ulm university universität uulm

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Design and Implementation of a Bluetooth 4.0 LE Infrastructure for Mobile Devices

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Christian Alexander Goosen
christian.goosen@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Marc Schickler

2014

Version of May 16, 2014

Satz: PDF-$\LaTeX\,2_\varepsilon$

Abstract

The new Bluetooth 4.0 low energy (Bluetooth Smart) specification provides developers a means to create unique and never before seen systems. Using low energy beacons, new systems are able to deliver location based information to the user, whenever it comes into close proximity of a certain beacon. BLExpo, a beacon based service developed for an enhanced user experience at expo's, relays location sensitive information saved on a server to the consumer, depending on his location, is described in this thesis in detail. Adapting the design of the service, a vast variety of different systems can emerge, and existing ones improved, by supplying beacons. A brief look at how Bluetooth low energy can make our lives more convenient is provided, as well as the issues that might arise when dealing with beacon based systems. Alternative designs are taken into account by describing systems that rely purely on Bluetooth low energy for both event triggering, and information exchange. Since every system mentioned in this thesis uses Bluetooth LE as the underlaying technology, it is also introduced and a brief overview is provided.

Contents

1 Introduction

In the modern world of today, there are certain things that people just cannot live without. A smartphone is probably one of the highest priorities after basic necessities. Today's smartphones have technologies and features that people might not know even existed, even though they own such a modern and scientifically advanced mobile device. But when it comes to Bluetooth, everyone knows what it is, or at least, has heard of it before, because it has been a feature in every mobile phone since 2000 [1]. However, not everyone uses Bluetooth in their daily lives. This might change, now that a new era has dawned upon Bluetooth and its users, an era known as Bluetooth Smart, an era that might change they way people and developers use Bluetooth in the future. It might change the way we shop, the way we live in our homes, and even the way we interact with our friends and family. Our perception of what Bluetooth stands for, and how it is used, needs to change. Bluetooth Smart does not replace classic Bluetooth that can be found on millions of devices all over the world, but it works alongside it. This new technology does not exist to connect your smartphone with your wearable Bluetooth headset, like the classic Bluetooth technology does, but rather to communicate and transfer small amounts of data between modern devices that look like they were taken from a set of a science fiction movie. Bracelets, watches, shoes, wearable heart-monitors, etc. are all implanted with dual-mode Bluetooth chips, which allow them to communicate with your Bluetooth Smart Ready smartphone, for example, and grants the user access to very unique information. It is particularly the health, sport, and fitness sectors that are the prime targets for this new market, but it is most definitely not limited to them [2][3].

Bluetooth Smart suddenly became very prevalent when Apple announced its new iOS7, back in September 2013 [4]. While normal users hyped over the new look and feel of their iPhones, developers were more looking forward to the silent, but yet very prominent iBeacon announcement. Although the announcement is already months old, many people still do not even know what it is exactly, and that might take them by surprise when iBeacons are introduced to many places around the world in the future. These so-called beacons are small, low cost, and low energy consuming devices, that have the sole purpose of sending out waves recognizable data packages, known as advertisements. Because they are small and can run on a single battery for almost two years, they can be hidden away from plain sight. In turn, devices that have the ability to scan and read such advertisements can also determine the approximate distance between a beacon and itself, which turns out to be a very useful feature, since applications can now be implemented to initiate events once a device gets within a certain distance of a beacon. For instance, walking into a coffee shop could make the respective application show you a full list of coffees available at that shop. Beacons are uniquely identifiable, and thusly allow applications to provide contextually relevant information, based on the consumer's location within a region. As in the example above, applications know when you enter a certain shop, or region, but this is already do-able with current GPS sensors. Beacons, though, also offer in-door functionality, because they do not rely on satellite communication, which sets them apart from GPS systems. Furthermore, it is possible to determine whether or not a device is within close proximity of a beacon, ranging from around ten metres to a few centimetres [5]. These ranges are not guaranteed to be completely accurate because the surroundings can negatively affect a beacon's signal. However, since beacons can be calibrated to adapt to the obstacles around them, these measurements are still accurate enough to be used in real world applications.

1.1 Aim of this Thesis

Besides exploring the new Bluetooth Low Energy specification, this thesis also focuses on the design and implementation of a system, labeled BLExpo, that uses beacons and Bluetooth 4.0 as the underlying technology to provide the user with location sensitive information. The system is designed to be used at exhibitions or conventions, where it is useful to have location based data. Since every exhibitionist, or companies as they are referred to in BLExpo, have beacons at their booths. When a consumer walks near that booth, it will receive relevant information concerning that company or its products. Alternatively, beacons can be set up to take on an active role. Walking nearby an active beacon will result in its activation, that can be anything it was set up to do, ranging from simply showing an image on a screen the beacon is connected to, to controlling hardware. The BLExpo client, which is an Android 4.4 application that runs on a mobile device, actively seeks beacon advertisements, and once found, displays the relevant information on screen using a predefined layout. In its current state, BLExpo offers a central station, where users have the ability to add beacons to the service, and customize how the information will be displayed on the mobile device. Although this is limited to only text and images, the system itself can be adapted and modified, to show most anything on screen. On the other side of the spectrum, the server uses a simple RESTful service, on which all the data is stored, to relay beacon information back to the client upon request. The system can keep track of all requests, which include client data as well as location sensitive information. This way, BLExpo knows when and from where a certain beacon was requested. This information can be used in the future to optimize certain aspects of an exhibition, like mapping out the next event to have the more popular booths at better locations. People who want to embrace this technology should be made aware of the fact, that their locations can be tracked by Bluetooth applications. Although this is done by BLExpo, as previously mentioned, it allows the users to be anonymous, if they do not wish to be identified. Tracking is only one of the issues future Bluetooth Smart and beacon systems need to overcome, in order to become prevalent in our lives. A look at other issues and alternative implementations is also provided within this thesis, and a look at how BLExpo can be adapted to feature in a variety of other systems is given.

Widespread adaptations are necessary for it to really take off. While we are still waiting on an exciting announcement from Google for official Bluetooth Smart support for its Android operating systems, and Microsoft for Windows Phone, Apple already offers developers an application programmable interface to use when designing new applications for its iOS systems. They have also equipped their major stores in the US to use iBeacons in order to streamline shopping experiences [6]. Also of note, Major League Baseball (MLB) and Apple collaborated to install beacons in most of MLB's stadiums, so users can get an enhanced experience when watching games. MLB's "At the Ballpark" application is required (Figure 1.1) and it offers features such as automatic check-ins, interactive maps, social media integration, team schedules, event and ticket information, access to food and beverage ordering, etc [7]. The information provided will, of course, differ from ballpark to ballpark.

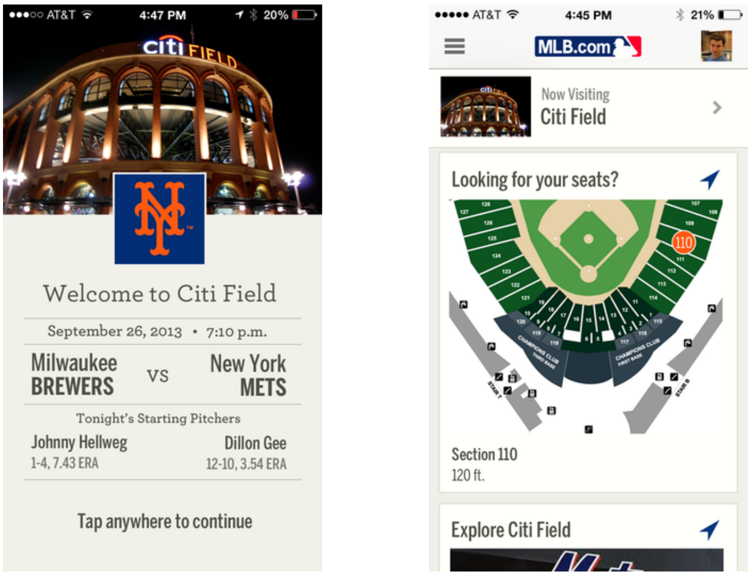


Figure 1.1: At the Ballpark by MLB [8]

If this technology proves popular amongst consumers, and if they are willing to use it, it will undoubtedly be introduced to more and more places around the world. It opens up a world of possibilities if you can provide consumers with location-sensitive information. In the future, we could go to the cinemas and receive the upcoming movie schedule as we walk through the door. Had we bought tickets online earlier, it could point us to the direction of where the movie is to be shown, its starting time and more. After the movie is done, the application then shows us additional activities that can be done close to the cinema. This technology can also be used at home. Turning your smartphone into a remote control when you sit on the couch to watch television, or controlling the lights depending on whether you walk into or out of a room, are just a few examples of what is possible with proximity based technology. But it all comes back to Bluetooth, without which, systems that depend on beacons, like BLExpo, cannot exist. We must first take a closer look at Bluetooth itself, to understand more clearly how the underlying technology works, and how it has come this far to potentially change the way we live our lives.

1.2 Structure of this Thesis

This thesis starts off with chapter 2, where a look at the technology that all the systems described in this thesis rely on, namely Bluetooth Smart, is given. A small comparison is given to briefly introduce the reader to both the classic and new Bluetooth specifications. The latter is described in more detail to provide the reader with basic knowledge of how systems and devices are able to make use of Bluetooth Smart, particularly beacons. Chapter 3 grants the reader an overview of how a basic system can be designed using such beacons. Followed by a concrete example of an implementation of a BLExpo service, where the user is provided with detailed information of how every part of the system functions and communicates, succeeded with the possible issues and limitations that can be encountered by the system in the real world in chapter 5. A purely Smart system, that differs from BLExpo by not relying on a server to communicate with the client, is introduced and described in chapter 6, along with its issues. A wide variety of existing and future systems, both loosely and closely based on the BLExpo system design, and some on the purely Smart system design, is looked at in chapter 7, and how they can be improved or implemented by using beacons. Chapter 8 provides the reader with short conclusions to every part of this thesis, as well as the current progress of Bluetooth Smart systems and what we can expect in the near future.

2 Bluetooth Low Energy

As the name implies, Bluetooth low energy (also called Bluetooth LE or BLE and is marketed as Bluetooth Smart[1]) is modelled to be much more energy efficient than classic Bluetooth, and was designed to allow small devices, running on coin-cell batteries, to communicate for many months, possibly years [9].

Development of Bluetooth Smart began as early as 2001, when Nokia realized that there was a need for low power consuming wireless technology. The results, published three years later, was named Bluetooth Low End Extension, is an adaptation of the Bluetooth standard. However, they did not just stop there. In October 2006, Nokia unveiled a wireless technology that would run alongside Bluetooth, known as Wibree [10]. It used just a fraction of the power that Bluetooth needed, but at a cost of speed; Bluetooth, at that time, could transfer data three times faster. After negotiating with the Bluetooth Special Interest Group (Bluetooth SIG), it was decided to include Wibree in future Bluetooth specifications, as Bluetooth ultra-low-power technology, which we know today as Bluetooth low energy [11].

2.1 Classic Bluetooth and BLE

Although BLE and classic Bluetooth use the same spectrum range (the 2.400 GHz-2.4835 GHz ISM band), they use different sets of channels. BLE uses only 40 (2MHz) channels, whereas classic Bluetooth uses 79 (1MHz). In order to mitigate interference caused by crowded bands, both technologies use a technique called frequency hopping, albeit with different details. For devices to be discovered, there are three RF channels available for advertisements that are allocated in different parts of the spectrum to avoid interference from Wi-Fi [12].

Another difference between classic Bluetooth and BLE are the data transfer rates. Classic Bluetooth, in its original release, could transfer data at 1Mbit/s. It had gone up to 3Mbit/s with the release of Bluetooth 2.0 Enhanced Data Rate version, and can reach even faster speeds with the High-Speed version of Bluetooth 3.0[2]. With Bluetooth 4.0, however, a fast data transfer rate was dropped in favour of lower power consumption. To achieve this, BLE sleeps for longer periods of time, send bursts of data less frequently, and it does not maintain connections, like classic Bluetooth, but can quickly re-establish links when communication is needed again. Services that

[1] The terms Bluetooth low energy (BLE) and Bluetooth Smart are interchangeable, they are different names for the same subset of the Bluetooth 4.0 specification.

[2] Bluetooth 3.0 + HS only used Bluetooth for building a connection, the actual transfer (at 24MBit/s) took place over a 802.11 link.

stream music, for example, will still be used by classic Bluetooth instead of the new low energy version.

Additionally, because of the low power consumption, BLE has a smaller time window to send data in. With classic Bluetooth having 100ms, and BLE less than 3ms, justifying the fact that BLE was designed to send less data than traditional Bluetooth [13]. To ensure that the data transfered in this time is not redundant, the Generic Attribute Profile (GATT) that is specific to Bluetooth v4.0, is used as a specification for data transmissions over BLE links. The Bluetooth SIG defines profiles for BLE devices, which describes how a device works in a certain application [14]. Profiles that are defined by the Bluetooth SIG include battery services, blood pressure services, heart rate services, and many more. (For the complete list see [15].)

Most of today's modern smartphones and tablets come equipped with a dual-mode Bluetooth chip, that allows them to communicate over classic Bluetooth, or over BLE. Since the Bluetooth v4.0 specification does not require devices to implement both the BLE and classic Bluetooth protocols, not all Bluetooth devices are backwards compatible, hence the three different kinds of Bluetooth devices that can be found today: Bluetooth, Bluetooth Smart, and Bluetooth Smart Ready.

Figure 2.1: Bluetooth Version Compatibility Comparison

Devices that only support classic Bluetooth are usually older, like peripherals, car systems, etc. Newer devices, especially smartphones and tablets, support Bluetooth Smart Ready (also known as Bluetooth 4.0 dual-mode devices, because they implement both protocols) are backwards compatible, and are able to communicate with classic Bluetooth devices. They do not benefit from the low power consumption of Bluetooth v4.0 when streaming data with a classic Bluetooth device. Bluetooth Smart devices, on the other hand, are not backwards compatible, they only support communication between BLE devices. In order to distinct between these different types of Bluetooth, a new set of logos were designed by the SIG in 2011 (Figure 2.2) [16].

Ultimately, since BLE and classic Bluetooth can co-exist, it is not a question over which is simply better, but rather which is better suited for certain applications.

Figure 2.2: Bluetooth Logos [17]

2.2 Technology Basics

The Generic Access Profile (GAP) defines different Bluetooth modes for the application layer. There are different modes in which a BLE device can operate:

- Advertising mode
- Scanning mode
- Master device mode (Central)
- Slave device mode (Peripheral)

In advertising mode, the BLE device periodically transmits a data package containing information about the device and its services. To establish a link, a scanning device must capture the advertisement and request a connection. A scanning device may also request additional information first. After successfully connecting, the scanning device will then take on the role of the master device, and the advertising device becomes the slave. A slave device can only have one connection at any given time, whereas a master can have many simultaneous connections to slaves.[3] In real world BLE applications, a smartphone or tablet would act as the master device, and sensors as the slaves, which allow them to be very small and cost efficient. The interval in which a BLE device send out advertisements, can be set to anything between 20ms and 10s. A shorter interval means a shorter battery life, but setting the interval too high can result unresponsive applications.

Over the air BLE data packets contain a preamble, an access address, a CRC, and a protocol data unit (PDU). The size of the PDU can vary from 2 bytes to 39 bytes, meaning the shortest packet can have 80 bits, the longest one 376 bits and can be transmitted within 0.3ms. For advertising devices, the PDU consists of a 16 bit header and up to 31 bytes of information. Since additional 31 bytes of data can be requested by the scanner, a reasonable chunk of data can be retrieved from the advertising device before even creating a connection between the two devices [18].

[3] In classic Bluetooth, this number is limited to 7, however, in BLE it can vary, depending on the implementation.

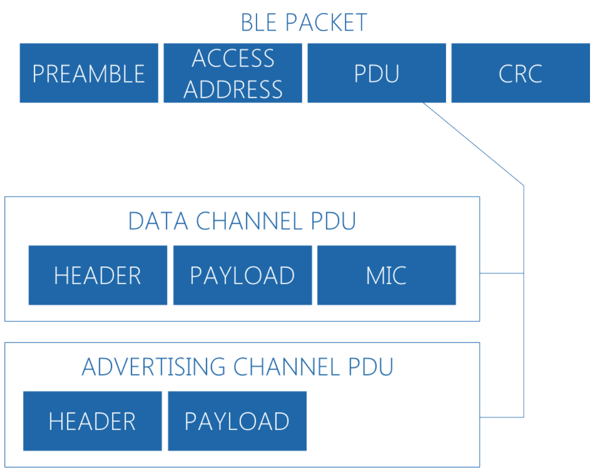


Figure 2.3: Different PDU Channel Uses

Upon establishing a connection, the data channel PDU is used, consisting of a data packet header and up to 37 bytes of payload, and finally 4 bytes of Message Integrity Check data, if the connection is encrypted. The master device starts each communication event, in which the master and the slave take turns to send and receive data packets until either one stops sending, which will suspend the event until the next one occurs.

2.2.1 Pairing

Since Bluetooth can allow control over a device or expose private data, it is essential to be able to recognize devices prior to the connection. And for user friendliness, be able to open connections to trusted devices without human interaction. The process to resolve this issue is called *bonding*. For two Bluetooth devices to bond, they need to go through the one-time pairing process first, which is also defined by the GAP.

In the pairing process, user interaction is usually needed to confirm the identity of the devices. There are three main types of (Secure Simple) pairing in Bluetooth v4.0 [19]:

- **Just works:** No human interaction is needed for this process, since it ”just works”. A device may, however, ask the user for confirmation to complete the process. Devices with limited input capabilities, like headsets, use this type of pairing. This is the only pairing method that is susceptible to man in the middle attacks.
- **Numeric comparisons:** When both devices are able to display a 6 digit numeric code, and at least one device can accept an input of yes or no, this method is used. If the codes shown on both devices are the same, the user should proceed to accept the bond.
- **Passkey Entry:** Used when one device has a display, to show the 6 digit code, and the other device has a keypad to enter and confirm it. Or when both devices have keypads, the bonding is successful if the same key was entered on both devices.

- **Out of band:** OOB methods are supported by external methods, such as NFC, to exchange data.

Bonding is only required when devices want to send and receive data over Bluetooth. It is also possible to design systems that rely on BLE without the bonding prerequisite.

2.3 Beacons

Using BLE, it is possible to create so-called ”Beacons”, which are very simple Bluetooth Smart devices, that does little else than to periodically advertise themselves with small 31 byte data packages. These special advertisements can then be interpreted by other Bluetooth Smart devices. The data that is sent with each advertisement pulse, includes a unique identifier (UUID), a power level indication (RSSI), and a major as well as a minor value (Figure 2.4).

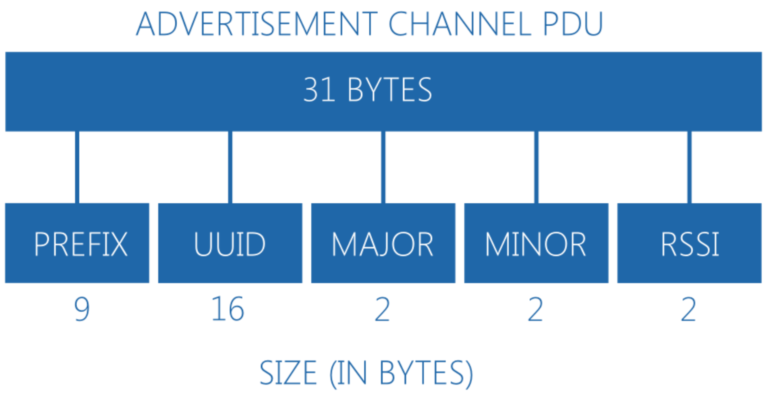


Figure 2.4: Beacon Advertisement PDU Breakdown

The advertisements that are sent over the air by the beacons are in binary format, but can be parsed into hex. One such advertisement, coming from a beacon with the UUID ”E2C56DB5-DFFB-48D2-B060-D0F5A71096E0”, and major and minor values of 0, looks like figure 2.5.

HEADER	d6 be 89 8e 40 24 05 a2 17 6e 3d 71
PREFIX	02 01 1a 1a ff 4c 00 02 15
UUID	e2 c5 6d b5 df fb 48 d2 b0 60 d0 f5 a7 10 96 e0
MAJOR	00 00
MINOR	00 00
RSSI	c5
CRC	52 ab 8d 38 a5

Figure 2.5: Intercepted Beacon Advertisement (Parsed into Hex) [20]

The advertisement channel PDU is shown in blue. All values are in hex and are separated by a single space character [21].

Beacons come in different sizes and formats, ranging from software implementations like virtual machines, to small USB sticks. The easiest way to build beacons into a system, is to buy ready-to-use beacons, like Estimote. Estimote offers an API, as well as a SDK with which buyers can customize beacon configurations, such as the UUID. Estimote beacons come in three different colours and a waterproof casing, allowing it to be installed outdoors and even underwater [20].

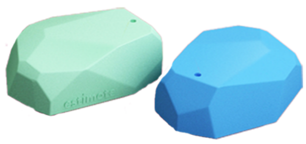


Figure 2.6: Estimote Beacons [20]

Instead of using an actual beacon, it can be emulated by running a virtual machine (e.g. provided by RadiusNetworks) in conjunction with a small USB Bluetooth 4.0 dongle [22]. By creating a beacon in this manner, it limits the amount of places that the beacon can be placed, since it has to be connected to a computer. However, it does offer more power and resources than other beacons, if it is required. If the mobility is not a problem, then a Raspberry Pi can also be used to emulate a beacon [23]. It is based on the same principles of the virtual machine, but is run on its own linux based operating system, Raspbian. A Raspberry Pi can be configured to start advertising the itself after it is plugged in, so no other hardware is needed, like screens or keyboards. The Raspberry Pi can also be set up to scan UUID's instead of advertising them, and thusly can swap its role with that of the smartphone [24]. Yet another alternative are the RadBeacon USB's, also created by RadiusNetworks. These real beacons receive power through a USB port, which again, limits the areas where such beacons can be placed. Their values can be configured by using the RadBeacon Configuration application for iOS [25].

2.3.1 UUID

The universally unique identifier (UUID) is represented in a special 128 bit (16-octets) string and is primarily used to identify beacons. This UUID is representative of a group, like a company, and is provided in the corresponding BLE application. The UUID is the largest part of the advertisement package, and in being so large, it almost nullifies the possibility of two groups having the same UUID. Although there is a slight chance of this happening, it is so little, that it can be unaccounted for [26]. Were it to happen, then beacons that are meant for a certain application could trigger events for another, which would only confuse the user.

2.3.2 Major and Minor Values

In addition to its UUID, a beacon can be further distinguished by the accompanying major and minor values. The major value is used to group together a set of beacons that share the same UUID, whereas the minor value differentiates a small, location based distinction [5]. For example, if a company, like Ikea, decides to use beacons in their shops, then they would create a UUID, which would be used with every beacon they own. The major value would then be used to differentiate between the stores, and the minor values would be utilized to pinpoint to an exact location within a store. How these values are assigned, is solely up to Ikea, who needs to use this information they assigned to each beacon accordingly in their application. Since both the major and minor values are both two bytes in size, there are many possible combinations of minor and major values[4], to ensure that all beacons can use the same UUID without conflicts.

2.3.3 Received Signal Strength Indication

Using the received signal strength indication (RSSI) sent with the advertisement package, it is possible to determine the approximate distance between a beacon and a user holding a BLE device. Although this measured distance is not completely accurate, and will deviate a little because of radio interference, it is still accurate enough to determine whether or not a user is within a certain distance from the beacon. Being able to obtain this kind of knowledge is what sets beacon based systems apart from other similar technologies. With NFC, probably the only competitor to BLE at this time, you can only determine the location of a user who touches a NFC tag with its device, but with beacons, it is possible to see all users in a region, and their distances from the beacon. The only advantages that NFC offers over BLE is the lack of battery, and its price. A beacon can be as cheap as 10€, obviously depending on the type of beacon, but is still a long way from the price of an NFC tag, which can be obtained at around 0.80€[27][28].

Notably, there are three ranges (distances) predefined by Apple: immediate, near and far (see figure 2.7) [5]. It is, however, very much possible to determine your own ranges when creating a system instead of using these values. Custom values should be well distinguishable to eliminate issues caused by deviances of measured distances, and in some cases, you might want to offer different interactions based on the user's distance form a beacon. Based on the application's requirements, there can be more than three defined ranges, or they can even be dynamic.

For some beacons, an option to calibrate is added in included applications, which allows for a more accurate measurement. In the calibration process, you can change the RSSI value that is sent with each advertisement package. When moving the beacon to a new location, with different obstacles, it is best to recalibrate for optimal user experience. However, it should also be noted, that some obstacles cannot be accounted for, like groups of people who might not always be there to cause interference.

[4] For the minor (or major) alone there are 2^{16} (= 65536) possible combinations.

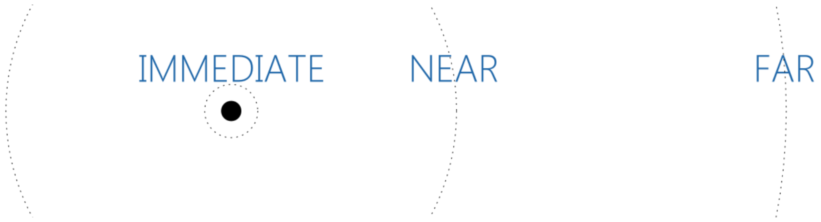


Figure 2.7: Beacon Proximity Ranges

Immediate

Immediate is only mere centimetres away from the beacon. At this short range, your BLE device must almost touch the beacon to trigger an event. This is useful, since it can be used as a more secure way of triggering an event, than it would be at a much further distance. This might seem very similar to near field communication (NFC), but it does not transfer any data, unlike NFC. It is still useful, however, since it requires the user to knowingly activate the event, by touching the beacon with his smartphone, and thusly it cannot be triggered by accident or just by walking close by the beacon. Again, similar to NFC, this can and will most likely be used for a secure way of paying. If you launch an application on your device that has the ability to transfer real world money, it will initiate the payment when your device is within immediate range. Although not every NFC use-case can be imitated using beacons, most can, and since beacons can also be used at longer ranges, it clarifies Apple's decision to not support NFC in any of their devices in favor of BLE.

Near

Near indicates a few metres, meaning that a device is within close proximity. Events will trigger when consumers stand in front or walk past the beacon. Being close to the beacon, but not actually having to touch it, allows beacon based systems to either display location sensitive information, or offer unique user interaction based on the beacon it is in close proximity to. Like in the "At The Ballpark" application, an example event that near could trigger is to show a map of how to get to your seat after entering the stadium.

Far

Being *far* away from a beacon, usually means the measured distance is over 10 metres. Far can be used to determine whether or not a user is in visible range of a store, for example, and the respective application can then notify the user of that fact, and might even show additional information such as special offers, catered to that user. Such an event should only be triggered once, when first recognizing a specific UUID.

3 Design

By using Bluetooth low energy, systems can take advantage of the distance tracking between devices and beacons, to ensure that a user is standing in the direct vicinity of such a beacon. Having that knowledge is key to the system, since now the user can be notified of area-specific information.

3.1 System Outline

The system consists of three main parts: The beacons, the client, and the server. Every part of the system plays an integral role and the system cannot function without any of them, although it is possible to create systems that only rely on Bluetooth communication, and thusly not require a server (see chapter 6: Purely Smart System). In the following section, the system in all its functionality will be briefly described.

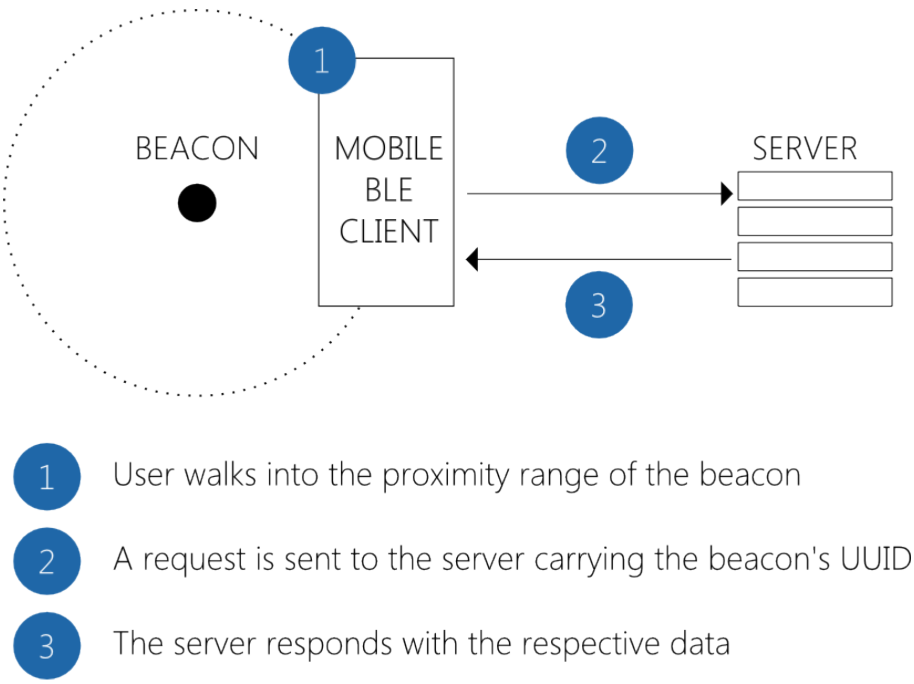


Figure 3.1: System Design

3.2 Process

It all begins with Bluetooth low energy beacons, that are configured to advertise themselves using their unique identifiers (UUID, minor and major values) as well as their signal strength. There are two kinds of beacons that can be used in this system, passive and active beacons. In one such system, there can be almost a countless[1] amount of unique beacons. At least one beacon is required in the system, otherwise the client and the server will never communicate.

Advertising beacons is not enough, since there needs to be something to see the advertisements, and eventually do something with the information that is included. This is where the client comes into play. The client, which could take on the form a smartphone, is running an application that is configured to scan for Bluetooth Smart beacons. Time between scans depends on the implementation and usage of both the client software and the beacons, it occurs often enough that reliable systems can be built using Bluetooth Smart technology, that is dependent on such scans. In this system, it is assumed that scanning takes place at least once a second, like it is implemented in RadiusNetwork's open BLE API for Android [29]. Scanning clients can identify beacons by their unique identifiers (UUID), and additional distinctions can be made by using the beacon's minor and major values, and then use this information to send a request to the server. The client only acts as a bridge between the server and the user, by relaying the information gotten from the server to the user.

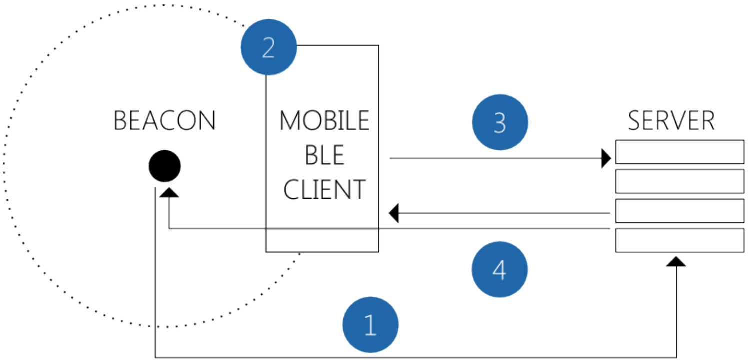
The user is the pivotal point of the system. It takes on the role of initiating the communication between the client and the server. As soon as a user enters the close proximity of a beacon, a request is sent to the server using the beacon's unique identifier as a parameter. It is up to the user to decide whether or not to initiate this process. Assuming it does, then the server comes into action, by gathering the necessary data from its database and sending it to the client, to ultimately show it to the user.

Essentially, in this system, the client takes on the active role of displaying data retrieved from a server, as well as the user actively initiating the communication between client and server.

3.3 Active Beacons

The client's active role can be reversed. Instead of the client doing something, like showing a new screen or loading a web page, the beacon can be commanded to initiate a process. Since beacons are programmed to just advertise, a workaround is needed to make them listen to incoming requests, like making them into web servers, whose IP addresses are made known to the server. Once a client requests an active beacon, the server can initiate a command in addition to sending the response back the the client, making the beacon active. Activities can include displaying images, videos, or switching hardware on or off, etc. This still needs a user to initiate the command. (Figure 3.2)

[1] The UUID is 128 bit, minor and major values are both 16 bit, so in total there are 2^{60} uniquely identifiable beacons.



1 Beacon syncs with the server, to make its IP known

2 User walks into the proximity range of the beacon

3 A request is sent to the server carrying the beacon's UUID

4 The server responds to the client and sends a command to the beacon

Figure 3.2: Active Beacon

3.4 Local and Remote Servers

The server part of the system can be implemented either locally or remotely. When running a remote server, it is important to note that the client then requires a working internet connection, in order to send and receive data from the server. Running the server locally will still require a connection to the network, but it can be acquired by signing into the local Wi-Fi network. Both setups have their advantages and disadvantages.

3.4.1 Local Server

Setting up a local server might sound like the easier option at first, but it does impose some difficulties. The server needs to be, as the name implies, local. It needs to be at the place the system is used, and more importantly, needs to be connected to the same local network all the clients will connect to. This connection is vital to the system. Having a local server means that the IP address of the server will be dynamically assigned according to the network,[2] thus making it harder for the clients to connect, since they need to know the server address, and be configured manually. To overcome this, a remote server can be used instead.

3.4.2 Remote Server

Using a remote server offers an easier setup. The server address is known and can be hard coded into the application, or it can be entered by the client using a known, resolvable address,

[2] Two different networks might not assign the same address to the same server.

for example www.blexpo.com/services/1. However, it comes at a price. The service has to be hosted elsewhere, so hosting costs will apply, and clients will all require a working internet connection to reach the host. Expanding the system, or if more than one service becomes available, the clients will then need to be configured accordingly, just like it is with local servers.

Additionally, setting up active beacons using a remote server becomes tricky, and will require a different implementations of parts of all three pieces of the system, in order to function correctly. For now, the beacons are only callable from the local network, to which the clients also need direct access to, to communicate with the beacons. Since the beacons cannot sync with a remote server, they need to make their IP known in a different manner, like converting it to a hex string and using it as the first part of the UUID, which can then be interpreted by the client to send a request or a command to that beacon, instead of the server, like it is implemented when using a server locally.

4 BLExpo

BLExpo, short for Bluetooth Low Energy Exhibition Service, is the name of the infrastructure designed using both Bluetooth Smart and HTTP technologies. It is based on the system described in the previous chapter. It is a service used to enhance the user experience at exhibitions and conventions, where companies can set up beacons at their booths and users can walk by to receive (extra) content, such as contact details and photos, much like a digital flyer, that you can take home with you. BLExpo was designed with the goal of delivering next-generation exhibition experience using existing Bluetooth technologies, and proving that we are already living in a time, where simply using Bluetooth 4.0 enabled devices, can improve our day-to-day lives.

4.1 The Vision

Imagine that you are visiting your favorite exhibition, that now supports BLExpo, and you decide to try it out. Luckily, your smartphone already supports BLE, so all that is needed now is the application that you install on your phone. Come convention day, and as you stand in line to purchase your entrance ticket, your phone vibrates. It is a message, that welcomes you to the convention and tells you to show this message to the clerk to get a discount. After already starting a great day you decide that it is most reasonable to first get a map of the convention hall, so you walk to an information stand, and before you can actually ask for a map, your phone vibrates again, and as you look down at it, you see a map and an event schedule. Quite satisfied by the confused look on the face of the helper behind the desk as you just turn around and walk away, you follow your new map to the first booth you like to visit. As you get to the booth, your phone vibrates, and you see photos, maybe even a video, and contact details, which you decide on looking at later, so you just press on the download button, close the application and continue on. The next booth you visit, seems different, it has a big screen hanging from the top, but it seemed to be turned off. You walk closer to get a better look at what that booth is, and as you approach, the TV magically switches on, and you are greeted on the TV, then a video starts playing, a video personalized just for you. After watching the video, you realize it is time to eat, so you go to the cafe to grab a quick snack. Just as you enter the cafe, your phone vibrates and notifies you about a hamburger and coke special, which every reasonable person would of course get for such a low price. After a very tiring day of visiting almost every booth, its time to take the flight back home. On the plane, you want to revisit a few booths that caught your attention, so you open the BLExpo application, open the downloads folder and have immediate access to all the information, images, videos, contacts, etc. that you collected earlier that day.

As you are approaching your house, the garage door opens to let you in. It closes behind you as you drive into the garage, and the lights switch on. Since you have been gone for a few days, the heaters were turned off. But recognizing your presence, they turn back on, and automatically adjust their temperatures to a comfortable level. You sit down in the living room, the TV switches on, the lights are dimmed and you enjoy a relaxing evening at home.

It is an imagination that you might have heard of before, might have imagined before yourself, but the truth of the matter is, this is reality. All this can be done, and is done by BLExpo to a certain degree. All the pieces of a BLExpo service described above, were introduced in the previous chapter, and will now be looked at in greater detail, having a deeper focus on the BLExpo implementation rather than the generalization of what each of those pieces do.

4.2 Server Side

The BLExpo service was first intended to run on a local server, but can certainly also run remotely, but this will imply difficulties when working with active beacons, as described earlier (section 3.4.2). The BLExpo server consists of two parts, the first being that of a web interface (JSP Pages) where it is possible to register and customize beacons, using a simple web page. This side of the server will mostly be seen by companies, who wish to set up beacons with a certain BLExpo service. It is important to note that you can register a beacon on as many services as you like, what the client will see at the end of the day depends on which service the client is connected to.

The second role of the server is hidden. It is servlets that are called only by the client-side application, to retrieve and send information to the server, via simple HTTP requests and responses. Requests include:

- finding a list of all registered beacons for a certain service
- finding information concerning a specific beacon
- getting images and logos belonging to a specific beacon

For easy adaptation and expansion, JSON objects are returned as a result of all of the above requests except for the images. The debug beacon, which is called to test the connection between the client and the server, also responds with a JSON object (Figure 4.1).

BEACON REQUEST
http://localhost:8080/BLExpo/response?UUID=debug

SERVER RESPONSE
{"title":"Connection Test",
"color_title":"#ffffff",
"color_title_background":"#294796",
"welcomeText":"Hello, <user>!",
"color_text":"#ffffff",
"color_background":"#000000",
"color_logo_background":"#294796",
"shortDescription":"If you can read this, it means that everything is set up correctly, and you will be able to use the app as intended!",
"description":"",
"color_gallery_background":"#000000",
"UUID":"debug"}

Figure 4.1: Example Request and its Response

4.2.1 Registering and Customizing Beacons

To register a beacon, as mentioned earlier, all that is required for the server, is the beacon's UUID. You can register and customize a new beacon, or you can edit existing beacons via the *beacons* subpage, that shows a list of beacons registered on that service. Aside from the UUID, you can add other information of that beacon, and customize how it will look like on the beacon preview screen of the BLExpo client. A live preview is also on screen the whole time while editing beacons, to make the setup process more user friendly (Figure 4.2). If a new beacon is added with the same UUID as an already existent one, then the old beacon is replaced with the new one, which essentially is just editing a beacon with completely new data. However, beacons with the same UUID can co-exist on different BLExpo services, meaning beacons can be re-used easily, by just registering it on a new service. If using unique data on both services for a single beacon, a user can choose which data-set to access, by defining the server address of a specific BLExpo service that should be called to request beacon information. Thusly, the same beacon could offer different preview screens on the client, depending on which server a user is connected to.

Once a beacon has been registered to the service, it can be access by client requests. These requests do not only enable the communication between the clients and the server, but also allows for the existence of a logger.

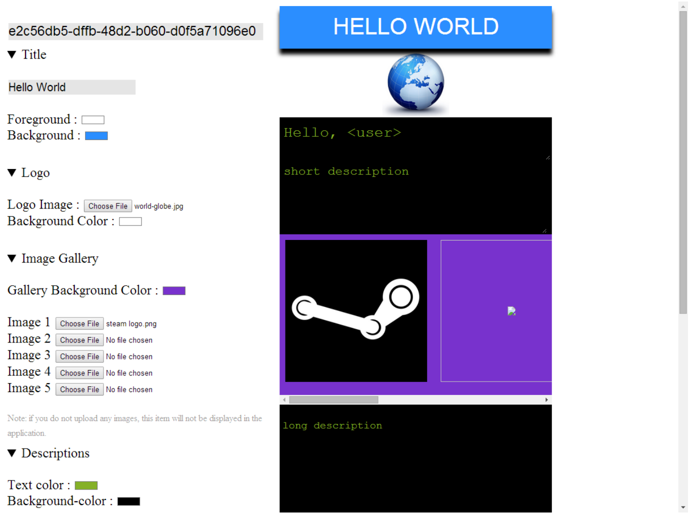


Figure 4.2: Beacon Customization Page

4.2.2 Event Log

By using a logger to capture every request, we open up a whole new world of possibilities. Thanks to Bluetooth LE, and willing users, it is now possible to see when a certain user visited a specific booth, and when the user left, by reading the timestamps of the logged events (Figure 4.3). We can not only see how many users visited a specific booth, but also when users were most active. We can look into the activities of the users, whether they are anonymous[1] or not, plays no role in the broader spectrum. This knowledge is invaluable, and can be used for user behavioural studies.

It does have one unique weakness, however. If a user does not want to use BLExpo client, we cannot say anything about that user, the service does not even knows he exists. So it is entirely up to the user to decide whether or not he wants to make use of the BLExpo service, and as a result be part of the user behaviour study that will most likely take place after each event.

4.3 Client Side

An Android application is used as the client, which also acts as a bridge between the server and the beacons. The client's job is simple: continuously scan for beacons, and once the user carrying the client device walks into a defined range[2] of a beacon that had previously been registered on the BLExpo service, make a HTTP request to that server using the beacon's unique identifier as a parameter. As previously described, the server will then respond with a JSON object, that includes the title, a welcome message, color formats, etc. that can be used by the client to relay the information to the user.

[1] Empty usernames will act as anonymous requests.
[2] The range can be defined by the user, although it will fluctuate, as described earlier.

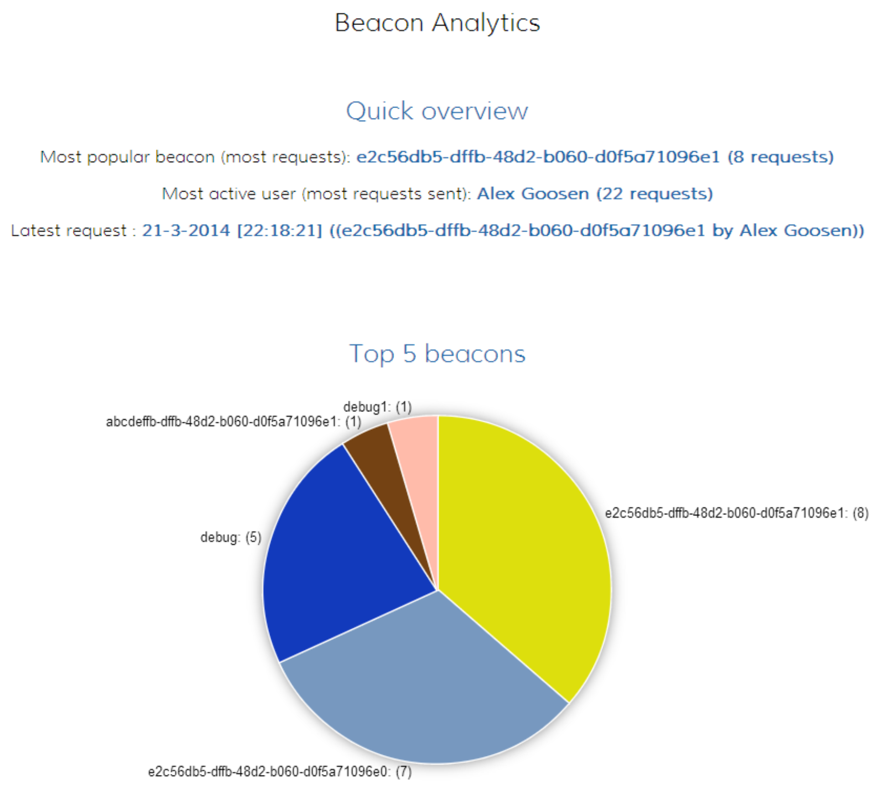
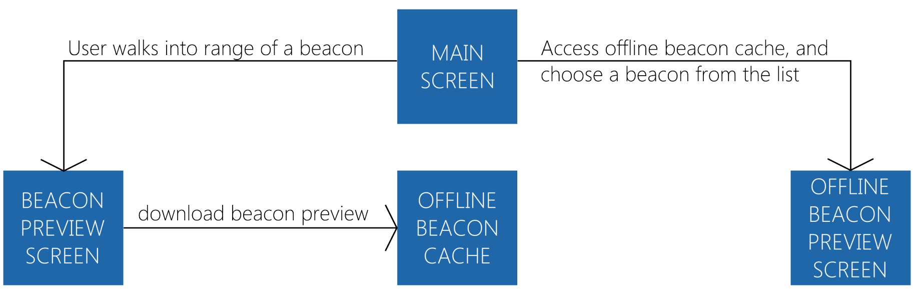


Figure 4.3: Analytics Page of a BLExpo Service

A new activity with a predefined layout is then launched, and after interpreting the JSON object, the layout can then be modified by the application and it can also start to download images if necessary. Part of that layout is an unchangeable download button, when pressed, saves the current layout parameters for later viewing. This happens instantly since the JSON object and images were already downloaded beforehand, and as such do not need to be donwloaded again. All the downloaded previews can be accessed via the offline beacons screen. It is possible to change certain preferences in the settings menu, such as which server to connect to, what username to use, and setting a beacon range. A simple connection tester is also built in that simply calls a default beacon, which should be available on every BLExpo service.



The client self was coded using the BLE API made public by RadiusNetworks [30]. This enables the application to scan for beacons once a second, and calculate their distance from the device using the RSSI that is sent with each advertisement. This distance, however, is not entirely accurate, and may fluctuate even if the device is not moved. It is accurate enough for the whole service to work, but should be used with caution when making applications that require precise distances. This way, the client never communicates directly with the beacons, but only receives data from their advertisements, which include the UUID as described in chapter 2.4. For simplicity's sake, only the UUID is sent as parameters with a request to the server, BLExpo does not make use of the minor or major values, although this could be added in future iterations. Since BLExpo relies on a seemingly new technology for the average person, it uses an introductory screen to acquaint users with the system and how it is supposed to work.

4.3.1 First Time Lauch

Upon launching the application for the very first time on a device, the user is greeted with a configuration setup, that asks the user to choose a username and server address, which can be edited later in the settings menu if needed. It is then briefly acquainted with BLExpo, how it works, and what the user needs to do, to use the service (Figure 4.4).

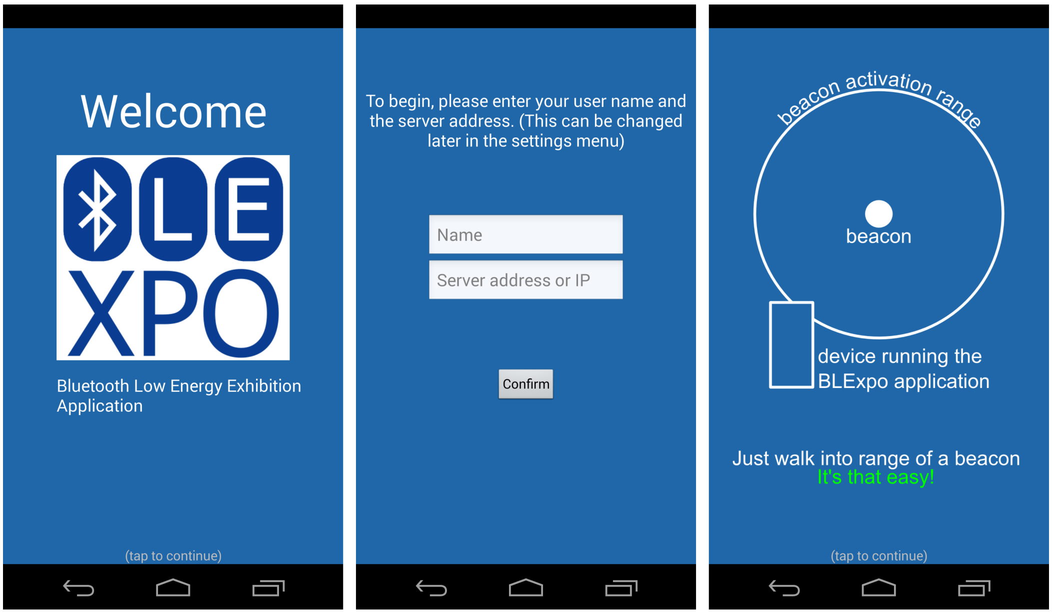


Figure 4.4: First Time Launch Screens

4.3.2 Main Screen

The main screen is a range indicator[3] that tells the user approximately how far it is from the nearest registered beacon, and also that it needs to move closer to that beacon to initiate its preview screen. Once that beacon has been seen, the client will then ignore that beacon until a different beacon is within activation range, as to not cause confusion when showing the distance to the closest beacon (Figure 4.5).

[3]As already mentioned, these ranges may differ from the actual distance to the beacon.

Figure 4.5: Main Screen

On the main screen, the user also has access to a menu, which grants the user ways of testing the connection to the server, listing all cached beacons saved for offline viewing, and also the link to the settings menu. Setting the beacon activation range, changing usernames or the server's address can all be done in the settings menu, as well as enabling or disabling vibration and toast notifications.

4.3.3 Beacon Preview Screen

More important than the main screen is the beacon preview screen, for it is the screen that connects the user to the beacon and the data stored on the server. In order to see this screen, the user must follow the instructions on the main screen, and walk into a beacon's close proximity. Its design is simple, and can easily be adapted and changed, if need be. For future iterations of BLExpo, additional preview screens may be added, and each beacon can indicate which screen it wants to display its information on. Currently there is only one preview screen, and it reserves space for:

- a title
- a logo
- a welcome message
- a short description
- a long description
- and an image gallery[4]

A predefined layout (Figure 4.6) is used to reserve space on the screen, and as soon as the client successfully receives and interprets the JSON object requested from the server, the layout is customized and then displayed. Images for the gallery are only downloaded after the layout

[4] If there are no images for the gallery, the image gallery will not be displayed on the client.

is created and shown. A download button is always present on the preview screen, and when pressed, saves the current preview to the internal application file directory. This means the beacon information can be viewed again later, by accessing it from the cached beacons screen.

Even under the lock-screen the application will still work. Because part of the application is running as a service, it continues to scan for beacons in the background, and can activate the beacon preview screen when a user walks into range of a beacon. The beacon preview screen is prevented from changing when the user is already viewing beacon information. So if a user wants to view a different beacon, he needs to go back to the main screen and walk into range of the beacon he wants to view next.

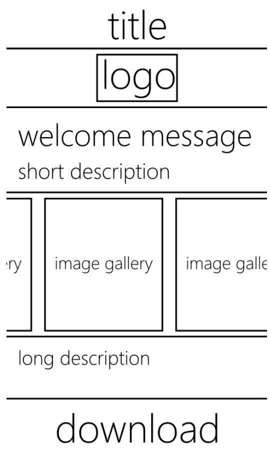


Figure 4.6: Preview Screen Layout

4.3.4 Offline Beacons Screen

Access to the offline beacons list is available via the menu on the main screen. The beacons saved on the device are listed in their respective titles and colors, and are saved using their UUID's as a primary key, which allows two or more beacons to share the same name and color combinations. If the user clicks on any of the listed beacons, he is taken to the already known preview screen[5], and will have access to all the previously shown information and pictures.

But, beacons are needed in order for the client and the server to ever communicate, and for the user to view them in the first place.

4.4 Passive Beacons

We need to differentiate between active and passive beacons. The latter would be the normal beacons that only send advertisements and nothing else, making the smartphone the active part of the system. Passive beacons used for BLExpo can be any kind of beacon as previously

[5]The download button is hidden now, since it is accessed offline.

described in chapter 2.3. Beacon registration and customization is handled on the server side of things, meaning that for a beacon to work with the system, all that is required, is to turn it on and register it. Assuming, of course, that the client is installed and set up correctly to connect to the server that the beacon is registered to. In BLExpo, Raspberry Pi's were used as passive beacons. In order to work as beacons, one script needs to be executed that reads the beacon's values from a configuration file, and starts to send the advertisements. The beacon data, which is limited to the UUID, RSSI, minor, and major values are all included in this file, and can be adjusted at will.

Although the BLExpo service currently only supports text, images, and colors on the beacon customization page, it is possible to expand and add more features, like videos and contacts, as long as it is also supported by the client. More advanced features would require *active beacons*.

4.5 Active Beacons

BLExpo active beacons are beacons that do not only send advertisements, but are also running web servers, that listen to incoming HTTP-Requests. Once a user walks into range of an active beacon, the normal process is initiated. However, alongside the server response, a new request is sent to the corresponding active beacon, to display whatever the beacon is set up to do, e.g. play a video, or display an image, or turn on a monitor.

Setting up an active beacon is much harder, since not every beacon might be suitable for it. You need a programmable beacon, and you need to know how to code. In the case of BLExpo, Raspberry Pi's were used to imitate a beacon to advertise BLE packages and simultaneously run a web server, which was written in Python. By sending a "sync"-request to the server when the beacon goes online, the IP-address of the beacon is saved for each session, and as such, the server can easily initiate commands when a request from a BLExpo client comes in, that requires an active beacon to do something. Every active beacon needs to be set up separately if they require unique actions. In BLExpo only one active beacon was emulated, and it displayed a picture on screen when a BLExpo client went into its range. Since active beacons can be set-up to do basically anything the developer wants it to do, it opens up a world of possibilities.

If a normal active beacon, like a Raspberry Pi, does not satisfy the computing power needs of the system, it can be emulated by using passive beacons. A beacon-computer pair can be set-up, whereby the computer takes care of the commands sent by the server, and relies on the passive beacon that is placed nearby to send the advertisements. Syncing will have to be done differently: the computer needs to be configured to send the "sync" request with the nearby beacon's UUID as a parameter, which will make the server believe that the IP-address of the computer belongs to the beacon. Using USB beacons for this situation would be ideal.

5 Possibile Issues and System Limitations

During the design and development of the BLExpo system, it was assumed that every visitor to a BLExpo supported exhibition has a Bluetooth Smart device, and is willing to turn on BLE and use the application as intended. They also have a stable internet connection, whether it is via 3G, 4G or Wi-Fi. But because we do not live in a perfect world, there will always be things that do not go according to plan. Assumptions do not reflect reality, and BLExpo might run into a few problems when run in the real world. There are two distinguishable kinds of problems that can be encountered when using BLExpo at conventions. One is a limiting factor, or user requirements for the system. The second is problems that mostly occur on the side of the system, and can be solved, but they may not be encountered every time.

5.1 Limitations

There is a limit on the amount of users such a system can have, dictated by the devices and technologies available to users, and their mindsets.

5.1.1 Device Limitations

The most obvious limitation is the Bluetooth low energy requirement for mobile devices. If a device does not support BLE, it cannot scan for advertisements, and as a result, it is impossible for the device to connect the user with the service. This requirement can only be overcome by using a more modern device. However, this should not pose to be such big of a problem in the future, since an estimate of over 90 percent of Bluetooth enabled devices would also support BLE by the end of 2018 [31]. Any modern smartphone bought today, probably already has support for BLE anyway, such as iPhones 4 and 5, any Android 4.3 or later device or any of the new Windows Phone 8 handsets[1] [32].

Since the client is currently only available as an Android application, it is also a definite limitation. However, since the objects retrieved from the server are just simple images and JSON objects, it enables the creation of applications run on operating systems other than Android, to be used with the same service.

[1] Although Windows Phone has BLE integration, it currently does not have an open API for developers to create applications that use BLE, and thusly it might be possible, later, for Windows Phones to run applications such as BLExpo.

5.1.2 End User Limitations

There are also other limitations to the system that might not be so obvious at first glance. Such as users who are unwilling to use the service because of privacy concerns. In the day and age we live in now, privacy has become a very big concern for the majority of citizens from any country, since it has come out that the formerly trusted NSA and other national security groups, had been spying on normal citizens [33]. This might scare users away from using technologies such as Bluetooth Smart to scan for beacons, since the user's location can be pinpointed within a few metres, without the help of a GPS. If GPS sensors were to be used for outdoor, and beacons for indoor tracking, we might not be able to "hide" from the system without turning off features such as BLE and GPS sensors. So even with such great technological advances, people might not even bother using them.

In the BLExpo service, a logger is used, as described in chapter 4.2.2. And in the settings menu, the users are warned that they might be tracked and identified by the username they use in the client application. Although the intentions for this tracking are pure, and are there to optimize convention layouts, it only takes one person to misuse the tracking information, gathered by the logger on the server, for malicious purposes. The only way not to be tracked by the service, is to not use it at all, unfortunately.

5.2 Possible Issues

Imitating beacons can be done easily, but there is not really a point to it, other than to cause a minor annoyance. To imitate a beacon, you need a beacon yourself, and set it up with the same UUID as the beacon you want to imitate. And since it is possible to scan all UUID's using an Android phone[2] you can imitate any beacon you found. The annoyance it creates is very minimal, since it tells the server that a user is at a certain location, but in reality that user is somewhere else. This will also confuse a user who sees that a beacon activated the preview activity on his phone, but he is nowhere near the booth. There currently is no way around this problem, since the beacon's advertisements are public, and thusly, can be cloned very easily.

Apple, on the other hand, currently does not support wildcard scanning, meaning that the UUID's of beacons have to be known beforehand [34]. Since we need to know the UUID, we can define one for the whole event. By using only one UUID, it limits the total number of unique beacons by a vast amount, becuase the only differences these beacons have, are the minor and major values. Even so, it leaves room for around 4 000 000 unique beacons, which is not that big of an issue then.

A greater issue that the system might have to face is frequency interference. As described in chapter 2, BLE uses the 2.4GHz radio frequency band, which is shared by many other devices

[2] For example, you can use the iBeacon Locate application by RadiusNetworks, or just code your own app that does the same.

such as Wi-Fi stations[3], microwave ovens, video transmitters and receivers, wireless speakers, certain external monitors and LCD displays, some power cables, etc. Any device that operates in the 2.4GHz - 2.5GHz frequency may cause radio interference with Bluetooth, and may result in connection issues. Placement of the beacons become important when considering radio interference. Not only other devices, but the surroundings may also be a source of interference [35].

MATERIAL	POTENTIAL INTERFERENCE
Wood	Low
Bricks	Medium
Concrete	High
Metal	Very High

Figure 5.1: Potential Sources of Bluetooth Interference

It should be obvious to place a beacon at a location where it would be less affected by radio interference, if at all possible. For instance, not placing it in something with a metal frame, or on the backside of a TV, as this might cause the distance tracking to become more inaccurate and thusly unreliable. Estimote recommends user to place their beacons above obstacles, such as crowds, in order to keep a clear line of sight between the mobile device and the beacon, as to minimize interference [36]. Also keep in mind to place beacons a set distance away from each other, as to not confuse the user who receives a preview screen for a certain beacon, but who is in reality standing at another beacon and was awaiting the preview screen of the latter.

Currently in BLExpo there is no implementation of a user base, which will be required for using this or similar systems in the actual world, for security reasons. Right now, anyone with access to the network can add, remove, or edit beacons. The server needs to be further implemented to support the needed features. For BLExpo, it was imagined that companies, who reserved a booth at a convention, only have access to beacons they registered.

Possibly the greatest limitation is BLExpo's dependence on a wireless network, to which all the active beacons, clients, and the server is connected to, in order to work properly. It can only be partially solved by using a remote server instead of relying on a local network, but that opens up new issues such as the internet requirement for users. Instead of trying to resolve the choice between a local and remote server, a different architecture can be imagined. A purely BLE system, that does not use a server.

[3]Or any other device that uses Wi-Fi.

6 Purely Smart System

Cutting out the underlying network connection between the client and the server could simplify the system, since there would be no need for a sever anymore (Figure 6.1). Communication, in this case, would be done purely via Bluetooth, where beacons and clients transfer data directly, instead of clients requesting data from the server. In this kind of setup, beacons have to be configured separately, since there is no server where beacons and their data can be saved on, the beacons themselves need to carry their data. When a client comes into range, a Bluetooth connection is opened between the client and the beacon and the transfer can begin. Although this system removes the need for a server, and thusly eliminates the necessity for users to supply a server address their clients have to connect to, it has some major implications that pose a threat to its real world implementation (See section 6.2).

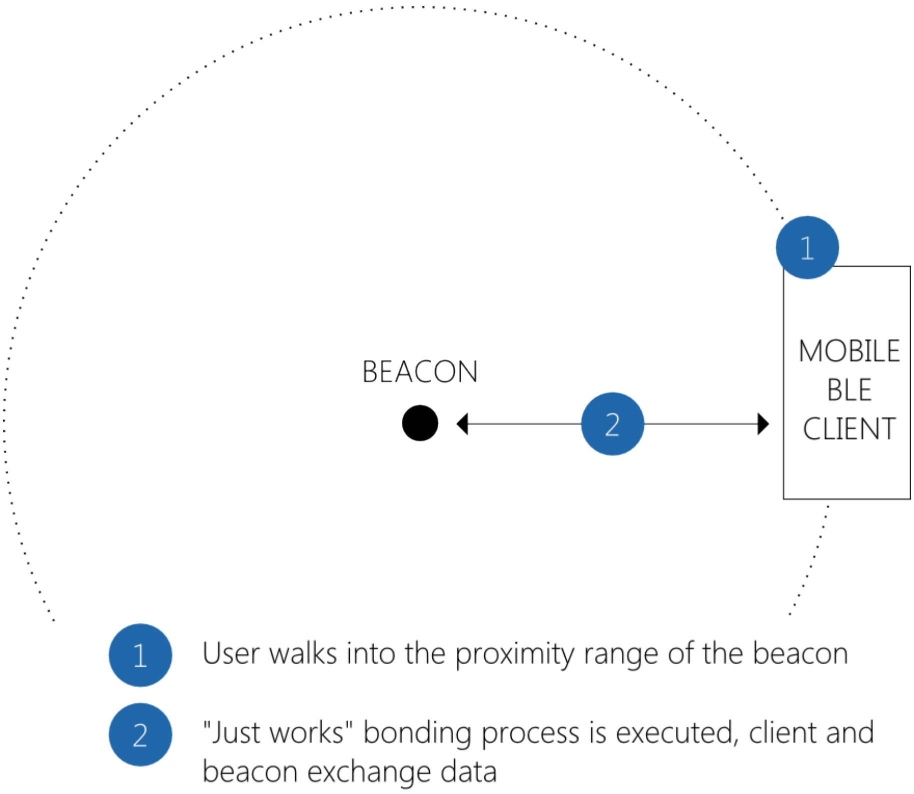


Figure 6.1: Purely Smart System Architecture

In a perfect world, if this system worked as intended, it would prove to be more user friendly towards users, since all that needs to be done on the their side is to activate BLE on their devices. Gathering user data is still possible, by either looking at each beacon's data separately, or by having all the beacons connected to a private network that upload their data every once in a while. This back end is only used by the beacons, and the client will never connect to this server. Because of the lack of a central server and service, where every beacon can be registered, each company that wants to take part in the exhibition must now create and set up their own beacons, on their own time.[1] It clearly has some drawbacks when trying to realize the BLExpo system with this architecture, but it might carry some advantages for other systems.

Systems that are implemented in areas that have a weak or complete lack of cellular network coverage would benefit the most from this kind of system architecture, since an internet connection is not required on the client.

6.1 Realization

Although using Bluetooth in its current state for this type of system architecture seems viable, there are several issues that can prevent its realization. Beacons, as described earlier (see chapter 2.3), are simple, non-connectible BLE devices that only sends out advertisements. You can, however, make beacons connectible in order to communicate with them via Bluetooth, like with the new PayPal beacons that act as a bridge between the server and the user, by forwarding requests and responses [37]. But, because Bluetooth 4.0 was optimized for low energy rather than data transfer speeds, an issue arises. It may still be possible and viable to send small amounts of data via BLE, but as soon as other types of data, like images or videos, have to be transferred or streamed, a faster connection is required. Managing file transfers and maintaining connections with the classic Bluetooth protocol instead of version 4.0 defeats the whole point of the system being low energy, and would also mean that beacons will not last as long compared to the ones that only use BLE.

Transferring data, at least on Android, requires two devices to be bonded beforehand. The most obvious pairing process that could be implemented here is the "just works" method, where minimal user interaction is required. As soon as the beacons wants to pair, the user has to either accept or decline the pairing [38]. Since users would already be active on their mobile devices when asked to bond, it is not as big an issue for one beacon. But, when an interaction from the user is needed for every beacon in order to send and receive data, it becomes quite user unfriendly, and might even stop users from using the application any further.

As mentioned in chapter 2, Bluetooth uses a master/slave architecture for the communication between devices. A master device would be something like a smartphone, and a slave device the beacon. Since the smartphone does the scanning and initiates a connection to the beacon, it takes on the master role. This leaves the beacon as the slave in the communication process. However, since slaves can only have one active connection, it implies that when two or more users are in range of a certain beacon, only one of them can actually get the data from the

[1] Assuming they know how. But a program can be created to aid in this endeavour.

beacon, leaving the second user waiting. Reversing the master and slave roles for the two devices resolves this issue, but it still severely limits the amount of users that can stream data from a beacon at the same time.

Even if all the above issues were resolved, this kind of system architecture only seems applicable for systems that can, by no means, make use the client's internet connection for data transfers. Beacon data can be changed from a central server to which all of the beacons are connected to. This limits the amount of beacons that can be used within the system, which then also prevents its expansion. Besides, according to current trends, more and more people are visiting websites and requesting data on the go from their mobile devices. As a result, no internet connectivity or weak cellular network coverage, even in buildings, might only affect the minority of BLE users in the future.

6.1.1 Alternative Purely Smart System

Another alternative does exist, if relying on a network is out of the question, and a simple and quick[2] solution is sought. Creating a completely new architecture that only uses BLE to scan for beacons, means scrapping all of the previously described systems. And, like a pure BLE system, it is also server-less, but still needs the beacons and the client. Everything, from the beacons UUID's to the layouts configurations used for each preview page, are included in the client this time. On this account, no data transfers are needed anymore, because the client already knows which data to relay to the user once a beacon is in its range. Neither a network connection, nor a Bluetooth connection is needed, and the ability to instantly show the preview page gives this system an edge over the other two, but in spite of that, there are clear drawbacks.

The lack of a central location to add and edit beacons means that every time, even the smallest adjustment to a beacon has to be made, the application needs to be updated. It also means that the application cannot be re-used for different events, instead, for every event a new application is made and released that uses data relative to that event, since it is all hard coded into the client. Companies that want to set up a beacon, cannot do it without speaking to the developers of the client prior to the event, and they cannot fully customize the layout to the degree that the BLExpo website offers.[3] Although this kind of system might be logical for non-dynamic applications and smaller events, where there are only a few beacons, it is definitely not viable on a larger scale.

[2] It is only quick if the whole system (including the server and the client) also needs to be implemented, otherwise just adopting and making minor changes to an existing system might be quicker.

[3] An online service could be offered to companies where they can use the BLExpo beacon customization page, and the data is gathered by the developers who include it in the application.

7 Future Expansions

Using beacon technology, it is possible to create a wide variety of systems. BLExpo is just one system in a whole new world of possible applications.

7.1 Dynamic Content

Although BLExpo does not offer dynamic content in its truest form, it does allow registered beacons to be changed on the fly. Meaning two requests for the same beacon may yield different results. Also, using user accounts that are found on the mobile device can provide the application with useful knowledge about the user. Content can be selectively changed, based on the user's preferences. A simple example of this would be that a user adds an item to his wish list, on a mobile shopping application, and when the application recognizes a beacon that is placed near that item in a real shop, the application will notify the user that the item is right in front of him. Or the exact opposite could happen. If the application knows that the user is not interested in a certain product, then it will make no attempt of notifying the user. Since the data that is shown to users can be catered to them, it allows applications to deliver more personal user experiences.

A small amount of data can be sent via the UUID's that beacons advertise. Instead of using an actual UUID, the data is encoded to Hex format and is used as the first part of the UUID. Android devices that have access to wildcard beacon scanning can pick up such advertisements and decode them for the user. Although the data that can be sent using this method is very limited (16 bytes, which is the length of the entire UUID) it can be adapted on the advertising beacon. It can change the UUID accordingly in respect to the information it is monitoring, and restart to advertise with the updated information. Since scanning devices might not be able to tell whether the data is coming from the same beacon, a few bytes of data can be dropped from the advertisement in favour of an identifier, or the minor and major values can be used for identification. Duplicating another beacon's UUID using this method is very unlikely, and should be of no concern (see chapter 2.3.1: UUID). Applications using this kind of dynamic content could include devices fitted with temperature sensor that report the data back to the user.

7.2 Global Expansion of BLExpo

BLExpo is just a simple example of how proximity based systems can be implemented and used. As described earlier, it only uses one preview screen, but it should be noted that it is possible for the system to be adapted to execute different actions based on the response coming from the server. Instead of showing a preview, the client application could instead load a web page or even launch a different application.

A much wilder and global expansion of BLExpo could see Apple, or Google, make a cloud based repository where all the beacon UUID's are saved. Bluetooth Smart and beacon systems would be more user friendly, since the repository would then be used to look up scanned UUID's and afterwards launch the respective application on the user's device. Other possible events may be possible, such as enabling a setting, like turning on Wi-Fi if a user walks into region with free internet access, or launching web pages. Even without directly building the BLE scanning into the operating systems of mobile devices, a third party application that would be very similar to the BLExpo client, can take over the role of assigning scanned UUID's to other applications and launching them when needed. Groups, individuals, or companies, who like to make use of the global beacon repository, can request a UUID to be registered and provide a unique ID of the respective application that will be launched when a consumer is in range of the UUID. The application itself then communicates with other servers in order to obtain the necessary data that is associated with the scanned beacon.

Since the newly released iOS7.1 update, Apple devices offer this kind of functionality built into the operating system. In contrast to the previous version of iOS, it now allows applications to scan for beacons in the background [39]. When an application recognizes a beacon's UUID, it is launched, or brought to the foreground if it was already running. Because iOS applications have to be provided with a UUID in order to start scanning, unknown beacons will be ignored. RadiusNetworks, on the other hand, offers wildcard scanning on Android devices with their API, meaning they can scan UUID's that are not associated with any installed application. Using the global repository, beacons have the possibility of triggering events on Android devices, such as opening the web browser with a specific URI. Of course, only with the user's permission. The functionality that this global repository grants, is to enable mobile devices to receive notifications and beacon events without the need of installing an application beforehand.[1] To prevent malicious events from occurring, such as automatically opening websites with dangerous content, users should be able to report such beacons, so that only trusted beacons[2] will trigger events. Additionally, the application's settings has to be adjustable in such a manner, that users have the option to be notified and asked whether or not the event should be triggered. Out of privacy concerns, user data can stay anonymous, the global repository can, however, still track the amount of requests that are made for each and every beacon, and whether or not the event was successfully triggered on the user's mobile device. Beacon requests can be analyzed to optimize application performance and the usefulness of certain beacons in a system.

The global repository does not replace small scale beacon based systems, it instead aids them in a very unique way. Assuming a certain user does not have the needed application to scan and receive beacon data from within a small beacon ecosystem, a request can be sent to the global

[1] If it is built into the OS, otherwise at least one application is needed to request beacon events from the cloud.
[2] In this case, trusted beacons are the ones that are registered to be on the global repository.

repository to retrieve a link to that application on the Playstore, or it can open a website that tells the user that an enhanced experience is available, and provide a link. On the other hand, if the respective application is already installed on the device, it can be opened and optionally brought into focus. It can also be possible that there is no assigned application, and the beacons are purely used for additional information or data via websites. Such a beacon based system will not be as rich as an application, and will most likely also be less sensitive of the user's location. The beacon repository, like BLExpo, will only make use of UUID's and not major or minor values, since the application associated with the UUID can make use of those values to relay proximity based data to the user.

Even here, dynamic content can be delivered to the users. Websites can provide time based information. For example, movie showing times based on the time the request was received. This content is not user based, since everyone requesting that beacon via the global repository will receive the same information. An application, that is launched or brought into focus, can filter content based on user preferences, like only showing movies of specified genres.

7.3 Role Reversal

As it stands right now, in every beacon based system there are two main devices, one that advertises, and one that does the scanning and upon UUID recognition will trigger some event. Reversing the roles of these two devices can lead to some very interesting applications. The beacon now does the scanning instead of advertising, which is taken over by the smartphone or any other capable mobile BLE device. Since the system was inverted, the event triggering happens on the beacon and not on the user's device, like it is emulated in BLExpo's active beacons. However, unlike BLExpo, nothing happens on the smartphone, since it is only advertising itself to the scanning beacons. This allows the system to replace smartphones with something smaller, more efficient, like bracelets, or building BLE capabilities into watches. These wearable advertising devices rarely leave the user's side, and will have a unique ID (UUID) associated with them. Having this UUID means that every user can be identified by their advertiser, and scanning beacons can trigger or allow events to be triggered when certain UUID's are in range. The scanning beacons, on the other hand, will most likely be replaced by devices with more available resources to allow the system to work fluently, and not be hindered by the hardware. It really depends on the system requirements whether or not more powerful scanning beacons should be used.

All the systems described earlier would most likely not always benefit from the reversed beacon role, but it facilitates the design and creation of others. BLExpo, for example, would be unimaginable using only scanning beacons, since only the active beacons would benefit from it, but not the passive beacons. A system that would gain greatly from this role reversal is the Smart Home.

7.4 Smart Homes

In Smart Homes, many electronic devices are connected to each other in order to automate tasks for the owner's convenience. A central computer offers a user interface to control those devices, and it comes equipped with a web server to enable remote access [40]. By using an application on a smartphone or tablet, a Smart Home owner can dim the lights, set the temperatures of certain rooms, or even set the oven to cook his food while he is still in another room. Not only does the owner have access to all the devices, but some tasks can be taken over by the home completely, like automatically turning on the air conditioning to cool down the house on a hot day. Beacons can aid in the home automation aspect. Wearable advertisers, or smartphones set to emulate beacons, can tell the home exactly in what room the user currently is in. With this knowledge, a Smart Home can detect if the owner is out, and thusly take control of certain electronics, like turning off televisions, to save energy, or ovens, to prevent potential fires. It can unlock the door when the user approaches the front door, or illuminate rooms as the user walks through the house.

Of course, a BLExpo based system in homes is also possible and beneficial. Your smartphone can be set up to show specific screens (or run certain applications) in certain rooms, to control different appliances or electronics, either directly or via the central computer.

Like NFC, beacons can also be used to update your social status. You can set up an application to update a certain mood or status on your favourite social network just by tapping, or holding your device near the beacon, like you would with NFC. The same beacons that are used in conjunction with other applications, can be used to update social statuses. For instance, the beacons used with "At The Ballpark" could also be used by a Facebook application to tell your online friends that you are attending the game. In Smart Homes, applications could automate social status updates, such as what you are watching on TV, what you are cooking, and more.

7.5 The Internet of Things

The internet of things (IoT), proposed by Kevin Ashton in 1999, is an idea that all things are uniquely identifiable, and can communicate with each other without the need for human interaction [41]. With beacons offering a very wide range of possible UUID's that can be assigned to everyday objects, all that is needed is the communication. Bluetooth Smart 4.1 allows BLE enabled devices to use IPv6 for connectivity, meaning that such devices can identify themselves and communicate over the internet directly, without relying on a smartphone [42]. In Smart Homes, the IoT provides the communication aspect via Bluetooth Low Energy, that can move data in a mesh-like topology without maintaining an actual mesh network [43]. According to ABI Research, about 30 billion devices will be part of the IoT by the end of 2020, of which Bluetooth Smart is recognized as the key enabler [44]. Having beacons in the IoT can prove beneficial, since they allow other devices to know your whereabouts. For us as consumers, the IoT exists to improve our lives by sharing information with other devices in the background, and by connecting themselves seamlessly to our world, purely for our convenience.

7.6 Indoor Mapping and Trilateration

Beacons offer us the unique ability to improve mapping applications by adding indoor maps. The beacons, in this case, have to stay at fixed locations within an establishment for the application to maintain its accuracy. The maps used can aid the user by including additional information, such as arrows pointing in a certain direction of how to get somewhere the user queried, or special points of interest unique to the user's location. If a user, for example, is looking for the nearest restroom, he can obtain a map with a highlighted path that shows him how to get there. While GPS sensors can aid us in the outside world, they mostly do not work indoors, which is where beacons can take over the role of providing the user's location.

There are two possible ways to create a application with mapping capabilities by using beacons. The first, and most simple way, is to use only one beacon that requires a user to come withing close proximity, e.g. one metre, to see his location on the map. Evidently, the closer the proximity requirement, the more accurate the location will be. However, it does leave much room for improvement, since the user's location will always be fixed to certain points. Pre-rendered images can be used to show the user's location depending on which beacon he uses to get the map. This method can be used by systems which do not have access to many beacons, or that do not require a live map.

A more complex way of implementing maps in applications would be trilateration. It is the process of determining one unique location by measuring distances, using the basic geometrical attributes of circles [45]. Having only one beacon leaves the application guessing where the user is, since it can only limit the user's location to a cirlce around the beacon. Adding another beacon limits the amount of possible locations to two points, because both distances towards both beacons are known. At least three beacons are needed to calculate the one exact location of the user (Figure 7.1).

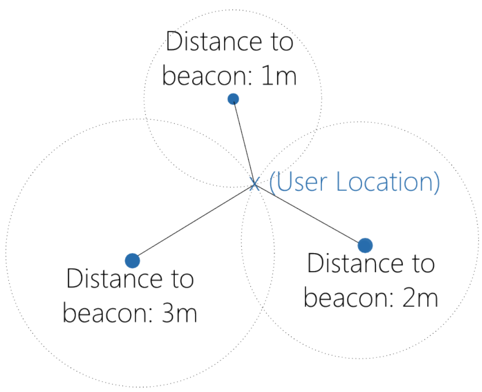


Figure 7.1: Trilateration with Beacons

In reality, however, the user's location might not reflect his actual real world position, because of distance measurement fluctuations caused by both the distance calculations and wireless interference. More research is needed to determine whether beacon trilateration will suffice for indoor navigation systems. If it does, then applications similar to Google Maps can be created, but with indoor maps. Applications, like AREA [46], that rely on distances between the user and an (identifiable) object, may make use of beacons for both the distance measurement and object identification for indoor use.

Geocaching, a popular part-time activity that is played using the GPS to find hidden caches around the world [47], can be played indoors using beacons. BLExpo, for example, could include an award system where users are awarded a virtual medal or trophy, for visiting booths that do not get a lot of visitors, to attract more people.

8 The Reality of the Situation

The birth of a new technology is always exciting, and leaves room to ponder whether it will survive in a world filled with electronic gadgets. Bluetooth is a special case, since it is not really a new technology, but has made very unique advances that lead up to Bluetooth 4.0. With its low energy specification, we can now leave Bluetooth running all the time, and not worry about draining the battery. In addition to the low power consumption, BLE enables developers to create unique and interesting systems that make use of billions of existing Bluetooth devices, and future ones yet to be invented.

Beacons, however, are real; they exist and are already being used by companies such as Apple and MLB. Mixed with the technologies available today, we are capable of creating unique and never before seen systems, that can revolutionize the ways we live, and the ways we use technology, as individuals and as a community. The future is becoming reality, and Bluetooth Smart is only one stepping stone on our way to live like the fictional characters in science fiction movies. Along with all the conveniences Bluetooth Smart offers, comes the impact it potentially has on the health and fitness industries [48]. Doctors of the future can remotely monitor a patient's heartbeat. Even we, as normal every-day people, can put on a heart rate monitor and connect it with our smartphones via Bluetooth Smart to monitor our performance as we go for a jog.

The BLExpo service implementation serves as a prototype for beacon based systems that want to deliver proximity based information to the consumer. Using both passive and active beacons, the BLExpo service can make use of the user's proximity to a beacon (passive), and the beacon's proximity to a scanning device (active). When using passive beacons, the client's role is to relay information from the server to the consumer, whereas with the role reversal in play, the active beacons can be addressed to trigger events. The beacon information hub, where all the beacons can be registered and customized, also serves as the central station from where the information is requested by the client, which can be tracked to offer developers and system administrators an insight as to how often a certain beacon is requested, and user statistics. BLExpo is faced with only a few limitations that can be easily overcome. Every part of the system can be adapted to offer different functionalities if required.

Bluetooth Smart offers endless possibilities when it comes to designing and creating proximity based systems. BLExpo is just a simple example of one such system, and it can be adapted and expanded to a global scale, in homes, at conventions, or wherever it is needed. Even systems that do not rely on active internet connections can exist, and offer the same functionality. We are able to use beacons in both passive or active manners, which creates a whole new dimension for beacon based systems, since not only can we interact with beacons, but they can also interact with us. We might, one day, all be wearing bracelets or have beacons implanted in us, to make us, humans, part of the internet of things.

By allowing Bluetooth Smart devices into our lives, we are also accepting the fact that developers can make use of our location history, for personalized applications. Advertisements can be aimed directly at an individual, which definitely has it advantages, but should be used with caution. If an application misuses beacons to only spam the user's device with advertisements, then it will undoubtedly fail. Beacons offer developers the unique chance to create amazing systems, and they should take advantages of them, to in turn, design extraordinary system they can deliver to consumers. Your smartphone can turn into a TV remote by simply entering the living room. And since everything is connected, the lights can be dimmed to enhance your TV watching experience, once you have interacted with the remote. A child can be stopped from watching a PG rated film if no adult is in its vicinity, your oven can recommend recipes based on the items in your refrigerator, the amount of systems that can be designed using BLE is countless.

The only real problem that beacon based systems face, is the essential internet connection on client devices. But, nowadays having no internet on your smartphone is a rarity, so it should not pose too big a threat in the future when even more coverage is provided by mobile networks that offer faster and faster mobile internet as time goes on. Since newer devices already support BLE, and devices not yet released will most likely support BLE, it is safe to say that the underlaying technology is here to stay. The only people who cannot make use of this technology, are the ones that have Bluetooth turned off.

The reality of the situation, as it stands, is that the stage is already set for Bluetooth; all modern smartphones and tablets come equipped with Bluetooth Smart Ready chips, and it will be available on devices for years to come. We can already buy BLE gadgets, such as wearable heartbeat sensors, smart watches and even shoes with BLE integration, and of course, beacons. This means, that we, as developers, can create systems with features that were unheard of before. And we, as consumers, should make use of these new systems for it to further develop. Ultimately, it all comes down to us. We are only standing at the starting line and looking at the track that lays open before us. Beacons exist, and so do devices that can intercept their signals. If we are willing to embrace these systems and integrate them into our day-to-day lives, then our future, like Bluetooth's, will also be Smart[1].

[1] Pun intended.

List of Figures

Bibliography

[1] ROHDE, Laura: *Ericsson demos first Bluetooth phone - PC Advisor*. http://www.pcadvisor.co.uk/news/desktop-pc/100/ericsson-demos-first-bluetooth-phone/, 2000. – [Online; accessed 22-April-2014]

[2] *Bluetooth Smart Devices | Bluetooth Technology Website*. http://www.bluetooth.com/Pages/Bluetooth-Smart-Devices.aspx, 2013. – [Online; accessed 11-May-2014]

[3] *Bluetooth Smart | Bluetooth Technology Website*. http://www.bluetooth.com/Pages/Bluetooth-Smart.aspx, 2013. – [Online; accessed 22-April-2014]

[4] GOTTIPATI, Hari: *With iBeacon, Apple is going to dump on NFC and embrace the internet of things - Tech News and Analysis*. http://gigaom.com/2013/09/10/with-ibeacon-apple-is-going-to-dump-on-nfc-and-embrace-the-internet-of-things, 2013. – [Online; accessed 22-April-2014]

[5] *How iBeacon Works | Mobile Wallet Cloud Solutions | PassKit.com*. http://passkit.com/ibeacon-devices/how-ibeacon-works/, 2013. – [Online; accessed 22-April-2014]

[6] TUNG, Liam: *Apple launches iBeacon in 254 stores to streamline shopping experience | ZDNet*. http://www.zdnet.com/apple-launches-ibeacon-in-254-stores-to-streamline-shopping-experience-7000024026/, 2013. – [Online; accessed 22-April-2014]

[7] O'GRADY, Jason D.: *iBeacons: Coming soon to a baseball stadium near you | ZDNet*. http://www.zdnet.com/ibeacons-coming-soon-to-a-baseball-stadium-near-you-7000021354/, 2013. – [Online; accessed 22-April-2014]

[8] *MLB.com At the Ballpark on the App Store on iTunes*. https://itunes.apple.com/us/app/mlb.com-at-the-ballpark/id513135722?mt=8, . – [Online; accessed 22-April-2014]

[9] *Fast Facts | Bluetooth Technology Website*. http://www.bluetooth.com/Pages/Fast-Facts.aspx, 2013. – [Online; accessed 04-May-2014]

[10] *BBC NEWS | Technology | Bluetooth rival unveiled by Nokia*. http://news.bbc.co.uk/2/hi/technology/5403564.stm, 2006. – [Online; accessed 02-May-2014]

[11] GRABIANOWSKI, Ed: *HowStuffWorks "Is Wibree going to rival Bluetooth?"*. http://www.howstuffworks.com/wibree.htm, 2006. – [Online; accessed 02-May-2014]

[12] *Basics | Bluetooth Technology Website*. `http://www.bluetooth.com/Pages/Basics.aspx`, 2013. – [Online; accessed 04-May-2014]

[13] *BLE Overview - Laird Technologies Wireless Connectivity Blog*. `http://www.summitdata.com/blog/ble-overview/`, . – [Online; accessed 02-May-2014]

[14] *Bluetooth Low Energy | Android Developers*. `http://developer.android.com/guide/topics/connectivity/bluetooth-le.html`, . – [Online; accessed 03-May-2014]

[15] *Specification | Adopted Documents | Bluetooth Technology Special Interest Group*. `https://www.bluetooth.org/en-us/specification/adopted-specifications`, . – [Online; accessed 03-May-2014]

[16] *Press Releases Detail | Bluetooth Technology Website*. `http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=138`, 2011. – [Online; accessed 02-May-2014]

[17] *Bluetooth v4 und Bluetooth Low Energy (BLE) Übersicht | Fitness Modern*. `http://fitnessmodern.de/bluetooth-v4-bluetooth-low-energy/`, 2014. – [Online; accessed 15-May-2014]

[18] GALEEV, Mikhail: *Bluetooth 4.0: An introduction to Bluetooth Low Energy-Part I | EE Times*. `http://www.eetimes.com/document.asp?doc_id=1278927`, 2011. – [Online; accessed 02-May-2014]

[19] *Bluetooth - Wikipedia, the free encyclopedia*. `http://en.wikipedia.org/wiki/Bluetooth#Pairing_and_bonding`, . – [Online; accessed 03-May-2014]

[20] *estimote.com*. `http://estimote.com/`, . – [Online; accessed 28-April-2014]

[21] YOUNG, David G.: *Radius Networks - Reverse Engineering the iBeacon Profile*. `http://developer.radiusnetworks.com/2013/10/01/reverse-engineering-the-ibeacon-profile.html`, 2013. – [Online; accessed 28-April-2014]

[22] *Radius Networks - Virtual Beacon*. `http://developer.radiusnetworks.com/ibeacon/virtual.html`, . – [Online; accessed 05-May-2014]

[23] NEBEKER, James ; YOUNG, David G.: *Radius Networks - How to Make an iBeacon Out of a Raspberry Pi*. `http://developer.radiusnetworks.com/2013/10/09/how-to-make-an-ibeacon-out-of-a-raspberry-pi.html`, 2013. – [Online; accessed 05-May-2014]

[24] NEBEKER, James: *Radius Networks - How To Make a Raspberry Pi Turn on a Lamp with an iBeacon*. `http://developer.radiusnetworks.com/2013/10/01/reverse-engineering-the-ibeacon-profile.html`, 2014. – [Online; accessed 05-May-2014]

[25] *Radius Networks - RadBeacon USB*. `http://www.radiusnetworks.com/ibeacon/radbeacon/`, . – [Online; accessed 05-May-2014]

[26] *Universally unique identifier - Wikipedia, the free encyclopedia*. `http://en.wikipedia.org/wiki/Universally_unique_identifier#Random_UUID_probability_of_duplicates`, . – [Online; accessed 05-May-2014]

[27] *Amazon.de - Bluetooth Dongle*. `http://www.amazon.de/s/ref=nb_sb_noss/276-1561267-7872732?__mk_de_DE=%C3%85M%C3%85%C5%BD%C3%95%C3%91&url=search-alias%3Daps&field-keywords=bluetooth+dongle`, . – [Online; accessed 05-May-2014]

[28] *The Global NFC Tag Superstore | GoToTags NFC Store*. `http://www.buynfctags.com/`, . – [Online; accessed 05-May-2014]

[29] *Radius Networks - Android API (Under Range Notifier*. `http://developer.radiusnetworks.com/android-ibeacon-service/doc/`, . – [Online; accessed 17-April-2014]

[30] *Radius Networks - Android iBeacon Library*. `http://developer.radiusnetworks.com/ibeacon/android/download.html`, . – [Online; accessed 14-April-2014]

[31] *Mobile Telephony Market | Bluetooth Technology Website*. `http://www.bluetooth.com/Pages/Mobile-Telephony-Market.aspx`, . – [Online; accessed 14-April-2014]

[32] *Bluetooth low energy - Wikipedia, the free encyclopedia*. `http://en.wikipedia.org/wiki/Bluetooth_low_energy#Operating_systems`, . – [Online; accessed 14-April-2014]

[33] *NSA spies on more US citizens than Russians – Snowden — RT News*. `http://rt.com/news/156088-nsa-spies-americans-russians/`, 2014. – [Online; accessed 11-May-2014]

[34] YOUNG, David G.: *Radius Networks - CoreBluetooth Doesn't Let You See iBeacons*. `http://developer.radiusnetworks.com/2013/10/21/corebluetooth-doesnt-let-you-see-ibeacons.html`, 2013. – [Online; accessed 19-April-2014]

[35] *Wi-Fi and Bluetooth: Potential sources of wireless interference*. `http://support.apple.com/kb/ht1365`, . – [Online; accessed 15-April-2014]

[36] STECZKIEWICZ, Agnieszka: *Ideal Beacon Placement – Estimote Community Portal*. `https://community.estimote.com/hc/en-us/articles/202041266-Ideal-Beacon-Placement`, 2014. – [Online; accessed 06-May-2014]

[37] LUNN, John: *How does PayPal Beacon work | PayPal | Developer - Blog*. `https://devblog.paypal.com/how-does-paypal-beacon-work/`, 2013. – [Online; accessed 17-April-2014]

[38] *Bluetooth | Android Developers*. http://developer.android.com/guide/topics/connectivity/bluetooth.html#ConnectingDevices, . – [Online; accessed 17-April-2014]

[39] THOMPSON, Doug: *Apple iOS 7.1 Launches Major iBeacon Improvement | BEEKn*. http://beekn.net/2014/03/apple-ios-7-1-launches-major-ibeacon-improvement/, 2014. – [Online; accessed 29-April-2014]

[40] *Home automation - Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Home_automation, . – [Online; accessed 12-April-2014]

[41] ROUSE, Margaret: *What is Internet of Things (IoT)? - Definition from WhatIs.com*. http://whatis.techtarget.com/definition/Internet-of-Things, 2013. – [Online; accessed 29-April-2014]

[42] LEE, Adriana: *Bluetooth Wants To Talk Directly To The Internet—And Soon It Will – ReadWrite*. http://readwrite.com/2014/04/09/bluetooth-internet-ditch-phone#feed=/tag/internet-of-things&awesm=~oDZvBLU5lW9rLL, 2014. – [Online; accessed 29-April-2014]

[43] *Bluetooth Low Energy Technology | Bluetooth Technology Website*. http://www.bluetooth.com/Pages/low-energy-tech-info.aspx, . – [Online; accessed 10-May-2014]

[44] *More Than 30 Billion Devices Will Wirelessly Connect to the Internet of Everything in 2020*. https://www.abiresearch.com/press/more-than-30-billion-devices-will-wirelessly-conne, 2013. – [Online; accessed 09-May-2014]

[45] *Trilateration - Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Trilateration, . – [Online; accessed 10-May-2014]

[46] GEIGER, Philip ; SCHICKLER, Marc ; PRYSS, Rüdiger ; SCHOBEL, Johannes ; REICHERT, Manfred: Location-based Mobile Augmented Reality Applications: Challenges, Examples, Lessons Learned. In: *10th Int'l Conference on Web Information Systems and Technologies (WEBIST 2014), Special Session on Business Apps*, 2014, 383–394

[47] *Geocaching > Geocaching 101*. http://www.geocaching.com/guide/, . – [Online; accessed 10-May-2014]

[48] SCHOBEL, Johannes ; SCHICKLER, Marc ; PRYSS, Rüdiger ; NIENHAUS, Hans ; REICHERT, Manfred: Using Vital Sensors in Mobile Healthcare Business Applications: Challenges, Examples, Lessons Learned. In: *9th Int'l Conference on Web Information Systems and Technologies (WEBIST 2013), Special Session on Business Apps*, 2013, 509–518

Name: Christian Alexander Goosen   Matrikelnummer: 725092

Erklärung

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den ....

Christian Alexander Goosen