance
                   ":3.171936276300526,
               "rssi":-87},
       "minor":2,
       "major":9999
      }
    ],
  "userId":"pratama"
}
```

Listing 1 shows a JSON object that consists of two nearby beacons, denoted in JSON array, that is sensed by user with ID `pratama`. Each nearby data is composed of proximity data, which indicates the distance in meter and the signal strength, and beacon data, which is denoted by the major and minor number of beacon [1]. If more beacons are set in the setting view, see Figure 3a, `nearby_data` object will have more element in its JSON array.

Moreover, a HTTP server is also built by using Play Scala framework in order to receive occupancy data. No logic is implemented in this HTTP server as this server only reads dispatched data and shows the reception timestamp.

3.3.2 BLE Communication Scheme

There are two main actors in BLE communication, the central and peripheral. A peripheral is basically the device that owns the data, while a central is a device that uses the information presented by the peripheral to perform some desired jobs. Referring to the classic client-server architecture, a peripheral can be seen as a server that has the data and a central is the client who consumes the data. An illustration depicting central and peripheral device is shown in Figure 4.

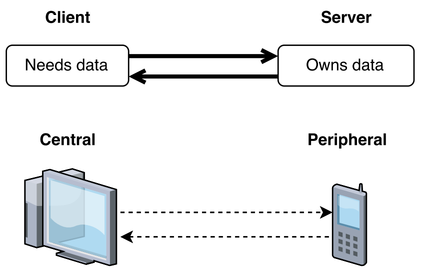


Fig. 4: Central and peripheral device illustration.

The way two BLE devices do data transmission is defined in Generic Attribute Profile (GATT) [3]. In this framework, GATT introduces concepts called Services and Characteristics. It utilizes a generic data

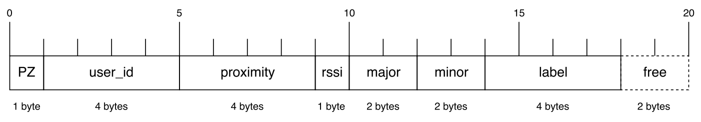


Fig. 5: BLE packet structure consisting of occupancy data and user ID.

protocol, namely Attribute Protocol (ATT), which is used to store Services, Characteristics and other related data.

According to [3], there four main methods to make use of BLE communication from central to peripheral that are defined in Characteristics Value Declaration, namely read, write, indication, and notification. Read and write characteristics are self-explanatory, i.e., it is used to read or write data from or to the peripheral. Indication and notification are the methods for pushing the data to central, which is good to send continuous data, e.g., heart rate. The important difference is indication characteristic requires an application level acknowledgment for every transmission of data. On the other hand, notification characteristics does not manage the acknowledgments by the application.

In this study, the iPhone, which provides occupancy data, serves as the peripheral, while the thin client acts as the central. Notification characteristic is used in this study because of its simplicity. Thus, the peripheral will advertise its characteristics with Unique User ID (UUID) [3] and the central will subscribe for this characteristics in order to get updated for notification.

A dummy data is also sent in BLE communication, with the maximum user data in packet is 20 bytes, as GATT protocol is used. This packet is relatively small to send occupancy data shown in Listings 1. That way, special format of data that can store all necessary data and fits with the constrains is created, as shown in Figure 5. In this format, 20 bytes of data represents a single beacon. Thus, if more than one beacon has to be transmitted, the peripheral will transmit the packets multiple times, e.g., 10 times of transmission for 10 beacons.

Initially Ian Harvey's bluepy library[2], which is a Python interface to Bluetooth LE on Linux, was used. However, it turned out that it does not work to handle the notification in BLE.

Later on, Sandeep Mistry's noble[3], a node.js BLE central module, was used. It turned out that it is capable to handle notification scheme in BLE communication.

iOS only supports BLE communication using central-peripheral communication scheme. It does not support classic Bluetooth communication.

In Swift, the OS is not responsible to handle unsuccessful packet data transmission, the application itself must be aware to handle packet data transmission failure.

Explain BLE 20 bytes of packet size policy, and explain how those packets are implemented. Cite specification of bluetooth and john abraham.

Changed raw username (utf8) to userid (UInt32) Changed proximity type from Double to Float Added random label to indicate a particular timestamp for a ceratain data packet

Bluetooth in noble is turned out to be unstable, it always loses connection when it reached 29th data packet. As a solution, the central application always reconnect to the peripheral when disconnected.

Bluetooth connection requires manual trigger to start scanning nearby BLE devices. In order to do the experiment efficiently, beside SSH server that is used to access the thin client remotely.

[give an image depicting central and peripheral communication]

Play Scala web server and Bluetooth server.

4 Results and Discussions

In order to inspect the effect of packet size, the number of sensors are increased significantly. That way, the energy consumption is assumed to be increased as well.

[1]`https://github.com/Alamofire/Alamofire`

[2]`https://github.com/IanHarvey/bluepy`

[3]`https://github.com/sandeepmistry/noble`

Fig. 6: HTTP over WiFi measurement result.

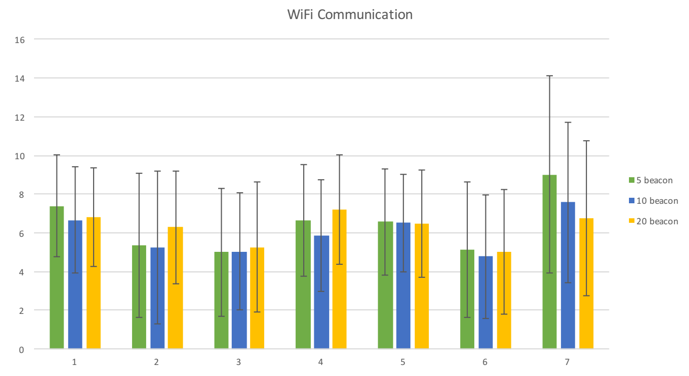


Fig. 7: BLE communication measurement result.

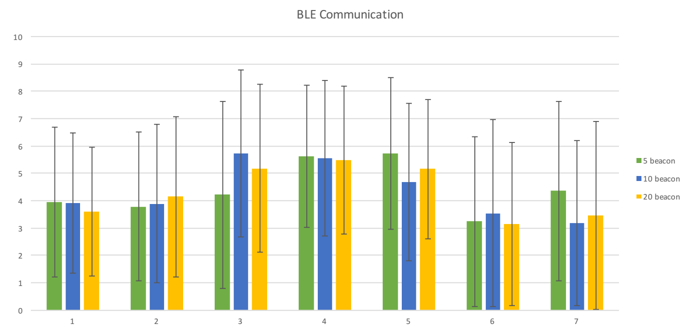


Fig. 8: HTTP over WiFi and BLE communication for high number of sensor.

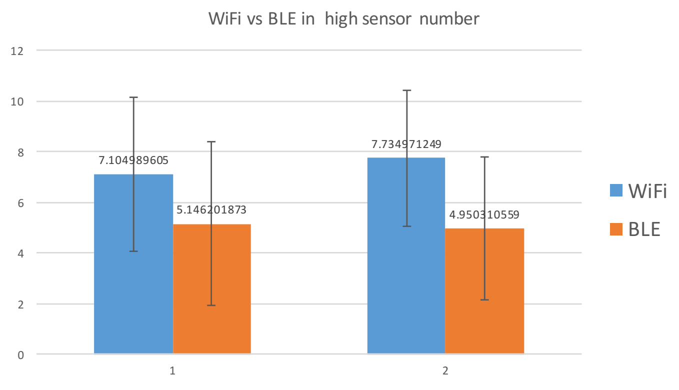
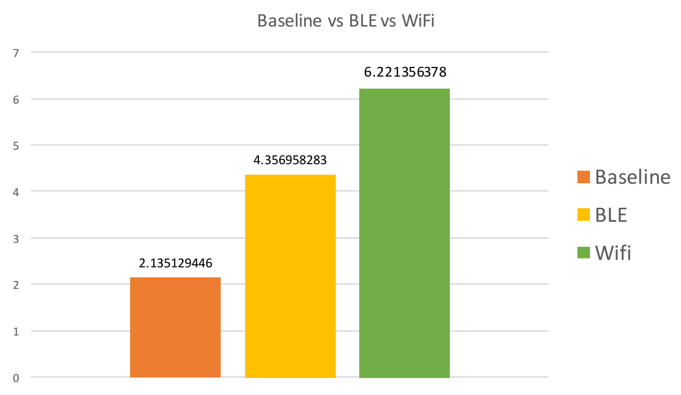


Fig. 9: Final energy comparison.



Explain in detail first, separately for bluetooth and wifi. Then explain the comparison of bluetooth and wifi and the baseline.

Time to send each BLE packet in this case is around 4ms. If the number of sensor is more that the threshold, some data will be lost.

In the experiment, brightness was proven to be one of significant source of energy loss as it consumes much energy to light up the screen.

BLE device's battery can lasts up to 3 years. Thus, if the BLE device which has small mAh of power can last up to 3 years, a mobile phone which has way more power than that must be able to last longer. The phone is transmitting the same amount of signal power regardless how far it is from the central -¿ there is no significant power consumption difference. If the central goes beyond the limit of BLE signal border, the communication simply un-do-able, i.e., the peripheral is not required to boost up the transmitting power to maintain the communication. There is no API to change the transmitting power of BLE in swift (AFAIK).

The scanning that still needs human interaction Energy reporting which is un-exportable. Solution: Creating applescript to copy and paste from instruments to excel automatically. The bluetooth is not stable: It always stops at 29 packets -¿ solution: using resubscribing method. It has to be triggered by GNOME (GUI) Bluetooth management tool. Why it is unstable? Is it because of noble? This research is only energy consumption for sending data. Without sensing etc. Explain, this may differ in the real implementation, as both bluetooth and wifi may be turned on. The result might be slightly different. Limitation: only tested for iPhone 6 running iOS 9.3.2.

5 Conclusion

We have presented that Bluetooth is more energy efficient than Wi-Fi.

Wifi may still be a consideration because usually mobilephone are also using wifi extensively.

Some drawbacks may persist if bluetooth is implemented, such as instability. Other implementation than what is impelemented here must be looked for.

Acknowledgments

The authors wish to thank A, B, C. This work was supported in part by a grant from XYZ.

References

[1] Apple Inc. Getting Started with iBeacon. pp. 1–11, 2014.
[2] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference - IMC '09*, pp. 280–293, 2009. doi: 10.1145/1644893.1644927
[3] Bluetooth SIG. *Specification of the Bluetooth System - Bluetooth Core Specification 4.2*. Number April. 2014.
[4] G. Conte, M. De Marchi, A. A. Nacci, V. Rana, and D. Sciuto. BlueSentinel: a first approach using iBeacon for an energy efficient occupancy detection

system. *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pp. 11–19, 2014. doi: 10.1145/2676061.2674078

[5] B. Lucchesi. Profiling Power and Network Usage for iOS, 2014. Accessed: July 2, 2016.