



A Dispatchable Bluetooth Indoor Location for Mobile Phones Calling for Emergency Services

Author: Alberto González Trastoy

Project Director: Carol Davids

Date: July 2016

Abstract

When a user calls the 9-1-1 emergency number he will be linked to the nearest emergency dispatch center. Once the connection is made, the operator in the center must ask for the location of the user and, although it is possible to get latitude and longitude from a cellphone, the operator will need a more precise location if the caller is inside the building.

In this context, in this project we analyzed and developed a proof of concept for improving the 911 emergency services location accuracy. In particular, we have gone through a Bluetooth based indoor location system -with a focus on the application development- capable of improving the 911 current system by offering to the emergency operator floor and room number as well as the street address of a caller. Additionally, the proposed system has been tested in different environments and conditions in order to validate our predictions and generate data sets for further research.

Resumen

Cuando un usuario llama al número de emergencias 9-1-1 esta llamada será direccionada al centro de emergencias más próximo. Una vez la llamada ha sido establecida, el operador debe preguntar al usuario cuál es su localización. A pesar de que es posible obtener una estimación aproximada de la latitud y longitud en la que se encuentra, esta información no es suficiente y se requiere una descripción más detallada de la localización del usuario si se encuentra dentro de un edificio.

Mediante este proyecto se ha investigado y analizado una posible solución para mejorar la precisión de la localización de una llamada al número de emergencias 911. En este documento se describe el sistema en general, haciendo especial hincapié en la aplicación móvil, capaz de mejorar el presente sistema que emergencias utiliza para estimar la posición de un usuario. Mediante este sistema el operador podrá recibir, de manera visual, información adicional sobre la localización del usuario como el edificio en el que se encuentra, el piso e incluso la habitación. Además, hemos analizado el sistema propuesto en diferentes condiciones y entornos para validar nuestras predicciones y generar conjuntos de datos para su investigación.

Acknowledgements

I would like to thank my project director at Illinois Institute of Technology, Carol Davids, for his support and guidance in this work. A special thanks goes to our Lab Mentor at IIT Real Time Communications Lab, Joe Cusimano for his advice and support doing our test with the Emergency Service Network of our Lab.

My gratitude, too, to the other members of the Real Time Communication IIT Lab that helped to design, develop or implement this system: Nthenya Matheka, Vipul Kumar Kothifoda, Clement Causse, Bharat Ramaswamy Nandakumar and Neilabh Okhandiar. Specially Bharat who worked both in the server side and put his effort to drastically reducing the iBeacons battery consumption, and Neil who developed a powerful management system for monitoring all the iBeacons.

In addition, a thank you to Cary Davids who helped us manufacturing all the iBeacon devices we needed and to all the Illinois Institute of Technology professors that gave me the tools to build this application and many more to come.

Lastly, I would like to acknowledge with gratitude the constant encouragement of my family and friends, which made my stay and studies in Chicago more enjoyable, and to Liyao who supported me and kept me going.

Index

1. Introduction	7
Technologies	8
Goals	10
Problems and Innovative Solutions	11
2. Scope and Requirements	12
2.1 Scope	12
2.2 Requirements	14
2.2.1 Hardware	15
2.2.2 Software	16
3. Architecture	19
3.1 IBeacon Module.....	20
3.2 Android Cellphone Module	20
3.2 Location Server Module	22
3.3 NG911 Network Module	26
4. Android Application.....	27
4.1 Libraries and Open Source Code	27
4.2 Design and Structure.....	30
5. Bluetooth Indoor Location Algorithms	35
5.1 Maximum Power Algorithm	35
5.2 Maximum Average Power Algorithm	36
6. Proof of Concept.....	38
6.1 Deployment.....	38
6.2 Data Set Generation	40
6.3 Test Results	44
7. Related Work	49
7.1. Commercial Solutions	49
7.2 Research.....	50
8. Conclusion and Future Development.....	51
Acronyms.....	52
Bibliography.....	53

Appendix A:	55
A1. AltBeacon Library	55
Appendix B:	56
B1. Nexus 5 Specifications	56
B2. Photon Particle Specifications	57
B3. Serial Bluetooth 4.0 BLE Module – iBeacon	59
Appendix C:	60
C1. Access to Location Server	60
C2. Specifications of the Location Server	60
Appendix D:	64
D1. NG911BL App Configuration	64
D2. ESInet Configuration	64
Appendix E:	67
E1. iBeacon Deployment Locations Map	67
Appendix F:	69
F1. SQL Scripts	69

Figures

FIGURE 1. STATE DIAGRAM OF THE BLUETOOTH LINK LAYER [3].	8
FIGURE 2. PROJECT PLAN NG911BL	13
FIGURE 3. EMERGENCY INDOOR LOCATION SYSTEM DIAGRAM.	14
FIGURE 4. PHOTON PARTICLE VIEW [5]	15
FIGURE 5. SPARKFUN BATTERY SHIELD	15
FIGURE 6. SIP CALL TAKER SOFTWARE.	18
FIGURE 7. INDOOR LOCATION SERVICE DIAGRAM.	19
FIGURE 8. MODULAR ARCHITECTURE	19
FIGURE 9. IBEACON MODULE	20
FIGURE 10. IBEACON MODULE – ANDROID CELLPHONE MODULE.	21
FIGURE 11. BLUETOOTH MODULE DIAGRAM	21
FIGURE 12. LOCATION SERVER MODULE.	22
FIGURE 13. ENTITY RELATIONSHIP DIAGRAM OF THE NG911BL DATABASE	23
FIGURE 14. LOCATION SERVER DATABASE. TABLE JOIN OF IBEACONS AND ITS LOCATION INFORMATION	24
FIGURE 15. PIDF-LO XML FORMATTED INDOOR LOCATION	25
FIGURE 16. NG911 ESINET MODULES	26
FIGURE 17. SIP CALL FLOW PSAP-CALL TAKER	26
FIGURE 18. SCREENSHOTS OF THE NG911BL APP. MAIN ACTIVITY AT THE LEFT AND SIPDROID CALL AT THE RIGHT.	30
FIGURE 19. SCREENSHOTS OF THE NG911BL APP. INFORMATION ACTIVITY AT THE LEFT AND THE SETTINGS ACTIVITY AT THE RIGHT.	31
FIGURE 20. NG911BL FLOW DIAGRAM	34
FIGURE 21. STUART BUILDING 1ST FLOOR WITH THE IBEACONS DISTRIBUTION	38
FIGURE 22. DEPLOYMENT AND TESTING DISTRIBUTION EXAMPLE	41

FIGURE 23. NG911BL TESTS TABLE EXAMPLE.....	41
FIGURE 24. NG911BL TESTING RESULTS DATA SET EXAMPLE	41
FIGURE 25. NG911BL TESTING APP. APPLICATION FOR DATA SET GENERATION.....	42
FIGURE 26.DEPLOYMENT AND DATA SET GENERATION FLOW CHART	43
FIGURE 27. STUART BUILDING SECOND FLOOR. TEST A.....	45
FIGURE 28.RECEIVED IBEACON IDENTIFIER-POWER AT THE LOCATION NUMBER 40	45
FIGURE 29.STUART BUILDING IBEACONS DEPLOYMENT LOCATIONS BASEMENT	67
FIGURE 30.STUART BUILDING IBEACONS DEPLOYMENT LOCATIONS FIRST FLOOR	67
FIGURE 31.STUART BUILDING IBEACONS DEPLOYMENT LOCATIONS SECOND FLOOR.....	68

Tables

TABLE 1.SIP REQUEST METHODS [4]	9
TABLE 2. LOCATION NUMBER, DEVICE, BUILDING HUMAN READABLE LOCATION RELATIONSHIP	36
TABLE 3. EXAMPLE OF THE SEEN IBEACONS SEEN DURING A TEST AT THE LOCATION NUMBER 21	36
TABLE 4.FIRST STAGE OF THE AVERAGE ALGORITHM TABLE, GROUPING BY LOCATION AND MINOR ID.....	37
TABLE 5. FINAL TABLE OF THE MAXIMUM AVERAGE ALGORITHM TABLE, GROUPING BY LOCATION AND MINOR ID	37
TABLE 6. TEST A INDOOR LOCATION RESULTS BY LOCATION. OBTAINED RESULTS (LEFT) AND IDEAL RESULTS (RIGHT)	46
TABLE 7.TEST B INDOOR LOCATION RESULTS BY LOCATION. OBTAINED RESULTS (LEFT) AND IDEAL RESULTS (RIGHT)	47
TABLE 8. NEXTNav, POLARIS AND QUALCOMM HORIZONTAL ERROR STATISTICS [12].....	49
TABLE 9.SIP RESPONSE CODES.....	64

1. Introduction

In 2013 about 147 [1] million calls were made to 9-1-1 in the U.S. in a year and, it is estimated that 240 million 9-1-1 calls were made in 2015. Of these, 70% of the calls to 911 are placed from cellphones. Most of this phones are capable of sending text, pictures, video or other kinds of information.

One very important piece of this information that is required by the 9-1-1 system is the location of the calling party -this location is used for two purpose: First, to route the call to the appropriate call taker (responder to the caller); Second, to allow the dispatcher to send help to the right place. Typically, Commercial Mobile Radio Service (CMRS) providers first attempt to locate the caller using A-GPS [2]. However, even with favorable conditions if the call is made from a mobile device inside a building, GPS signals may be blocked compromising the accuracy and reliability of GPS technology¹. Even if the emergency call sends its accurate location the first responder will only obtain the latitude and longitude and/or the street address but not the floor and nearby room. Without this information the first responder won't easily find the caller.

In 2015, the Federal Communications Commission (FCC) and the four largest cellphone carriers worked on a new set of rules to help 911 responders better locate wireless callers. In the final document, they agreed on increasing the percentage of cellphone calls to 911 that transmit location data. Using new technologies, including Wi-Fi and Bluetooth, they are beginning an effort to increase the availability and the accuracy of location data from cellphone calls [3]. Additional location data such as the "Z" or altitude of the caller would definitively reduce emergency response time.

In this context, with this project we have designed, analyzed and developed a proof of concept of a Bluetooth indoor location system, NG911BL, capable of increasing indoor location accuracy. The system consists of 4 modules: First, the Bluetooth transmitters; Second, the mobile phone application that starts the emergency call and forwards the call, included location information, to the call dispatcher; Third, a location server module that serves as the backend of the application and applies the algorithm for estimating the location of the Bluetooth transmitter -Bluetooth iBeacon device- closest to the caller and returning this location; And in the fourth module we have the Emergency Services network to route the emergency call to the Public Safety Answering Point (PSAP). This last module is the NG 9-1-1 Network Module.

¹ Some CMRS providers report an A-GPS failure rate of approximately 25 percent [2]

Technologies

The Bluetooth Next Generation 911 system will use the following technologies:

Bluetooth LE

Bluetooth LE works in the bands of 2.4 GHz where accommodates 40 channels with 2 MHz channel spacing. It has an outdoor range of around 30 meters and uses GFSK, $\pi/4$ DQOSK or 8DPSK (3Mbps) modulations. Although it works at the same bands than Wifi, the interference between Wifi and Bluetooth is rare due to the Bluetooth adaptive channel hopping feature.

The operation of the Link Layer can be described in terms of a state machine with the following five states [3]: Standby State; Advertising State; Scanning State; Initiating State; Connection State.

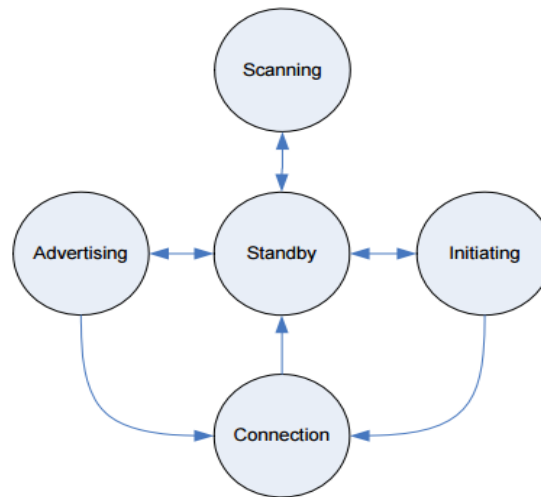


Figure 1. State Diagram of the Bluetooth link layer [3].

In this project, as we want to detect and capture the Bluetooth transmitter identifiers, we will be capturing sensors when they are in advertising state. On the other hand, the cellphone will be executing an active scan and capturing advertising state devices when its Bluetooth module is in scanning state.

Why Bluetooth LE?

We decided to use a Bluetooth LE system because: We didn't need a high bit rate for transferring data. The transmissions need to send very little information and; It has low power consumption. The device will be always on, therefore, consumption is important.

iBeacon

Introduced by Apple in 2013, this protocol has been growing in usage during the recent years. The iBeacons are a class of Bluetooth low energy (LE) devices that broadcast their identifier to the electronic devices within their range. With this

technology smartphones or tablets are able to perform different operations when they are near an iBeacon.

The iBeacons use Bluetooth LE proximity sensing to transmit a universally unique identifier (UUID) picked up by a compatible application or operating system. Amongst other uses, the identifier and several bytes sent with it can be used to determine the device's physical location based on the RSSI value.

We used iBeacons because: a) Are cheap devices; b) Its default configuration offers us a data package with an identifier and the signal strength; c) Are wireless, with a relatively small coverage range; d) It is a scalable system. iBeacons are easy to install and deploy; e) They have a low battery consumption

VoIP

Through Voice over IP technology we will be able to deliver multimedia through IP networks. For this real time multimedia communications, we will need a protocol, the Session Initiation Protocol (SIP), which will signal and control multimedia communications. For performing this control SIP uses methods:

Request Method	Description
ACK	Confirms that the client has received a final response to an INVITE request
BYE	Terminates a call and can be sent by either the caller or the callee
CANCEL	Cancels any pending request
INFO	Sends mid-session information that does not modify the session state
INVITE	Desires to initiate a call session with other user client(s) Asks a server to establish a session
MESSAGE	Transports instant message using SIP
NOTIFY	Notify the subscriber of a new Event
OPTIONS	Queries the capabilities of servers
PRACK	PROvisional ACKnowledgement Indicates the callee to reserve network resource
PUBLISH	Publishes an event to the server
REGISTER	Registers the address listed in the To header field with a SIP server
REFER	Asks recipient to issue SIP request used for call transfer
SUBSCRIBE	Subscribes for an Event of Notification from the Notifier
UPDATE	Modifies the state of a session without changing the state of the dialog Informs the callee of finishing network resource reservation

Table 1.SIP Request Methods [4]

We used SIP signaling to place the call because it is the standard of the Next Generation 911 Services specified by National Emergency Number Association (NENA) [5]. The body of the SIP INVITE will contain the location information with the additional indoor location information and is displayed on the NG911 PSAP screen.

Goals

The goal of this project is to develop and deploy a proof of concept system based on Bluetooth that provides the Indoor Location of an emergency caller calling from a smartphone. This report will focus on the first 3 modules: Bluetooth iBeacons; Android Application; Location Server. The fourth module is the NG911 Network module, which includes the PSAP where the call will be answered and the location of the caller will be displayed. The system as developed gives greater location accuracy than the FCC currently requires.

The first module, the iBeacon, will be configured to be in advertising state and to use minimum battery. In the second module, the goal will be the development of a cellphone application to capture Bluetooth iBeacons information and to provide the indoor location of the caller using information from these Bluetooth sensors.

In order to achieve it, we will require a third module where an indoor location algorithm will be implemented. This algorithm will use the information from Bluetooth sensors and other sources to identify the location of the caller, which will serve as a source for the future Next Generation 911 implementation.

Through this project, we are proposing a new 9-1-1 indoor location feature based on the usage of Bluetooth sensors iBeacons, for providing indoor location and, at the same time, we will take advantage of the VoIP technologies for transmitting this location of the caller to the 911 dispatcher. Also, it is important to point out that this system has been developed to give to the user protection of his location information. It will be the cellphone which provides the indoor location and the iBeacons will not keep any information about the location of the caller.

After having the complete system, the validity of the entire application will be tested in real world environments and, additionally, this Bluetooth or IEEE 802.15.1 indoor location will be compared with previously developed Wifi or IEEE 802.11 indoor location systems.

Problems and Innovative Solutions

The main question we want to answer in this project is: how can use Bluetooth to deliver location information more accurately without changing the emergency services infrastructure?

Nowadays, indoor location accuracy provided by cellphone operators doesn't satisfy the FCC future requirements. Requirements with the purpose to improve the current 911 emergency services technology². In this project we will propose and test a solution that can satisfy those requirements and improve the 911 technology.

The solution uses information obtained from iBeacons and sends this data to the location server which uses that information to calculate the location of the caller. Then, this location information is then sent to the phone application which uses the SIP INVITE to carry the location information and the PSAP.

Under this problem we faced several sub problems that appeared during the project development. First, we will need to identify each deployed iBeacon. An identification that will add complexity. To solve this problem, we purpose the usage of the build-in Bluetooth parameters, UUID, major and minor, for uniquely identifying each received Bluetooth signal from an iBeacon.

In addition to the deployment distribution problem, we also had to develop a solution capable of managing the distributed iBeacons. We needed a way to deploy, control and update all the devices in a location easily. This was solved through the proposed flow process described in the point 6 of this document.

A critical problem faced during the project development is the Bluetooth battery drain. Initially, with the preconfigured settings and first design version sensors, the iBeacon sensor consumption was so high that we had to charge it every 3 days. This was mainly caused because we used an iBeacon management system which required Wifi connection and produced periodic logs sent to a cloud server. We solved the high consumption rate doing two things. First, using lithium-ion batteries with higher capacity. And secondly, turning the microprocessor of the iBeacon into sleep mode and changing the update frequency of the management system to once a day.

Another problem that needs to be studied is the optimal amount or density of iBeacons per floor and room. While increasing the amount of iBeacons per floor would increase accuracy, it would also add complexity and congestion to the Android device and the Location Server.

² The current system is based on an ALI database with the location of their users obtained using radiolocation or GPS systems [10].

2. Scope and Requirements

2.1 Scope

The FCC requires that all CRMS providers should be able to provide 50-meter accuracy, 50 percent of the time in two years for both indoor and outdoor emergency location calls. In a near term period, 5 years, they require CRMS to achieve an 80 percent reliability when locating calls [6]. Also, in the FCC standards 30 seconds after the call is sent is defined as an acceptable time limit for obtaining location information [7].

In particular, in this project, in our scope we will designing a system for smartphones with 4.0 Bluetooth technology which will provide information that can be used to determine indoor location of the caller.

We want to reach a degree of accuracy that allow us to locate people reaching room level identification in order to save people. The distance between real location and user location is not the priority. Instead, we aim to determine the user location at the right floor and area on that floor.

We are making a IP based SIP phone call rather than using a voice channel on the cellular infrastructure.

Also, we will design the hardware iBeacon devices and develop the Android Application with which we will get indoor location data and interact with a currently available ESI-net. We decided to use the available ESI-net infrastructure that was available in the IIT Real-Time Communications Lab. This infrastructure conforms the NG 911 NENA standards.

The current work is based on using the received power for obtaining the indoor location. The current work does not use nearest neighbor, fingerprinting or other available technics.

Finally, we will develop and test multiple algorithms. Our goal will be to collect and generate a set of data and test those algorithms over the same set of data.

Project Plan

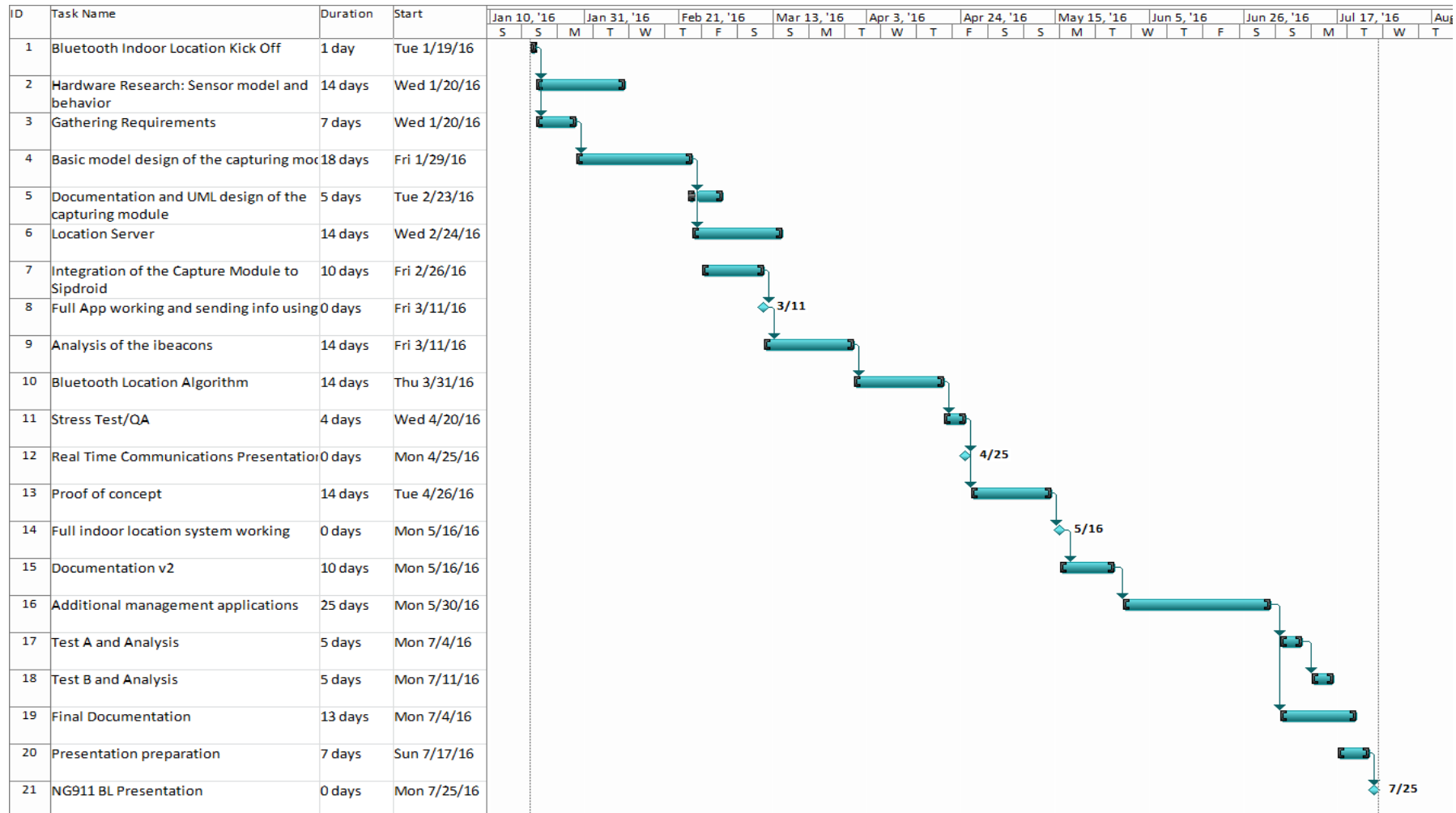


Figure 2. Project Plan NG911BL

Regarding our project plan we didn't have major delays and we were able to have a complete working prototype working. We were able to integrate and develop all the elements of the system shown in the Figure 3.

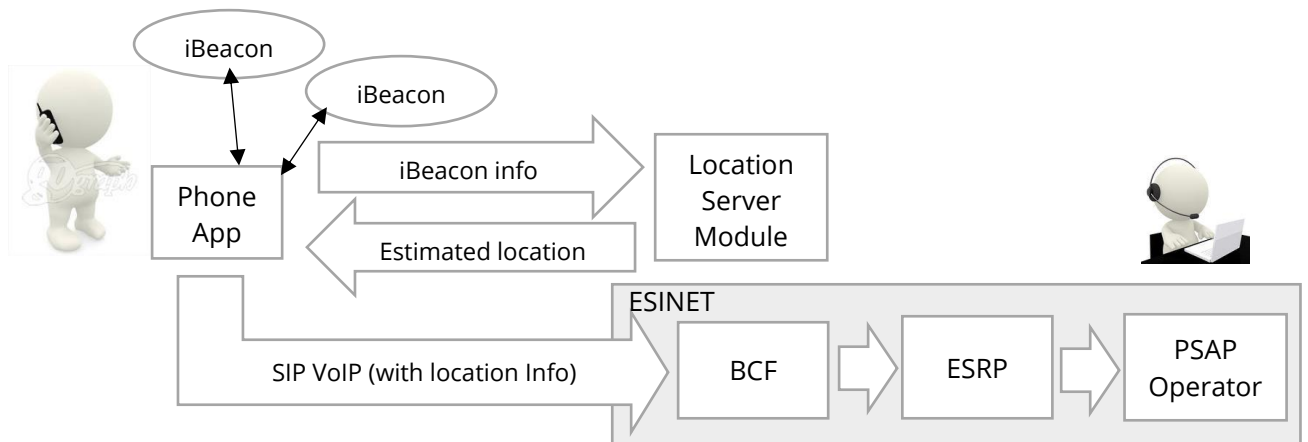


Figure 3. Emergency Indoor Location System Diagram

As it is shown in the Figure 3, the iBeacons are continuously in Bluetooth LE advertising mode and the cellphone will capture their identifiers and send them to the server to process and retrieve the indoor location of the caller. Finally, the cellphone will create a SIP INVITE with this information and it will be sent to the call dispatcher.

2.2 Requirements

For the research, design and development of the NG 9-1-1 Bluetooth Indoor Location for Emergency Services we had the following high level requirements:

- (1) The working prototype should be finished by July.
- (2) The Application has to be compatible with a cellphone currently available in the market. It has to be compatible with at least one of the major mobile Operating systems. And it has to be a smartphone.
- (3) Call and information has to be sent following NENA i3 architecture standards for NG 9-1-1.
- (4) The NG911BL system has to be able to establish an emergency call in less than 30 seconds.
- (5) The prototype has to provide indoor location within 50-meter accuracy more than the 80 percent of the times.

For the development of our purposed indoor location and the completion of the above high level requirements of the system we will require:

2.2.1 Hardware

Below are defined the hardware requirements of each of the devices used in the system.

iBeacon transmitter

This will be a build device which will send the Bluetooth LE signals (iBeacons) to the cellphone. This build device is composed by the following parts:

- Photon Particle ³
It is a prototype-to-production platform for developing an Internet of Things product.

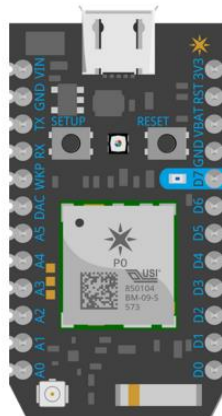


Figure 4. Photon Particle view [5]

- Sparkfun Battery Shield
It is basically an adapter for using a LiPo battery with the Photon Particle Hardware.

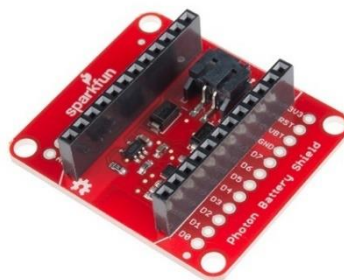


Figure 5. Sparkfun Battery Shield

- Polymer Lithium Ion Battery of 2500 mAh

³ For the specifications of the Photon Particle device go to the Appendix B2

- Serial Bluetooth 4.0 Module⁴

It is the Bluetooth antenna which will send the iBeacon identifier.

Cellphone device

This element will be a smartphone with:

- Android OS SDK 21 -LOLLIPOP- or higher version.
- 3G/4G and/or Wifi compatibility.
- Bluetooth 4.0 Low Energy compatibility.
- An Android NG911 application. This cellphone will have to have installed the app for receiving the beacons from the iBeacon transmitter and then sending this information, translated into an indoor location, to the 911 dispatcher.

The cellphone used in this project is the Google Nexus 5⁵.

Location server

This server will be applying the algorithm which will return the estimated indoor location to the cellphone.

The used server works with 2.6.32 x86_64 GNU/Linux, CPU uses 3 Intel Core 2 Quad Q9400 processors and has 100GB of available memory⁶.

Also, the location server will contain the database with all the indoor location data.

NG911 Network Module. ESInet

This is a set of hardware modules that form a Network module to route the emergency call to the PSAP. We will use the NG9-1-1 Test Bed at the IIT Real-Time Communications Lab. This test bed, developed before the start of this project, works as a real PSAP and, therefore, we can be sure that the NG911 Indoor Location solution is compatible with all the ESInets that meet NG9-1-1 NENA i3 standards.

2.2.2 Software

The software requirements for each one of the devices will be:

iBeacon transmitter

Will use a flexible free OS in the device with which we can change beacons frequency battery and many other Bluetooth oriented features.

Should be low battery oriented. It should use minimal battery drain consumption software.

Also, we evaluated an easy-to-use and simple software positively.

⁴ Specifications of the Bluetooth Module in the Appendix B3

⁵ Specifications of the Nexus 5 can be found in the Appendix B1

⁶ More Specifications of the Location Server can be found in the Appendix C2

In the project we designed and developed iBeacons with FreeRTOS, a Real Time Operating System kernel for embedded devices.

Cellphone device

As this is a proof of concept we will use the most popular cellphone OS, Android. The Android OS version used will be SDK 21 (LOLLIPOP) or higher. The version used for this project is SDK 21.

The cellphone will have an installed Android App which will be:

- Flexible with user and VoIP configuration options.
- Capable of managing received Bluetooth beacons.
- Fast and capable of sending parsed captured information to the location server.
- Using standard HTTP transmissions.
- Modular. It will be easy to monitor and debug the application.

Location server

This server working with:

- PHP and Node.js which will execute the location algorithm. Though this scripting languages we will communicate with the web server and the more probable indoor location will be retrieved from the database and returned as an XML file to the cellphone.
- A MySQL database which will contain information about identifiers and its location.

Why Node.js?

Regarding the chosen language, we decided to use Node.js due to its asynchronous nature. It can handle far more number of concurrent requests as compared to Apache. Although we might have problems finding modules/libraries which substitute some of Apache functionalities, as we don't need too much functionalities in this case, Node.js is a most efficient way to build a fastest server.

GEOPRIV Presence Information Data Format Location Object (PIDF-LO)

In the server we will parse the devices information gathered by the cellphone and obtain the indoor location of the user. For the representation of this indoor location we will use the standard for representing location information, RFC 5491.

NG911 Network Module. ESInet

The NG911 Network module will guide the SIP package to the PSAP. First, the SIP package reaches the public IP of the SBC which acts as a proxy firewall. After that, the SBC will conduct the VoIP signals. This signals will contain the SIP package username and password which will be authenticated against an ASC device and, if the credentials are valid, the package will be redirected to the ESInet through the ESRP. The ESRP will route the call to the PSAP, at the RTC Lab, and the Sip Call Taker software will parse the SIP message.

Then, as shown in the Figure 6, the call dispatcher will be able to see the contact calling and its location.

The screenshot displays the 'sipcalltaker' application window. The main interface is divided into several sections: 'Caller Information', 'Emergency Information', 'Telephone Controls', and 'Links'. The 'Caller Information' section shows details for a caller from '10 W35th St, 9th Floor, Used iBeacon Location system solution'. A 'Change Emergency Location' dialog box is open, allowing the user to update the caller's location. The dialog box contains fields for Geographic grid (Latitude and Longitude), Country (us), State (IL), County (chicago), City, Borough, Neighborhood, Street (35th), Leading-direction (W), Trailing-direction, Street-suffix (St), House-number (10), House-suffix, Landmark, Additional (Used iBeacon Location), Floor (9), and Name. The 'Telephone Controls' section includes buttons for Answer, Dial, Call Back, Hold, Conf, Transfer, Release, Mute, Full, and Partial. The 'Links' section shows a 'Call Taker Status' of 'busy' and a 'registered' status.

Caller Information	
Caller Location	10 W35th St, 9th Floor, Used iBeacon Location system solution
Community	
Contact Info	<sip:android2@64.131.109.30>
Alternate	
Name	android2@64.131.109.30
Service Provider	
Secondary Contact Info	
Latitude	

Emergency Information	
Emergency Location	10 W35th St, 9th Floor, Used iBeacon Location system solution
Community	
Latitude	
Longitude	
Emergency Type	
Emergency Secondary Type	
Name	android2@64.131.109.30
Notes	

Telephone Controls	
Answer	Dial
Hold	Conf
Release	Mute
Call Back	Transfer
Full	Partial

Links	
Call Taker Status	busy
registered	

Figure 6. SIP Call Taker Software

In this example, the user was located at 10 W 35th Street, 9th Floor, 9E3-1 Lab.

3. Architecture

All the required elements intercommunicate following different architecture types. In the Figure 7 we can see how the iBeacons are advertising and the cellphone scans the environment (1). Then, the cellphone sends the scanned data to a set of servers (2) building a Client/Server network architecture. After that, using the received data, the server applies the location algorithm, queries the MySQL database (3) (4) and the indoor location information is sent to the cellphone (5) which sends a SIP INVITE with the location information to the call dispatcher (6).

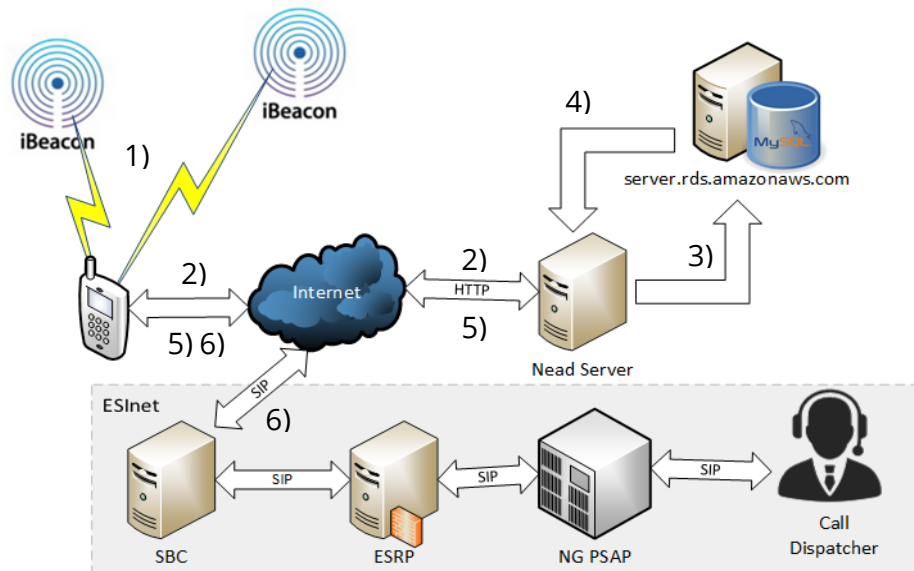


Figure 7. Indoor Location Service diagram

The system architecture will have 4 modules: the iBeacon module, the Android Application module, the Location Server module and the NG911 Network module. We can see a basic diagram with the different modules in the Figure 8.

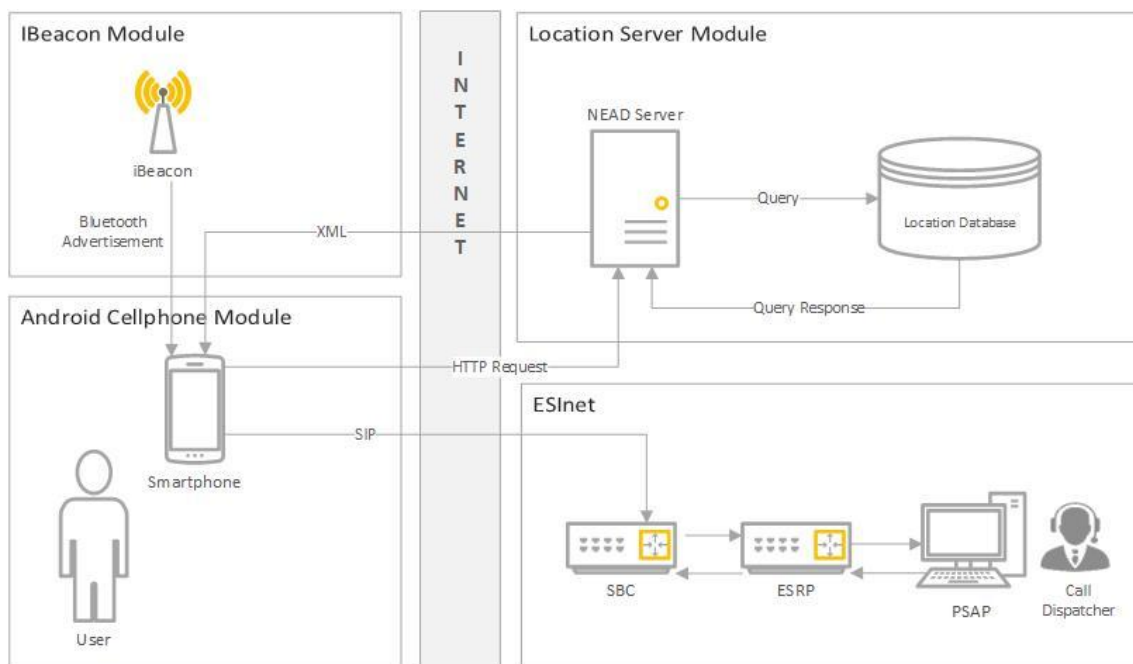


Figure 8. Modular Architecture

As we can see in Figure 8, each module is formed by several submodules or devices. The general system flow is as follows:

1. The iBeacons advertises its identifier and application captures this identifier
2. The identifier and received signal strength are send as HTTP Request to the location server module.
3. The Location server parses the data and executes an algorithm for estimation of the user indoor location based on signal strength.
4. The server queries the location database and retrieves the location of the iBeacon which is estimated to be the nearest to the user.
5. The server returns an XML file containing the estimated indoor location of the user.
6. The application attaches this XML file to the other XML information required for a successful SIP INVITE message.
7. The SIP INVITE is received by the SIP Call Taker application in the PSAP. After accepting the call though the application the call dispatcher will visualize the indoor location of the caller.

3.1 IBeacon Module

This first module of the indoor location emergency system is the iBeacon. The iBeacon will be in Bluetooth LE advertising state sending its unique identifier to all the devices in range. It contains the 4 main submodules shown in the Figure 9: The iBeacon Management application; The Bluetooth module; The Wifi module; And the small and simple FreeRTOS OS for embedded systems.

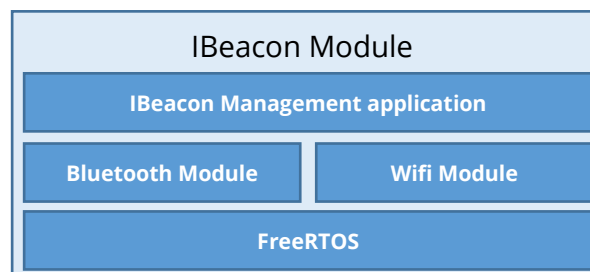


Figure 9. IBeacon Module

The iBeacon management application has been specifically designed to manage and control the devices status while optimizing battery consumption. This application has reduced the Bluetooth module advertising time period and stablished a sleep period of 24 hours to all the iBeacon module with the exception of the Bluetooth module; this one will be sending advertising signals every 500ms.

3.2 Android Cellphone Module

This second module of the indoor location emergency system is formed by three submodules: The NG911BL Application; The Bluetooth Module; And the Android OS. In the Android Cellphone Module, once the user dials 911 or calls through the NG911BL Android App, the device will start scanning those Bluetooth advertising signals from all the iBeacons within the range.

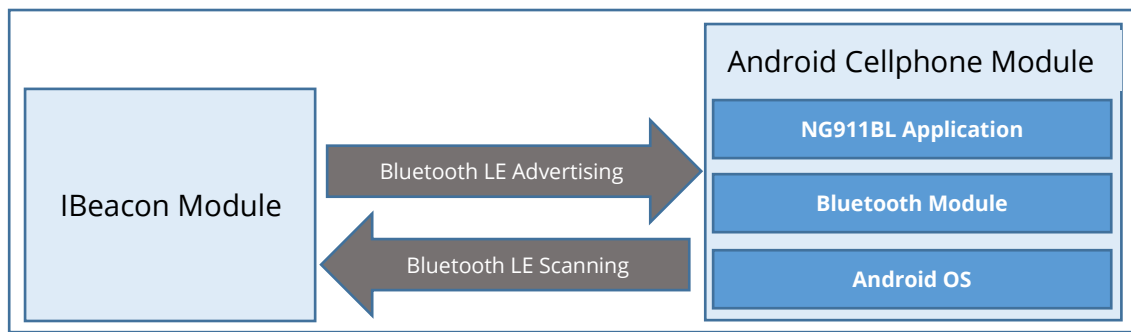


Figure 10. iBeacon Module – Android Cellphone Module

As we can see in the Figure 10, the Bluetooth submodule of the cellphone module will be capturing the Bluetooth LE advertisements. In order to perform this operation, the NG911BL App will be sending commands to set the Bluetooth submodule to scanning state. Then, the application will capture Bluetooth Advertised packages and will parse its RSSIs and iBeacon identifiers which will be sent in JSON format to the location server module using HTTP. This identifier present in each iBeacon is formed by two identifiers called major and minor, and those values will be used to differentiate devices.

Bluetooth Module

This submodule consists of a host and a controller. The host will send commands to the controller. Through what it is called HCI (Host Controller Interface), the NG911BL application will manage and execute commands form the host, and the controller will execute the corresponding functions (E.g.: Set Bluetooth Module to scanning state) and will return the subsequent event.

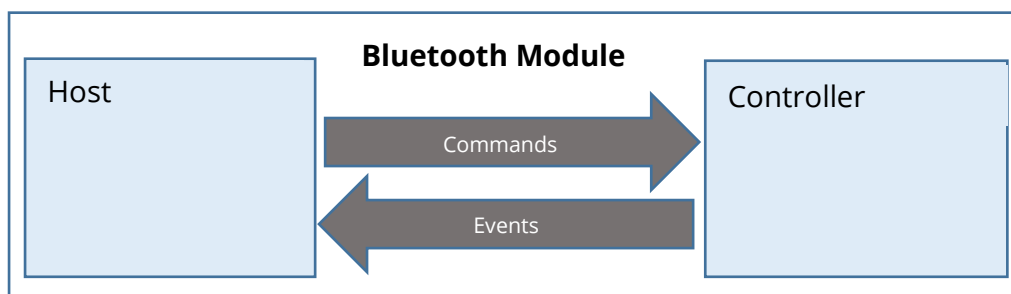


Figure 11. Bluetooth Module Diagram

The hardware elements of this module are: Android mobile phone; Particle Photon microcontroller; Serial Bluetooth 4.0 Module; Sparkfun battery Shield; Li-On battery.

It should be noted that in the system, the developed Android Application plays a very important role. First, will act as a Bluetooth scanning tool, then as a client of the location server, and later on as a SIP client.

3.2 Location Server Module

In this module, first, the captured information will be translated in to a known indoor location, and after that, this location will be returned to the On-Site module cellphone.

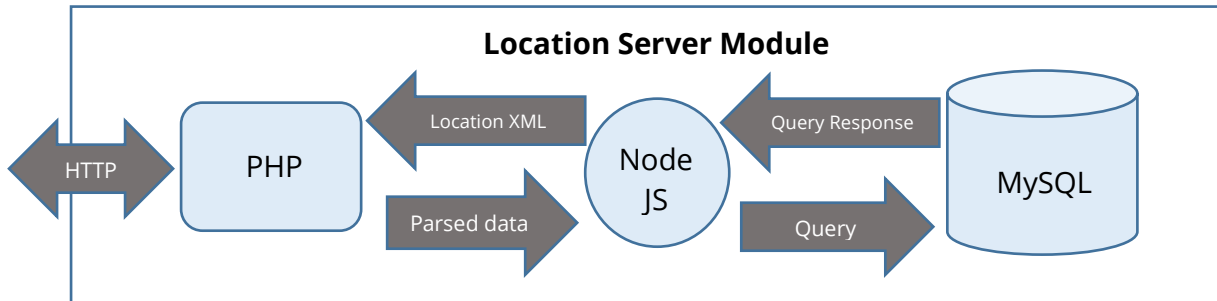


Figure 12. Location Server Module

As it is represented in the Figure 12, this module will be formed by 3 submodules: PHP script, Node JS script and a MySQL database.

The flow process is as follows:

- 1) HTTP GET of the cellphone captured data is received in the location server.
- 2) The PHP script will parse the JSON formatted data and execute the JavaScript location algorithm⁷.
- 3) The parsed data is received by the Node JS environment and, after applying the location algorithm, will query the database with the identifier of the device nearest to the cellphone.
- 4) Database returns the location which matches the introduced identifier.
- 5) The query response is embedded inside an XML file.
- 6) PHP script executes an echo of the XML file generated in the Node JS module.
- 7) HTML response is shown in the web browser and can be retrieved by the cellphone.

As mentioned above, this module is composed by a server with the location information. The server will integrate a database with the relationship between Bluetooth sensor identification and indoor location (room and building).

For example, when a new iBeacon is deployed in a new room, we will have to configure the database accordingly. We will recognize the major and minor identifiers and save them with the room location in the same database tuple.

⁷ For this first proof of concept we used a location algorithm based on defining the location of the user as the location of the iBeacon with higher received signal strength.

The Entity Relationship Diagram of the location database is as follows:

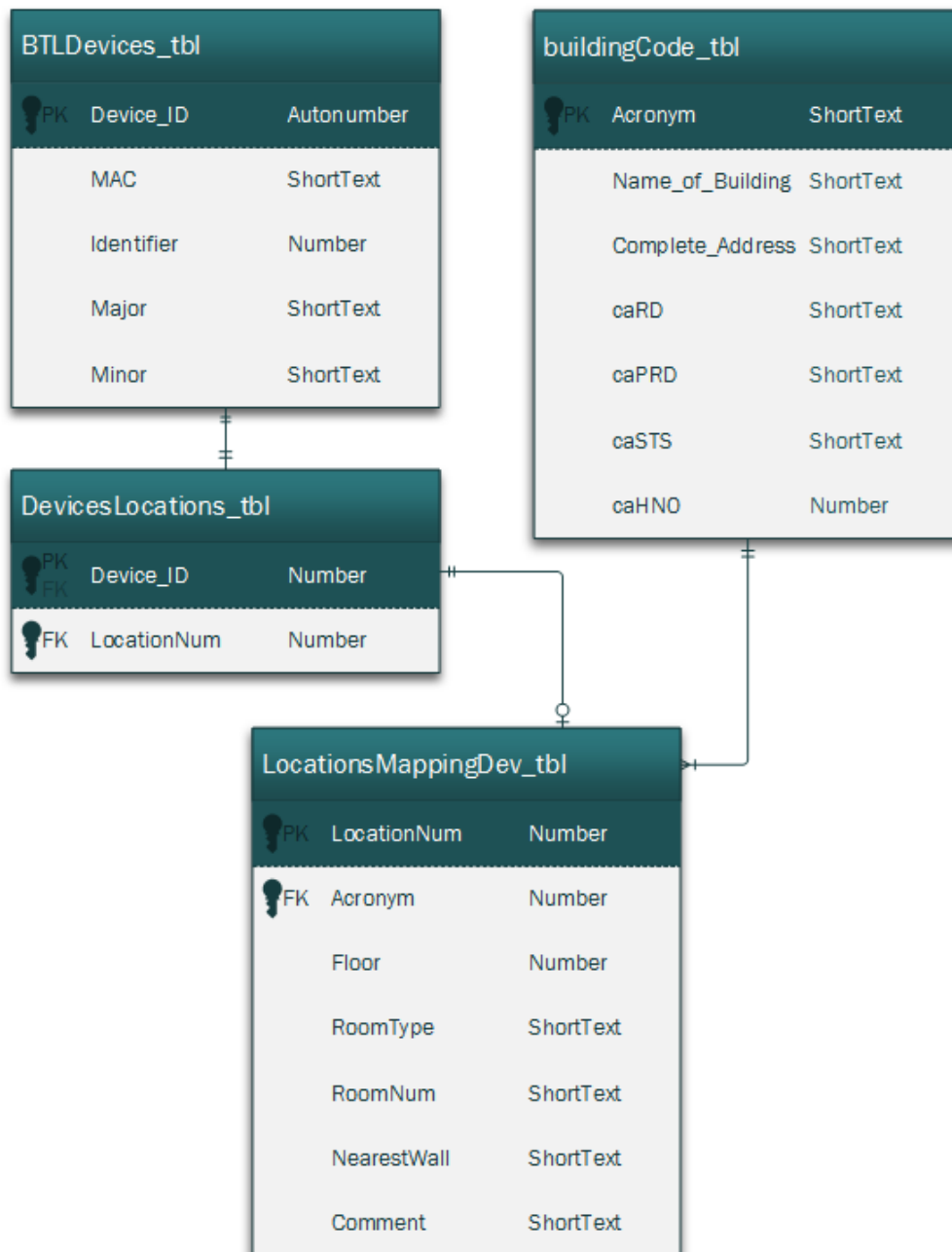


Figure 13. Entity Relationship Diagram of the NG911BL Database

As we can see the database will contain the following entities:

- BTLDevices**: This table contains **Device_ID** as the primary key, MAC address of each iBeacon, identifier, major and minor attributes.
- DevicesLocations**: This table will interconnect the **Device_ID** as primary key and foreign key of **BTLDevices**, with **LocationNum** as foreign key from **LocationsMappingDev**.
- BuildingCode**: The purpose of this table will be to provide the buildings information from Illinois Institute of Technology. It contains **Acronym** as

primary key and the Name_of_Building, Complete_Address, caRD, caPRD, caSTS and caHNO attributes which define the address of each building.

- d) LocationsMappingDev: This table is the core of the database with the deployment location information. This table will have LocationNum as primary key with Acronym as foreign key from the BuildingCode table and with Floor, RoomType, RoomNum, NearestWall and Comment as attributes.

Below we can see an example of the join between the BTLDevices table (major, minor and coreid field value identifiers) and the LocationMappingDev table with the location information:

major	minor	coreid	Acronym	Floor	RoomType	RoomNum	NearestWall	Major	Minor	LocComment
101	110	3c0043001547343433313338	SB	2	WestHallway	214	W	101	110	Ceiling
101	131	3c003c000947343337373738	SB	2	NorthHallway	218	N	101	131	Ceiling
101	109	3e002f000f47343432313031	SB	2	NorthHallway	220	N	101	109	Ceiling
101	125	23002a000247343138333038	SB	2	NorthHallway	223	N	101	125	Ceiling
101	108	3c0038001447343338333633	SB	2	NorthHallway	225	N	101	108	Ceiling
101	126	2c0044001047343432313031	SB	2	NorthHallway	226	N	101	126	Ceiling
101	124	3e002d000347343138333038	SB	2	EastHallway	230	E	101	124	Ceiling
101	106	390039000e47343432313031	SB	2	EastHallway	231	W	101	106	Ceiling
101	129	250023000d47343233323032	SB	2	EastHallway	233	E	101	129	Ceiling
101	105	420033000547343138333038	SB	2	SouthHallway	237	S	101	105	Ceiling
101	123	280036000347343233323032	SB	2	SouthHallway	239	S	101	123	Ceiling
101	112	28001d000d47343432313031	SB	2	SouthHallway	201	S	101	112	Ceiling
101	121	2b002d000c47343432313031	SB	2	SouthHallway	204	S	101	121	Ceiling
101	134	230023000d47343233323032	SB	2	SouthHallway	206	S	101	134	Ceiling
101	116	410024000647343232363230	SB	2	WestHallway	210	W	101	116	Ceiling

Figure 14. Location Server Database. Table join of iBeacons and its location information

Messages with the following order can be seen when the indoor location module is being used:

- 1) The HTTP request is received by the server and the sent URL will have the following format:

```
http://nead.bramsoft.com/index.php?json=[{"major":100,"minor":4,"rssi":-35},{"major":100,"minor":4,"rssi":-35},{"major":100,"minor":4,"rssi":-35},{"major":100,"minor":4,"rssi":-31}]
```

- 2) The PHP module parses the identifiers values of “major” and “minor” and the “rssi”, value of received power from a device in dBm.
- 3) In the Node JS module, the location algorithm is executed and, based on the “rssi”, the algorithm chooses the most probable identifier and queries the devices table.
- 4) The Node JS embeds the location in PIDF-LO XML format.


```

<?xml version="1.0" encoding="ISO-8859-1"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
  xmlns:gp="urn:ietf:params:xml:ns:pidf:geopriv10"
  xmlns:ca="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr"
  xmlns:gml="http://www.opengis.net/gml"
  entity="sip:caller@64.131.109.27">
<tuple id="id82848">
  <status>
    <gp:geopriv>
      <gp:location-info>
        <ca:civicAddress>
          <ca:country>us</ca:country>
          <ca:A1>IL</ca:A1>
          <ca:A2>Chicago</ca:A2>
          <ca:A6>35th</ca:A6>
          <ca:PRD>W</ca:PRD>
          <ca:STS>St</ca:STS>
          <ca:HNO>10</ca:HNO>
          <ca:LOC> Used iBeacon Location system solution</ca:LOC>
          <ca:FLR>8</ca:FLR>
          <ca:ROOM>8E3.3</ca:ROOM>
          <ca:PLC>Lab</ca:PLC>
        </ca:civicAddress>
      </gp:location-info>
      <gp:usage-rules/>
      <gp:method>Manual</gp:method>
    </gp:geopriv>
  </status>
  <contact priority="0.8">sip:caller@64.131.109.27</contact>
<timestamp>2016-04-07T07:01:18.059Z</timestamp>
</tuple>
</presence>

```

Figure 15. PIDF-LO XML formatted indoor location

- 5) Finally, the PHP Module echoes the XML location file to the cellphone and responds with HTTP/1.1 200 OK and the XML formatted location file.

3.3 NG911 Network Module

This module is a Network to route emergency calls to a call dispatcher, a Next Generation 9-1-1 Test Bed [8]. This Test Bed, which will allow us to place an end-to-end call to a PSAP through a ESInet, is made up of 3 main submodules: BCF, ESRP and PSAP.

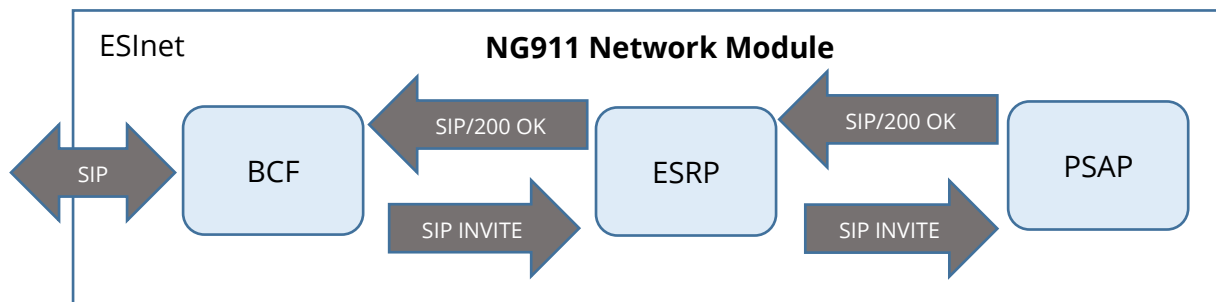


Figure 16. NG911 ESInet Modules

The BCF will authenticate the VoIP call which will be routed through the Emergency Services Routing Proxy (ESRP). Then, the ESRP, will ask to the Emergency Call Routing Function (ECRF) using a location to service translation (LoST) protocol; a protocol used for two functions: call routing and location validation.

After that, the call will be validated and the ECRF will route it to the nearest PSAP. A satisfactory VoIP connection behavior from the cellphone to the ESInet will generate the following flow of packages in the PSAP:



Figure 17. SIP call flow PSAP-Call Taker

As we can see, the PSAP receives the SIP INVITE, which will contain the indoor location as an XML file, and the Call taker 911 SIP software will start ringing and will respond with the 200 OK confirmation produced when the call dispatcher answers the call.

4. Android Application

One of the developed modules to provide the Bluetooth Indoor Location Service is the NG911BL application. This application will manage the core operations of the system and will allow us to send the indoor location to the 911 call dispatcher. The mobile application is built on the Android OS and includes the functions of a SIP User Agent as well as the Bluetooth sniffer. The cellphone will connect to the location module for getting his precise location and will perform the 911 call sending the indoor location through VoIP.

4.1 Libraries and Open Source Code

The application has been developed using the following 3 main elements: The AltBeacon API, the Android Volley API and the Sipdroid application Open Source code.

AltBeacon API

This first library will be used for sniffing Bluetooth packages in the environment. The android app will use predefined methods for communicating with the Bluetooth controller of the cell phone, detecting Bluetooth signals or beacons and retrieving the values of interest. This capturing Bluetooth signals from iBeacons operation will be done doing ranging.

What is ranging?

Ranging is an operation in which the Bluetooth controller of the cellphone will be performing continuous scans and will detect all the devices within the range. The difference with monitoring is that in this case the cellphone won't detect the exact moment in which a specific iBeacon is in or out of range. Through ranging we will get a full list of matching beacons currently in range—complete with their UUID, major, minor and RSSI values.

The drawbacks of using ranging are the high power consumption. However, as the ranging won't start until the user does the 911 call, this will be a very sporadic operation where battery usage isn't relevant.

For working with this ranging functionality of the AltBeacon library, in our activity we will implement BeaconConsumer in the calling activity and we will use a BeaconManager which will start the Scanning Service:

```
//Manager for Bluetooth beacons
beaconManager = BeaconManager.getInstanceForApplication(this);
// To detect proprietary beacons, you must add a line like below corresponding
to your beacon.
// type. Do a web search for "setBeaconLayout" to get the proper expression.
beaconManager.getBeaconParsers().add(new BeaconParser().setBeaconLayout("m:2-
3=0215,i:4-19,i:20-21,i:22-23,p:24-24,d:25-25"));
//Binds an Android Activity sipdroid to the BeaconService.
beaconManager.bind(this);
```

Where the `setBeaconLayout` functionality is defined in the appendix A1. Also, in this call activity we will use `setRangeNotifier(new CustomRangeNotifier())`. Which will specify a class that should be called each time the `BeaconService` gets ranging data. Another used method is the `didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region)` which will be overridden for working with the captured Beacons and saving the values of interest into an array.

Now that we have the unique identifiers and RSSI, how do we translate RSSIs into indoor location?

First, in our first approach we were considering the possibility of using the build in distance estimation feature in the Android AltBeacon Library, however, after testing it we couldn't estimate distance with precision.

The API provides estimates of the distance to a beacon in meters based on the signal strength (RSSI) between the device and the beacon. To do so this, library applies a filter to the RSSI based on past measurements. However, although it throws out the top 10% and bottom 10% of measurements and averages the rest [7], the estimation can have a distance estimation error higher than 500%. Also, although less important, this filtering can produce time lag in distance updates.

Due to this problem we decided to take a very different approach. We would develop an algorithm in the post execution. Once the android application has captured all the beacons in its range, this RSSIs -and its identifiers- will be sent to the server. The server will apply an algorithm which will take into account all the detected devices and the RSSI. Through this algorithm, which will be explained in the next chapter, we will have a more reliable estimation of the indoor location.

Volley API

This library will be used for the HTTP transmissions between the NG911BL App and the location server. We used this library because it simplifies the networking process and makes it faster.

Volley support of raw strings, images, and JSON will make easier for the developer to use one formatting type or another. Also, as the HTTP transmissions will be little amounts of data, Volley functionality of holding responses in memory during parsing and organizing those responses later won't be a problem but a modularity and flexibility application increase.

Sipdroid Open Source Application

Sipdroid is an open source application for making VoIP calls using the SIP protocol developed by i-p-tel GmbH.

We decided to use a SIP client for making calls to the PSAP operator because this protocol allows us to send multimedia information over IP.

Instead of using the voice networks we will use the data network for being able to send data such as the indoor location to the user. Another possible sent information could be the picture of an offender or the identity and caller.

Sipdroid modifications

- org.sipdroid.sipua.ui.Caller.java:

This class is a broadcast Receiver which will be listening the dial pad of the cellphone. If the user dials 767, our 911 number for testing purposes, the application will start and run the Bluetooth LE scanning and the SIP call protocol.

- org.sipdroid.sipua.ui.Sipdroid.java:

This class generates the main activity of the Sipdroid SIP client. It was edited for executing a call to 767 automatically when the main sipdroid activity starts we will execute call_menu(AutoCompleteTextView sip_uri_box) for performing a call to 767 automatically.

- org.zoolu.sip.call.ExtendedCall.java:

The method callNG911 was created to handle NG9-1-1 calls. The method doesn't change any call parameters, but redirects the flow of the application calling the method inviteNG911 inside InviteDialog.java.

- org.zoolu.sip.dialog.InviteDialog.java:

The method inviteNG911 was created in this class to handle NG9-1-1 calls. When it gets a response from the server, embeds the location inside an INVITE message using the method createInviteNG911 defined in NG911MessageFactory.java inside the com package.

- org.sipdroid.sipua.SipdroidEngine.java:

In this class, the method call has been modified for recognizing the call number and, if it is 767, executing the call method of the UserAgent.java class.

- org.sipdroid.sipua.UserAgent.java:

The call method was modified for recognizing NG9-1-1 calls and executing the methods which will lead this call to send the indoor location to the PSAP. A new Boolean attribute was added in order to recognize if the call is an emergency call or it is not. If it is an emergency call to the 767 CallNG911 method in the ExtendedCall class will be called.

- org.sipdroid.sipua.ui.Settings.java:

Additional settings options were added to the Sipdroid Settings (SIP Settings). Those settings are user information such as Name, age, language and medical information. Also, it includes optional Facebook registration for sending automatically the user information to the call dispatcher at the PSAP.

4.2 Design and Structure

Design

Java programming language, using the Android SDK (Software Development Kit), has been used for the development and implementation of the NG911 Bluetooth Indoor Location App. Screenshots of the NG911BL App developed are shown in the figure below.

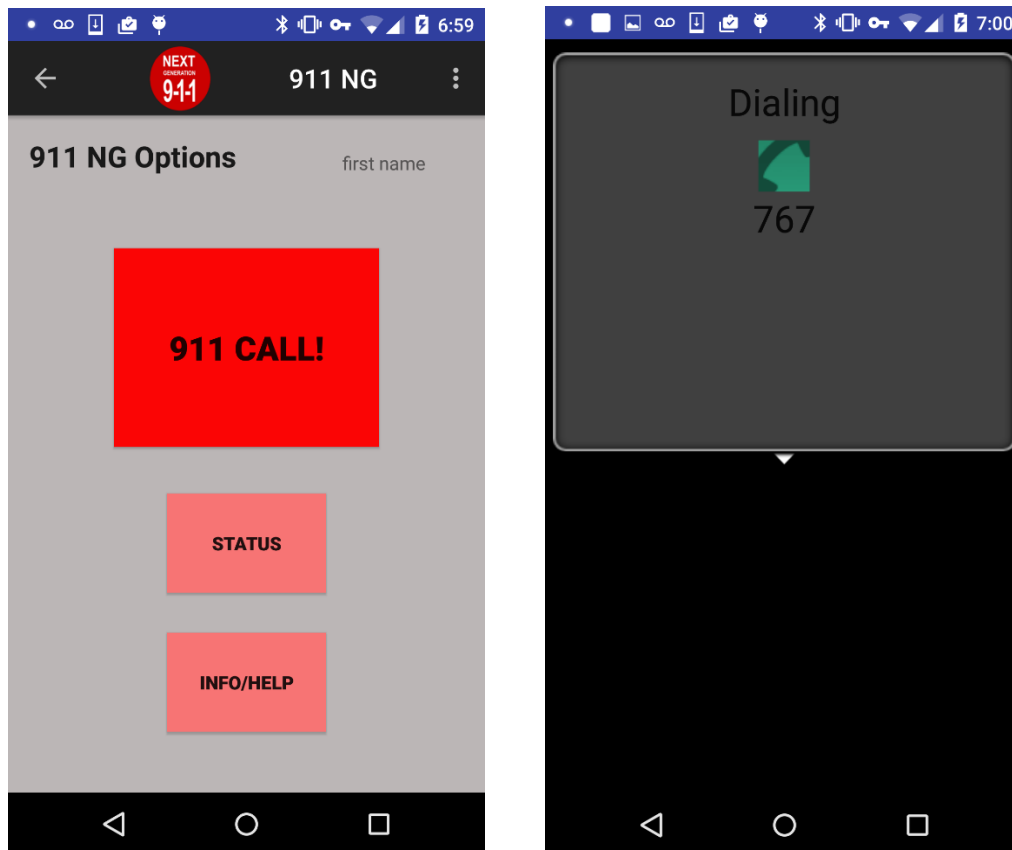


Figure 18. Screenshots of the NG911BL App. Main activity at the left and Sipdroid call at the right.

In this figure 12, the user is currently pressing the “911 CALL!” button and; after capturing the beacons in the environment and receiving the its indoor location; the screenshot at the right will appear. This means that the call is currently ringing at the PSAP. After that, the call dispatcher will have to accept the call and, once it is accepted, he/she will see the indoor location of the call.

Although there are additional options such as the status or information buttons this App has been developed to be executed without a needed pre-configuration. In other words, if a user has installed the App but never used it, he/she will be able to make a call and reach the PSAP in less than 8 seconds by pressing the “911 CALL!” button only.

Also, this application runs in the background listening to the cellphone dial pad. If the user calls as he would normally do, dialing 9-1-1, the android app will auto execute the NG911BL application and send the indoor location to the PSAP.

Additionally, other activities have been developed to bring emergency awareness to the user as well as testing the 911 system. Our goal was to bring more information that what a call dispatcher has nowadays. Although, it already sends the indoor location, we wanted to add another feature for filling user information which could be useful for the call dispatcher at the PSAP.

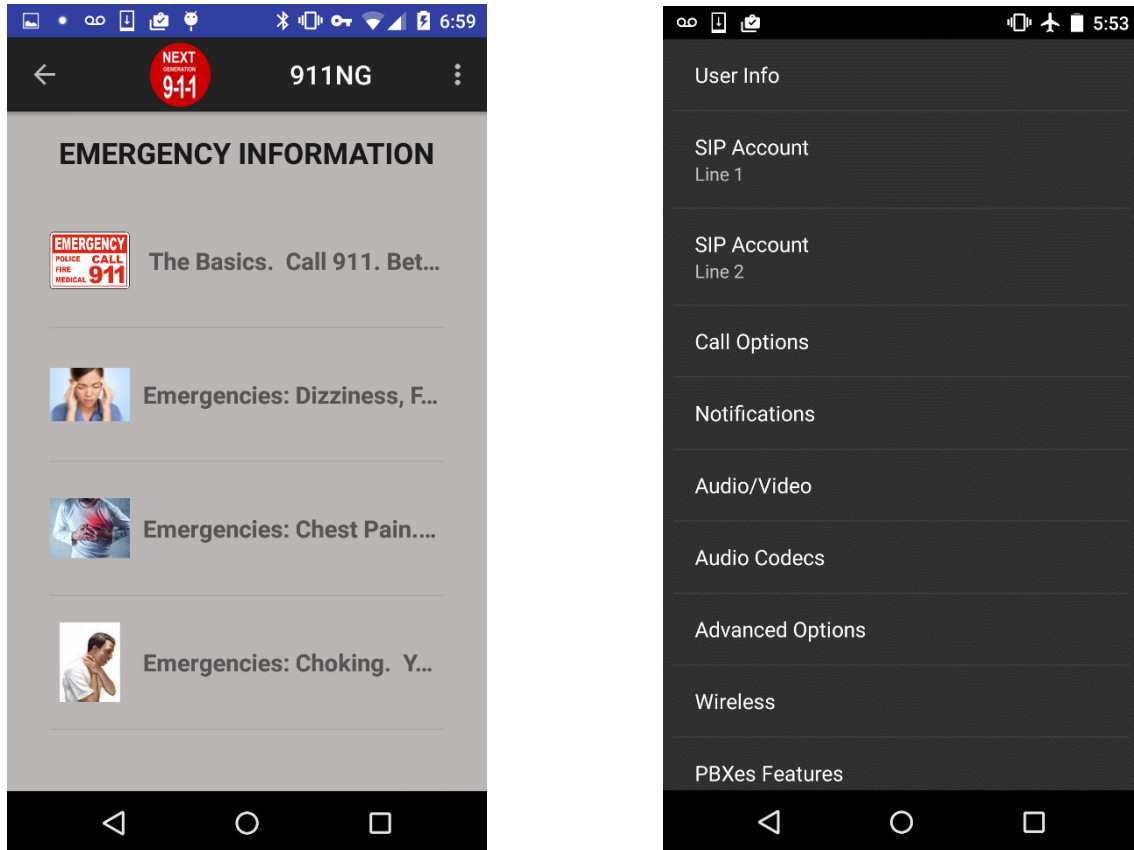
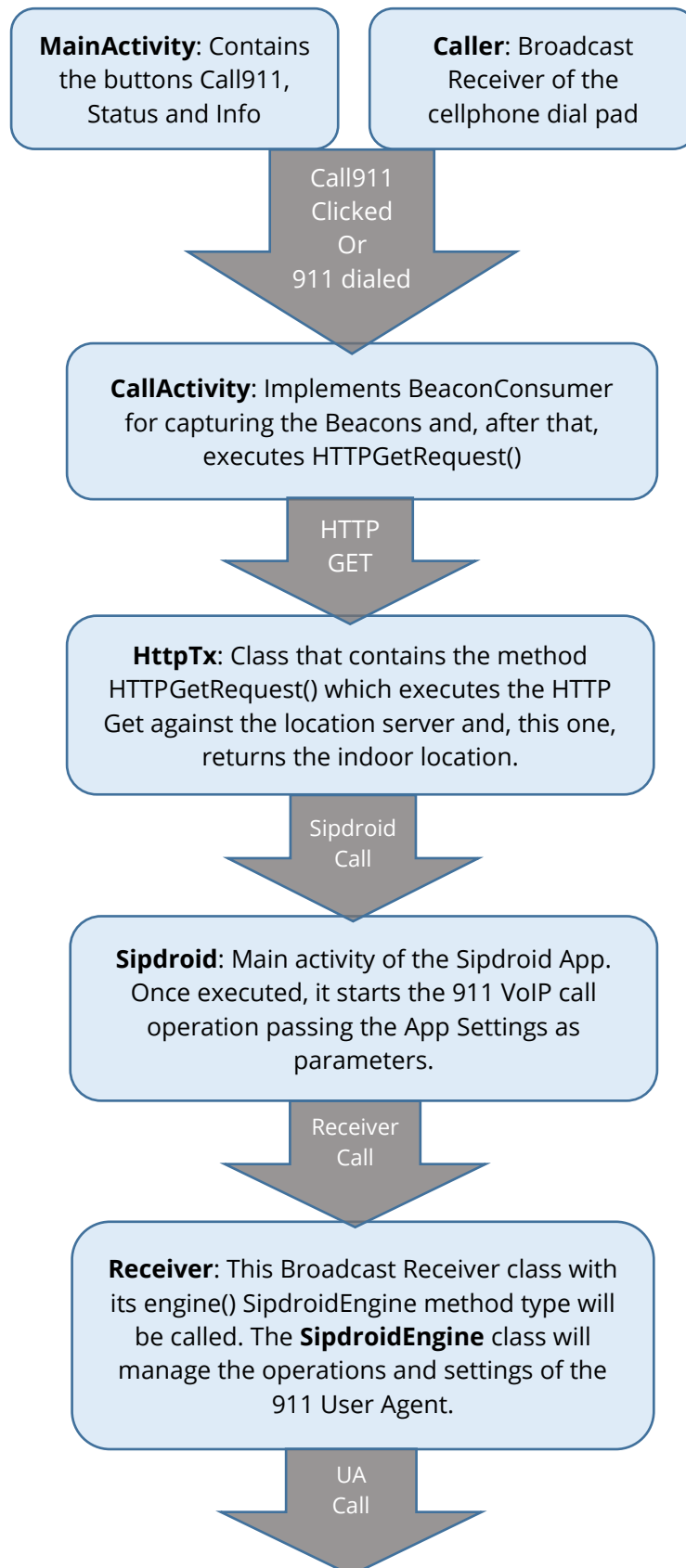


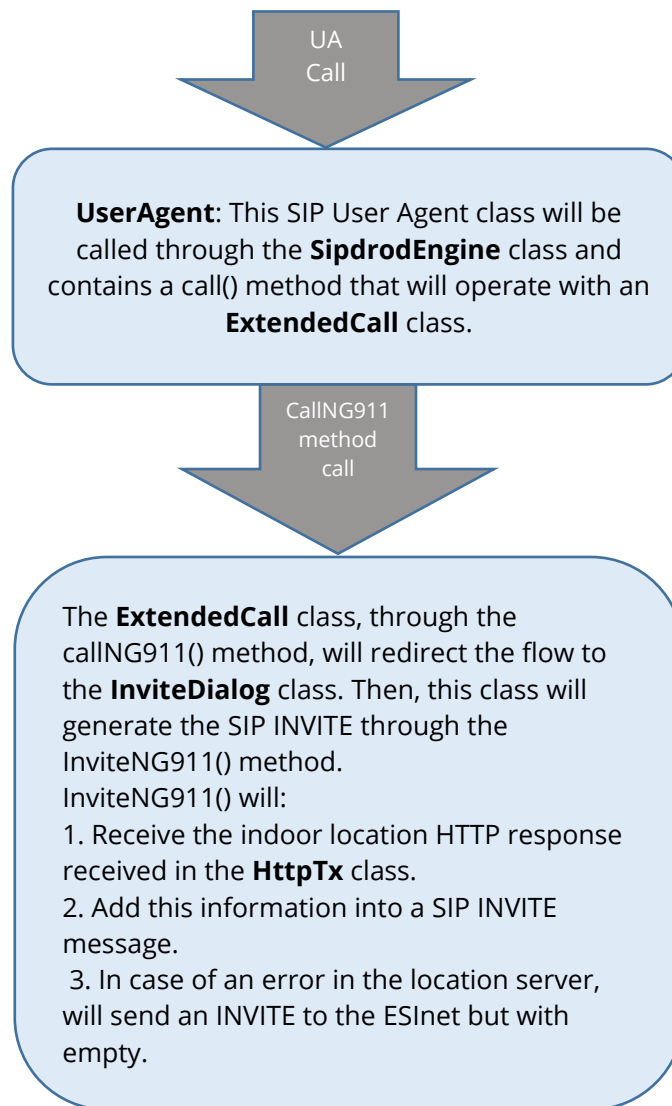
Figure 19. Screenshots of the NG911BL App. Information activity at the left and the Settings activity at the right.

Shown in the Figure 13, the user can read some Emergency Information (at the left) or can go to Settings and fill the user information (at the right). Although not yet fully implemented, this user information would be sent and parsed in in the PSAP, reducing, therefore, the call dispatcher response time.

Structure

On the one hand, based on the software code, the NG911BL application core will be structured with the following class modules:





On the other hand, based on the executed operations, since the user clicks the “call911” button until the dispatcher receipts the indoor location, the system will have the below flow diagram.

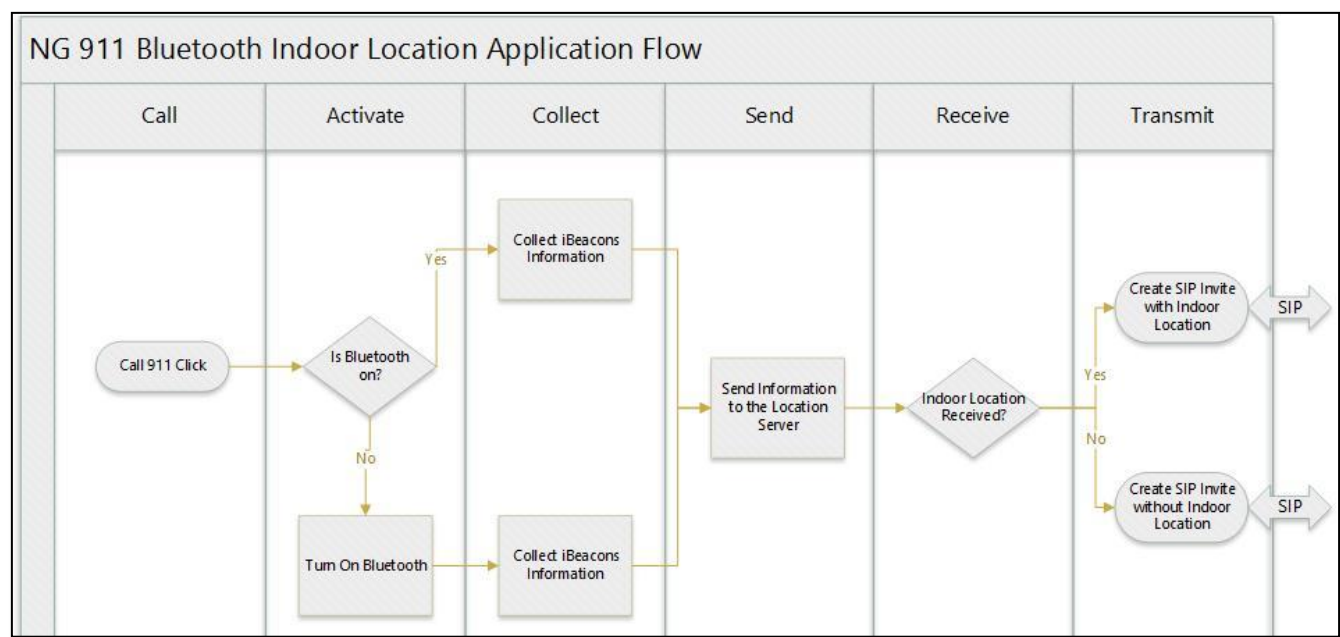


Figure 20. NG911BL flow diagram

First, after starting the call, the application will be capturing all the advertised Beacons from the surroundings and sending this captured information to the location server. Before sending this information NG911BL will be filling a JSON array with JSON objects which will contain the identifiers and RSSI of each iBeacon advertisement package. After filling this array during 8⁸ seconds, this array will be sent as HTTP GET to the location server.

As the location server estimation needs all the captured values in order to obtain a better location estimation, in this case, we will send the entire JSON array in one only package at the end of the 8 seconds. Instead of sending multiple HTTP GETs each time we receive a package, we will send an only HTTP GET one time as we can't do anything until we have all of them.

We chose JSON data exchange formatting because it is faster than other formats, such as XML, it is more efficient way of transferring data and it is a very easy method for parsing data.

This process would include the stages: Call, Activate, Collect and Send.

After that, the application will receive the server response and will create the SIP INVITE with the indoor location sent by the server. In the previous figure we can see this process in the stages: Receive and Transmit.

⁸ We obtained this time value through testing the minimal time which allowed the cellphone to scan and detect devices, to send a HTTP GET to the server and to obtain a response.

5. Bluetooth Indoor Location Algorithms

Although this project is not focused on the research and development of an indoor location algorithm, for obtaining the first test results we used some basic algorithms which will be explained below.

It is possible to estimate the indoor location of an object by using the signal AOA - Angle of Arrival, TOA - Time of Arrival or RSS - Received Signal Strength. However, in this first stage of the project we used indoor location algorithms which use the Received Signal Strength only.

The signal strength depends on distance and broadcasting signal strength value. However, knowing the transmission power of the iBeacons is not an important factor in our case. As we are not trying to estimate the exact location we do not need to use triangulation or fingerprinting techniques. In our case, we are using the received signal strength as a relative value to compare against other signal strengths that the cellphone received. As all the devices are sending approximately the same power (all iBeacons are made with the same components) we won't need to estimate distance but compare maximum received powers or average powers.

Variability of the signal strength is high. External factors influence radio waves. Absorption when going through different materials, interferences from other radio waves or diffraction are some of the factors that cause fluctuations in the RSSI. Even in the best case scenario, having direct line of sight we will obtain signal strength variance of 3.5 dB. In a more typical scenario, without direct line of sight to the beacon and being 5.5 meters away from the iBeacon we can see variances of 15 dB. The further away the device is from the beacon, the more unstable the RSSI becomes. Therefore, we need a lot of iBeacons to be as near as possible to the caller.

Regarding the coverage or range of the iBeacons in an indoor location without having direct line of sight and in a location without very thick walls, less than 20cm thick, has been proved to be around 15-25 meters in real world conditions.

5.1 Maximum Power Algorithm

This algorithm is the default algorithm used in the location server of the system. It receives all the single RSSI values -in dBm- from the iBeacons in range and the algorithm chooses the highest received power in this location. The location of the device which sent a highest power will be the location of the caller.

First we will have a table like the Table 2 with all the location numbers and its human readable corresponding locations.

Location Number	Minor	Acronym	Floor	Room Type	Room Number	Nearest wall
21	154	SB	1	NorthHallway	105	NW
22	158	SB	1	NorthHallway	107	N
23	193	SB	1	NorhtHallway	108	N

24	197	SB	1	NorthHallway	110	NE
25	167	SB	1	EastAtrium	111	N

Table 2. Location Number, device, building human readable location relationship

For example, standing in the location Number 21 (SB building, Floor 1, NorthHallway room 105) we obtained the Table 3 advertised identifiers and RSSI ordered from highest to lowest value.

Major	Minor	RSSI	Location Number	Timestamp
101	154	-41	21	7/14/2016
101	158	-49	21	7/14/2016
101	154	-56	21	7/14/2016
101	179	-68	21	7/14/2016
101	193	-69	21	7/14/2016
101	179	-70	21	7/14/2016
101	193	-70	21	7/14/2016
101	158	-70	21	7/14/2016

Table 3. Example of the seen iBeacons seen during a test at the location number 21

As we can see in the Table 3, we will see repeated advertised Major and Minor identifiers from different iBeacons, for example, the cellphone received two Bluetooth advertisements with RSSI -41 dBm and -56 dBm. When using the maximum algorithm, we will be choosing the identifiers of the signal which sent the highest power. In this case the location of the caller is next to the iBeacon with Major:101 and Minor:154.

Finally, we have the nearest iBeacon identifier and we will be querying the database to obtain the location of the device with Major:101 and Minor: 154. In this case the location of this iBeacon; as shown in the Table 2; is 21, therefore, the location of the caller when doing the test was properly estimated with maximum power algorithm.

5.2 Maximum Average Power Algorithm

We will have a data set of iBeacons detected for each of the locations. At each location the cellphone might receive more than one Bluetooth advertisement from the same device.

First we will group by minor and average the received power. As the Bluetooth waves are non-continuous waves and taking into account that dBm are logarithmic units, we will have to convert this power to W and apply the average operation.

$$P(\text{dBm}) = 10 * \log(P_r(\text{mW}))$$

Equation 1. Power P (dBm) depending on Pr (mW)

For example, as shown in the Table, being 0.7 meters away from a static iBeacon we will see that there is the following difference between averaging the power in dBm or in Watts.

Method	P(dBm)	P (dBm) ⁹
Average	-46.89	-45.67

Although there is not a huge difference, we will be using the Equation 1 for converting dBm to mW in order to be more accurate. Then, we will apply this conversion into our script and we will group the average RSSI values.

```
SELECT TestLocationNum, minor, avg(power(10,rsi/10)*power(10,6)) as avg_
FROM BluetoothTestVals
group by TestLocationNum, minor
order by TestLocationNum, avg_ desc;
```

And we will obtain a table like this reduced version table.

Average Power
per Testing
Location Number
(E.g.: 21) and
Minor (E.g.:111).

}

Location Number	Minor	Average Pow (nW)
21	111	0.219
21	154	40.972
21	158	0.789
21	169	0.681
22	111	0.011
22	158	2.513
22	186	0.048
23	193	2.001
23	149	0.001
23	174	0.053

Table 4.First stage of the Average Algorithm Table, grouping by location and minor id.

Now, we will choose the highest average value per location number¹⁰ and we will obtain the following reduced version table.

Location Number	Minor	Average RSSI (nW)
21	154	40.97
22	158	2.51
23	193	2.00

Table 5. Final Table of the Maximum Average Algorithm Table, grouping by location and minor id

In this case the user was at the location 21 of the Table 2, and was estimated to be next to the iBeacon with Minor identifier 154.

Finally, we checked the location of the iBeacon with Minor 154 and effectively it was at the location 21.

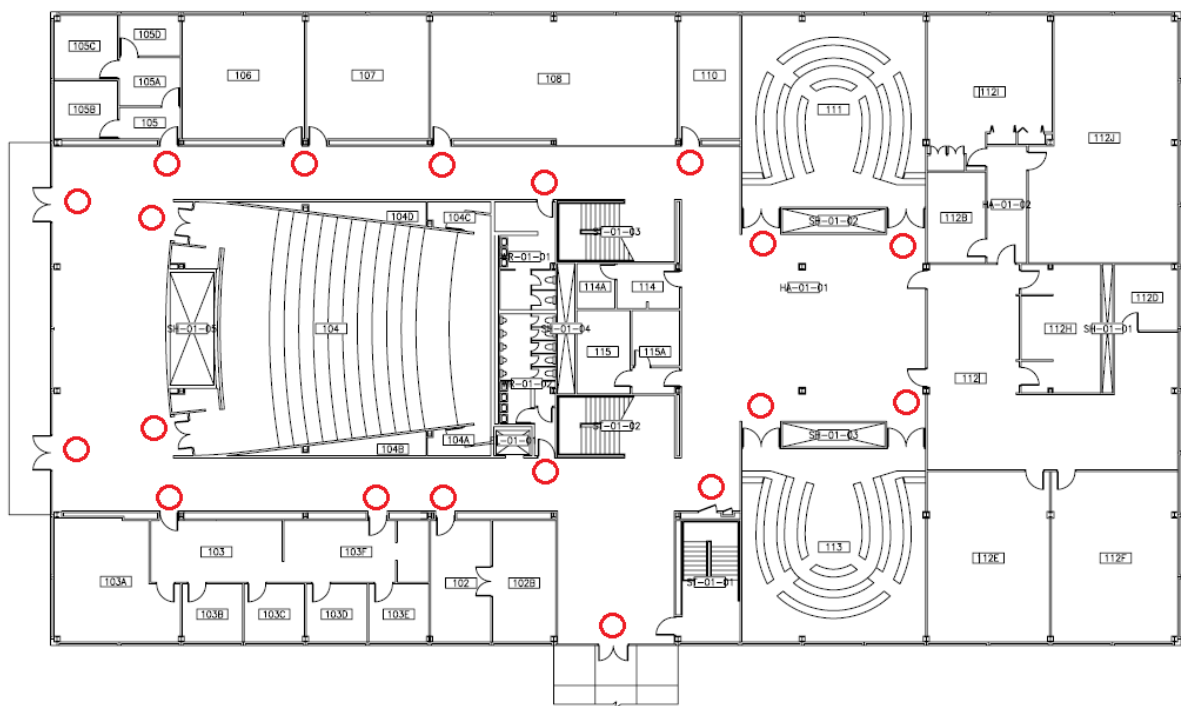
⁹ Values obtained doing the average in W units instead of doing it directly in dBm.

¹⁰ Complete SQL Scripts are available in the Appendix F1 of this document.

6. Proof of Concept

6.1 Deployment

For implementing our first proof of concept of the NG911 Bluetooth indoor location, we will deploy our devices into the Stuart Building at the Illinois Institute of Technology. This is a 3 floors building with high ceilings and multiple rooms and walls. Its approximate general measures are 60 meters wide and 35 meters' long. The iBeacons distribution will be based on the iBeacons range -Covering the maximum building area with the minimum number of devices- and, at the same time, achieving an organic distribution where we will follow logical patterns depending on the building architecture.



from the floor. However, in our case, we didn't have the permission for having the devices visible in the building so we had to attach the iBeacons on the false ceiling, about 3 meters above the floor. This won't maximize gain but, as all out iBeacons will be in the ceiling, the attenuation produced in each iBeacon will be; in theory; the same for each one, therefore, the indoor location accuracy won't be affected.

But the physical deployment isn't all the work. We also studied the operational cost of deployment and we analyzed and build a solution to ease the deployment process.

First, we used the map shown above with numerical values written over it; values or identifiers that would show us where to deploy the iBeacons. At the same time, for this system to work, we need to have each iBeacon identified and located in one of the deployment locations in our map. Having to follow the following process:

- 1) Identify Each iBeacon.
- 2) Identify each deployment location.
- 3) Deploy each iBeacon in a deployment location and update the database accordingly.

Additionally, for the identification of the iBeacons and the location definition we should follow the same convention.

Convention

For the deployment we agreed on a unique procedure in order to have everyone following the same steps when deploying the iBeacons. We standardized a naming convention and a deployment procedure.

Regarding the naming convention for the location we would use the following fields: A location number for each of the deployment locations or circles represented in the Figure 16. And each of this location numbers will have an Acronym (Building acronym), Floor, RoomType, RoomNum, NearestWall and Comment.

Deployment App

As it can be seen in the tables below, we will have one table with the information of each iBeacon, one table which makes a relationship between the device ID and the iBeacon Location and, one last table which will contain the information of each location.

The process for deployment will be:

- 1) In the first table the three first fields are auto generated and we only have to manually assign a major and minor to a device and label the box accordingly.

ID	MAC	Identifier	Major	Minor
1	6c:0b:84:59:1c:c9	3e0022000847343337373738	100	1
2	6c:0b:84:5a:ec:b5	210026001747343338333633	100	2

- 2) Through the deployment app we will be able to modify this table and assign to a specific iBeacon device a location number which corresponds to a specific location in the building map. For example, in this case we are deploying the iBeacon with major 100 and minor 1 at the location number 2

ID	LocationNum
1	②
2	3

Finally, although we manually introduced the location information in this table, we believe that it would be possible to use software to automatize this process. An application with a map and clickable location numbers over it might be a possible approach.

LocationNum	Acronym	Floor	RoomType	RoomNum	NearestWall
1	SB	0	NorthHallway	010	W
②	SB	0	NorthHallway	011	N

Through the usage of numbers which uniquely identify a location, we avoid confusion when describing the location of an iBeacon, we will substitute old iBeacons with new ones efficiently and will be able to use NG911BL to estimate the location of an emergency caller with the updated values.

6.2 Data Set Generation

Once we have the system working we will have to test it. After the deployment we will generate data sets to be analyzed in the future. Those data sets will contain information of the detected iBeacons in different preselected testing locations. With this data sets, generated with an app specifically created for the purpose, we will be able to test the system, research about the optimum algorithm and compare this system with other solutions.

For testing the system, we will run several tests positioning ourselves in different locations inside the building such as it is represented in the Figure 17. In our first test our location will be next to the devices and in our second test we will be in between of 2 or more devices in order to get the results of a worst case scenario.

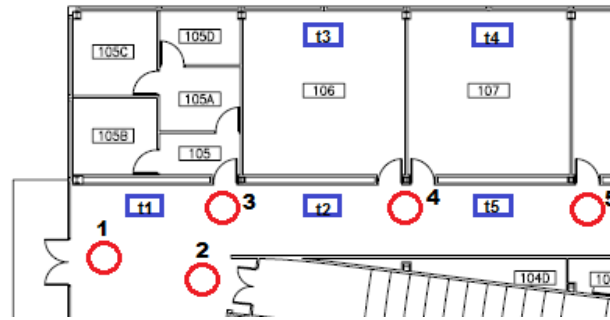


Figure 22. Deployment and Testing distribution example

In the figure we can see the deployed iBeacons (red) and the testing locations where we will run our tests (blue).

Convention

For the testing of the NG911BL system we followed the same convention than the one used for the deployment of the iBeacons but, in this case, we gathered additional fields of information such as the device running the test app in order to differentiate and be able to execute several tests simultaneously.

The tester will go to one of the testing locations previously assigned in a map (blue), and later he will use the app to add its testing location number (E.g.: t1) and run the test.

Below, in the Figure 17, we can see an example of the table Tests where we have a list of tests run with the same device 4 times.

TestNum	PhoneMac	DeviceModel
1	30:76:6f:fb:4f:2e	LGE-Nexus
2	30:76:6f:fb:4f:2e	LGE-Nexus
3	30:76:6f:fb:4f:2e	LGE-Nexus
4	30:76:6f:fb:4f:2e	LGE-Nexus

Figure 23. NG911BL tests table example

At the same time this tests will be related to a table containing all the information of each test. Below we have the table Bluetooth Tests where, when running the test 1 in the location 24, we received one single Bluetooth probe request from the iBeacon with major:100 and minor:4 with a signal strength of -44 dBm.

pid	major	minor	rsi	TestNum	TestLocationNum	captured_time
1	100	4	-44	1	24	2016-06-27 19:10:43
2	100	4	-35	2	24	2016-06-27 19:11:29
3	100	4	-35	2	24	2016-06-27 19:11:29
4	100	4	-34	2	24	2016-06-27 19:11:29
5	100	4	-31	2	24	2016-06-27 19:11:29
6	100	4	-31	2	24	2016-06-27 19:11:29

Figure 24. NG911BL testing results data set example

With this information and the information of the location of each device, it will be possible to analyze this sets of data and extract the degree of location accuracy and other aspects of the NG911 solution.

NG911BL Indoor Location Testing App

This Application will be designed for the generation of data sets in different locations of the building. In the future those data sets will be used for analysis purposes, for example, the percentage of indoor location failure or precision of the system in the estimation of the indoor location.

The tester will open the Android App and the following UI will be shown:

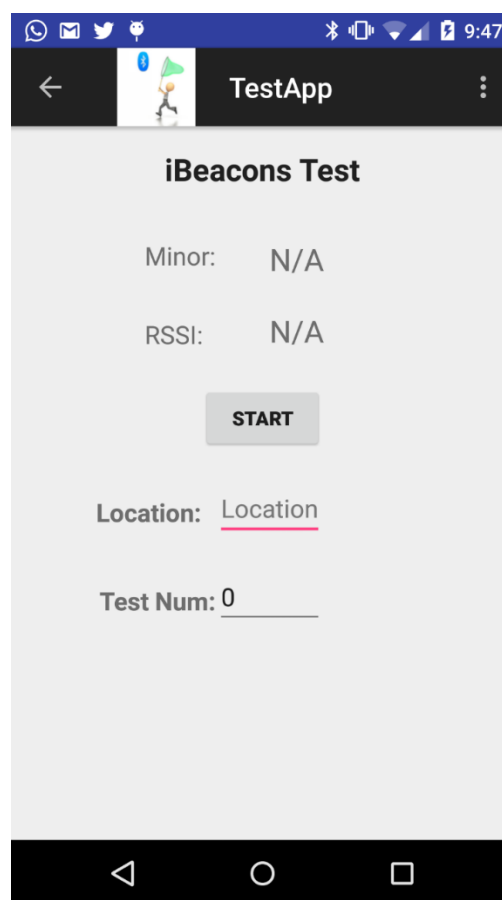


Figure 25. NG911BL Testing App. Application for data set generation.

As we can see in the Figure 25, for the generation of each data set, the tester will introduce his location and will press start. What this button will do is execute a Bluetooth LE scan of the environment for 10 seconds.

After that the tester will be able to run another test in the same location or move to another new location signaled into the map with a t and a number.

For simplification and organization of the deployment and data set generation process, the following flow chart was used:

IBEACONS DEPLOYMENT AND DATA SET GENERATION

NG911 Bluetooth Indoor Location

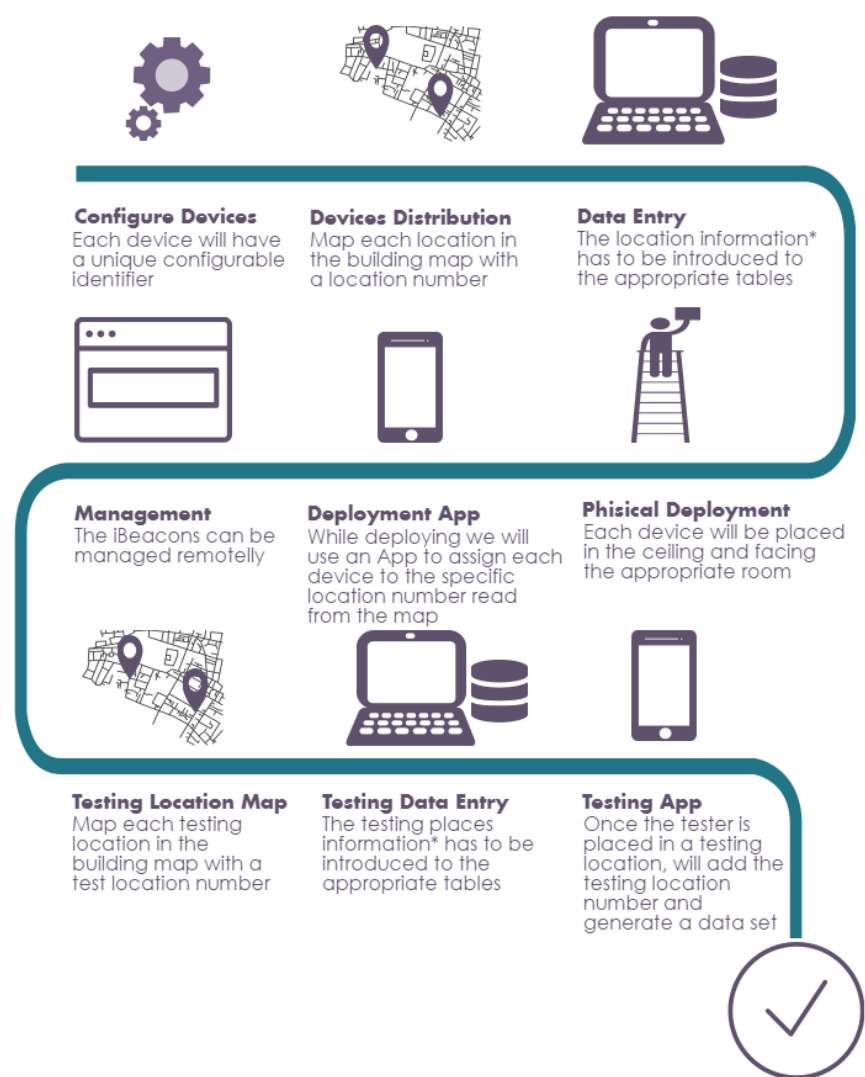


Figure 26.Deployment and Data Set Generation Flow Chart

This figure summarizes the process that we followed once our NG911BL was working properly in our Lab. It is important to take into account that, although we obtained a working solution, this system requires us to follow the tasks shown in the Figure 21 in order to have it properly working in a real world environment.

6.3 Test Results

After several test in different environments we could confirm that the latency or calling to PSAP time is about 8-10 seconds. From this call establishment time more than 75% of this time the application is capturing advertised iBeacons and communicating with the server. And more than 2/3 of this time the cellphone is capturing iBeacon packages.

We can predict that this time could be notably minimized, however, the goal of this project wasn't code optimization and, as we exceeded by far the FCC requirements, we prioritized other tasks.

Also, if we are in a very busy environment with 7 or more iBeacons we might need to consider the time discovery delay effects and to let more time to the cellphone to gather iBeacon package. For example, the difference between the time needed to discover 3 iBeacons in region and 10 iBeacons in region is more than 1 second [10].

About 25% of the Next Generation 911 emergency call operation will be due to the server algorithm operation and the cellphone-server communication.

We did not perform any stress test to the server as it is being used for a proof of concept and not for a final large scale implementation. This project is an easy scalable solution which would allow us the improvement of the current specifications of the server if needed.

Also, in our test results we proved that the battery drain was of the order of 10 percentage points per day using a management system which checks the status of the iBeacon in an hourly basis. After a lot of research being done by the team we were able to reach a battery drain under 1 %, of the full battery, per day.

For this first set of tests we will be defining success when the location error is 10 meters or less and the caller is estimated to be in the right floor. Taking into account that the FCC requires a minimum of 50 meters' accuracy, we are being more ambitious and going above and beyond those requirements that, we predict, will change to increase accuracy requirements in a near future.

Results

Test A

In this first test we obtained the accuracy of the system being next to one of the iBeacons when doing the test. We will define different locations with a location number and we will run the tests in each of this locations to prove the accuracy of the system using Maximum Signal and Maximum Average algorithms.

For example, the NG911BL tester will run the testing app in the location 40 shown in the Figure 28. Once in this location the cellphone will capture all the Bluetooth advertisements within the range. The generation of the data set in each location will be taken within a 10 seconds time period.

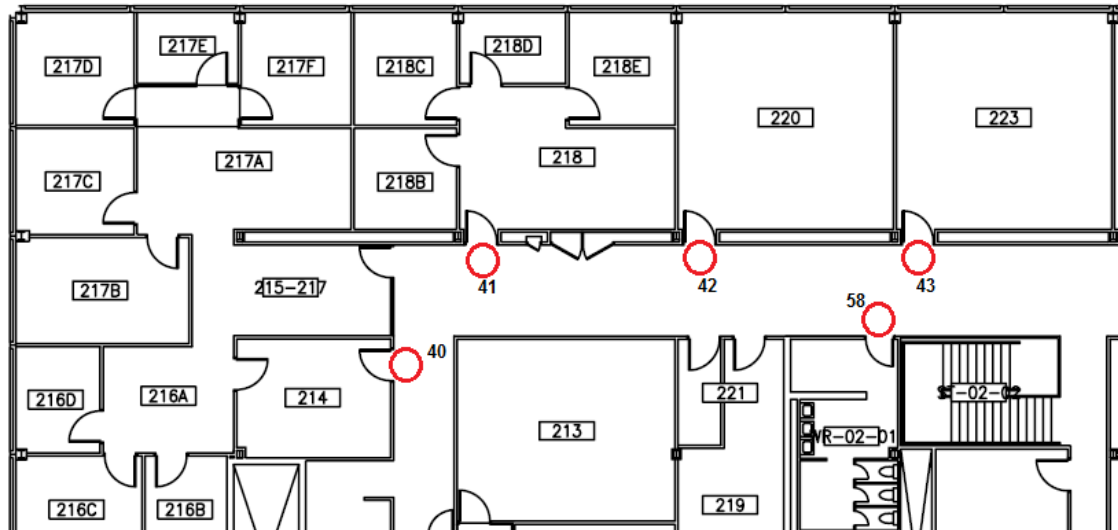


Figure 27. Stuart Building Second Floor. Test A

The Bluetooth LE advertisement packages received by the cellphone had the minor identifiers and RSSI shown in the Figure 29.

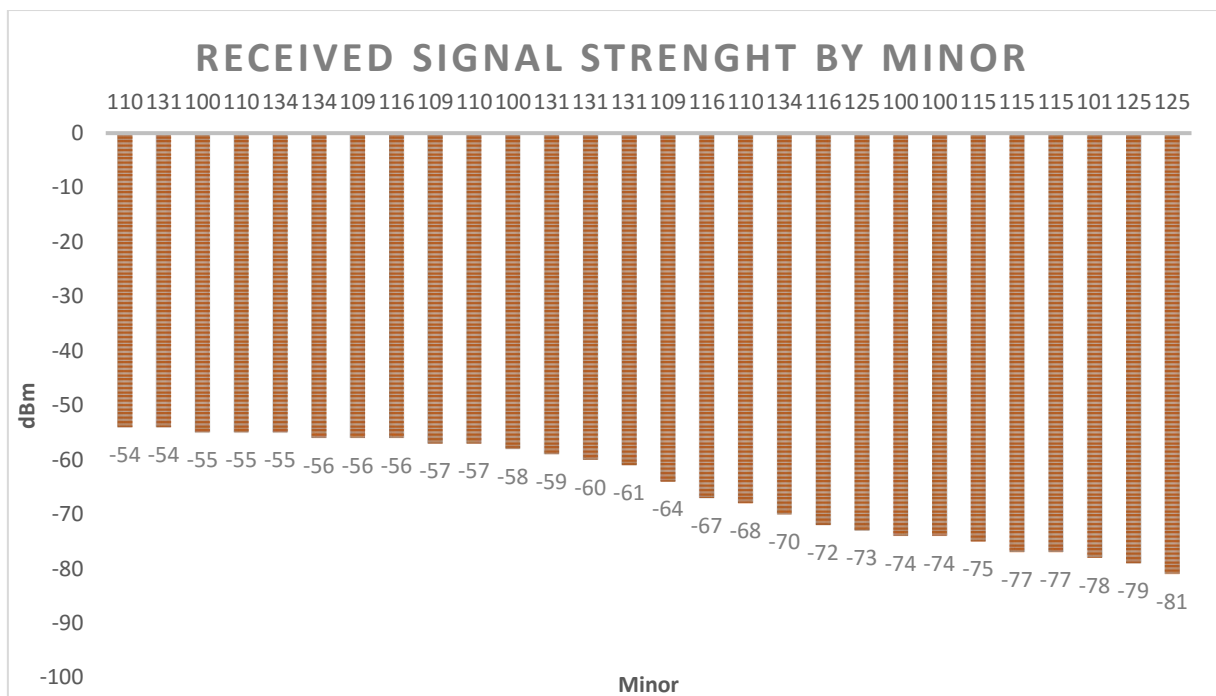



Figure 28. Received iBeacon Identifier-Power at the location number 40

As we can see, the cellphone received a maximum power of -54 dBm from the iBeacons with minor 110 and 131.

In this situation, the highest power algorithm based on choosing the device sending a more powerful signal wouldn't be enough in order to estimate indoor location with room accuracy. However, 110 and 131 minors correspond to the locations 40 and 41 (Figure 23), therefore, in this test case we reached an accuracy of less than 3 meters in the XY plane.

In other tests we got better accuracy. For instance, when doing the test in the location 41 it was possible to estimate the location with an error of less than 2 meters. The highest power was from the iBeacon in the location 41.

After analyzing all the data obtained in this first test in two dimensions we can conclude that when we are near to one of one iBeacon, 2 meters or less, we can be located with accuracy under 2 meters 64% of the times. In the other 36% of the tests cases the location of the tester is estimated to be next to the second nearest iBeacon or in between (as explained in the above example in the location 40).

Location Number	Minor	RSSI		Location Number	Minor
40	110 / 131	-54		40	110
41	131	-61		41	131
43	109	-54		43	125
45	126	-62		45	126
47	106	-52		47	106
49	123	-57		49	105
51	121	-41		51	112
52	107	-41		52	121
53	134	-52		53	134
54	116	-46		54	116
55	101	-67		55	101
56	115	-58		56	115
57	127	-54		57	127
58	100	-52		58	100

64.29%	Tests where the user was estimated to be less than 2 meters away from his real location.
100%	Tests where the user was estimated to be less than 10 meters away from its real location.

Table 6. Test A indoor location results by location. Obtained results (left) and ideal results (right)

In the Table 4 we can see in light green the times when, while being 2 meters from an iBeacon, another iBeacon was detected as the nearest one. In other words, the cellphone received the highest power signal from another iBeacon than the nearest one. However, in all of the tests the tester was estimated to be located not farther than 10 meters from its real location.

Also, applying the Maximum Average Algorithm we obtained better results than with the Maximum Power Algorithm. More than 85% of the tests offered us indoor location with less than 2 meters' error.

In sum, in the test A, in two dimensions, the indoor location offered is accurate with indoor location estimation errors generally ranging between 1 to 5 meters.

Test B

For this test we will analyze the indoor location accuracy offered in a 3D environment, this is being in a 3 floor building; with a basement, first floor and

second floor; and we will be doing the test in the first floor. Also, as we did in the test A, we will be doing the tests next to the deployed iBeacons. In other words, we are testing the best case scenario.

We did this test following the same steps that we followed for the Test A and first, using the Maximum Power Algorithm, we obtained the results of the Table 7.

Location Number	Minor	RSSI		Location Number	Minor
21	154	-41		21	154
22	158	-56		22	158
23	111	-55		23	193
24	197	-51		24	197
25	167	-69		25	167
26	189	-65		26	189
27	132	-65		27	132
28	174	-63		28	174
29	194	-57		29	194
30	155	-56		30	152
31	147	-47		31	147
33	148	-54		33	148
34	169	-53		34	169
35	180	-51		35	180
36	195	-59		36	195
37	179	-55		37	179
38	186	-62		38	186
39	111	-58		39	111

Table 7. Test B indoor location results by location. Obtained results (left) and ideal results (right)

Based on the Table 7, we proved that we can reach results where 94.4% of the locations are estimated to be less than 10 meters away from its real location. In the other 5.6 % of the tests, when using the Maximum Power algorithm, we located the caller in the right building but in the wrong floor (light green).

Using Maximum Average Algorithm, we improved our results. In this test we were able to estimate all the tested indoor locations of the caller with errors under 2 meters and always in the right floor, in other words, we always obtained the user nearest iBeacon as the user location.

Test C

For this test we also made our test in 3D but in this case we weren't next to an iBeacon. In this case we were positioned in random locations in between different iBeacons to obtain results of a more realistic scenario. This test was made in all the floors of the Stuart Building ¹².

Using the Maximum Power Algorithm, we obtained the following results. First, 100% of the times the user was located in the right floor, (as the floor height is about 3 meters) this leads us to an accuracy below 3 meters in the vertical plane. Also, based on our definition of successfully indoor location, we obtained 96.4 % of success rate and, if we enlarge the location ring to 20 meters, we would be talking about a 100 % of success.

On the other hand, using the Maximum Average Algorithm we reached the same level of accuracy, 96.4 % of the times the tester was successfully located. Although in this case we are more accurate, reducing error estimation in some cases, the percentages success rate still is the same than with the Maximum Power Algorithm. Also, with 100 % of the locations were estimated in the right floor and being less than 20 meters away from the real tester location.

In conclusion, we got very encouraging results in all 3 test sets and we met all the requirements of the FCC for the year 2021. This means that our system can become a viable solution for CRMS providers which have to meet those requirements and, at the same time, want to offer an indoor location solution as a service which can provide indoor location with room level identification to dense urban buildings.

¹² Deployment Location Maps available at the Appendix E1

7. Related Work

7.1. Commercial Solutions

The 911 Emergency Services market is a specific slow market where government and private companies melt. Some progress has been made in order to stablish standards and rules for the Next Generation 911 emergency system. However, these necessary standards are still in development.

As the standards and norms hasn't been fully specified there are several solutions which aim to provide indoor location for emergency services. The Communications Security, Reliability and Interoperability Council (CSRIC), advisor and recommendation provider to the FCC, identified 3 major companies working on 911 indoor location [12]: NextNav, Qualcomm and Polaris.

NextNav uses GPS-like technology for providing indoor location. They deploy beacons transmitting in the 900 MHz licensed band. Those beacons will be sending the location information to the receiver and this one will be computing the information.

Another company that also worked with GPS technologies is Qualcomm. However, in this case the add complementary wireless networks technologies, it is a hybrid solution. It uses GPS and cell tower data to enhance location accuracy.

Regarding Polaris, RF pattern matching (RFPM) will be used for providing indoor location. This solution uses what is called fingerprinting. This methodology is based on measuring and comparing mobile measurements (signal strength, time delay, etc.) against a known referenced location.

All those solutions offer very interesting solutions are easy to implement and they rely on currently available technologies, however, would be below the levels of location accuracy we are reaching with our solution.

Horizontal Error Statistics (m)								
Building ID	Total Number of Calls	67 th	90 th	95 th	Average Error	Standard	Max Error	Min Error
		Percentile	Percentile	Percentile		Deviation		
NextNav_All Dense Urban Buildings	4859	57.1	102.4	154.0	57.5	64.9	1059.2	0.6
NextNav_All Urban Buildings	4238	62.8	141.1	196.1	69.5	99.9	4367.2	2.1
NextNav_All Suburban Buildings	3581	28.6	52.9	62.2	27.2	99.7	5854.2	0.4
NextNav_All Rural Buildings	820	28.4	44.9	60.3	70.3	1231.5	35255.9	1.5
Polaris_All Dense Urban Buildings	5372	116.7	400.1	569.3	150.3	193.3	1656.1	2.2
Polaris_All Urban Buildings	3874	198.4	447.8	729.9	203.0	225.9	3131.9	0.4
Polaris_All Suburban Buildings	3489	232.1	420.7	571.4	215.1	161.9	1089.1	8.4
Polaris_All Rural Buildings	726	575.7	3005.1	3072.3	845.6	961.3	5809.2	66.2
Qualcomm_All Dense Urban Bldgs	5145	155.8	267.5	328.1	136.4	94.7	722.5	0.5
Qualcomm_All Urban Buildings	4338	226.8	449.3	507.1	233.9	547.7	18236.7	1.6
Qualcomm_All Suburban Buildings	3716	75.1	204.8	295.7	92.0	173.6	4639.4	0.2
Qualcomm_All Rural Buildings	709	48.5	210.1	312.3	639.9	2999.2	27782.4	1.0

Table 8. NextNav, Polaris and Qualcomm Horizontal Error Statistics [12]

Also, there are other companies working on more accurate solutions for indoor location positioning using Wifi technologies. Solutions that clearly are more similar to our solution.

One of those companies is True Position [13]. True Position uses Skyhook Wireless Wi-Fi solution which combines GPS and Wifi beacons. Through a Wifi location database they are able to identify a location with high levels of accuracy. However, this is a solution uses proprietary software and hardware which reduces flexibility of the solution. Also, although Skyhook mentions that their system could work for emergency services, it is not clear if their system is ready to be integrated with a real world Next Generation (NG) 911 Network module.

7.2 Research

Regarding the research that has been done during this recent years, there are multiple academic research papers that aim to solve indoor positioning.

A lot of research has been done around Wifi Indoor Location. Some more concerned about the algorithm implementation and some others in the technology used or the calibration methodologies. An interesting approach for Wifi RSSI based indoor location is the usage of fingerprinting following organic deployments, taking into account the architecture of the building and behavior of the people on it [14]. Although we used an organic deployment for our devices, we did not research about possible fingerprinting algorithms and we followed an indoor location survey-driven deployment instead. Also, although similar to fingerprinting, our approach is a more simplistic algorithm which has the benefit of not requiring fingerprint calibration.

Other proposed systems use Bluetooth LE technology [15], [16]. Usually those systems will use RSSI to estimate the distance of the device and use fingerprints or triangulate its position. Although it can be very accurate (accuracy to less than 1 meter), those techniques need calibration and have a high dependence on the environment. In our case we do not use distance information but we basically use the advertised iBeacons and its RSSI to estimate the nearest location.

Another indoor location research is based on using barometric sensors for improving location accuracy [17]. Measuring the difference in the barometric-pressure is possible to recognize the floor of the cellphone user. However, although this research is encouraging regarding vertical plane location, it does not mention anything about horizontal plane indoor location.

In sum, a lot of research has been done around indoor location using many of the available technologies (GPS, Wi-Fi, Bluetooth, ZigBee, RFID, barometric sensors, etc.) and analyzing different techniques (fingerprinting, triangulation, etc.). However, the majority of this solutions are broad and do not have a specific goal in mind. Researching about 911 Bluetooth indoor location for emergency systems still is a topic that have not been widely researched and has not been added as a Bluetooth user case by the Bluetooth community yet.

8. Conclusion and Future Development

Through this project we propose a first Bluetooth Indoor Location for Emergency Services prototype where all the elements, with the exception of the NG911 Network Module, has been specifically designed for the emergencies indoor location goal. For the design and development of the iBeacons; including their management system; the development of the application frontend and backend we used easily available hardware elements and open source tools. The system has been developed while its limitations were studied and its major problems were solved. Problems such as developing an Android App capable of operating with the NG911 Emergency Module as SIP client, deciding which deployment methodology to use, reaching unanimity and having a convention for determining a caller location and testing it or the manufacturing of the 100 iBeacons used for this project.

Although the goal of the project was to achieve accuracy levels of less than 10 meters, which we achieved with test results with accuracy levels of less than 1 meter, we believe that the improvement of the location estimation might be one of the key elements in future developments. Despite we reached very encouraging levels of accuracy, we believe that a more in deep analysis of the gathered data with special emphasis on studying other location algorithms would lead us to an improvement in our system.

Another interesting point which could have further development is the additional user information. Information such as the name, age, language or even the blood type of the caller might be useful information that we could send to the PSAP. Therefore, we would be able to recognize with accuracy where is an emergency and, for example, what is the name or preferred language of the caller.

For a long term future, we would also like to keep researching about the design and development of an advances database system which could be divided in two database modules: an external available database for emergency response services and an internal database. This division would allow us not only offer additional services for different parties but obtain a forensics tool for any kind of criminal or emergency investigation.

We visualize the future of this project as a complete NG911 Indoor Location secure platform. A platform which integrates the Bluetooth indoor location system with other positioning systems that use technologies such as Wifi or GPS.

Acronyms

ALI - Automatic Location Information

API - Application program interface

BCF - Border Control Function

Bluetooth LE - Bluetooth Low Energy

CMRS - Commercial Mobile Radio Service

CTIA - Cellular Telephone Industries Association

ECRF - Emergency Call Routing Function

ESInet - Emergency Services IP Network

ESRP - Emergency Services Routing Proxy

FCC - Federal Communications Commission

HCI - Host Controller Interface

IEEE - Institute of Electrical and Electronics Engineers

LoST - Location to Service Translation

NENA - National Emergency Number Association

PIDF-LO - Presence Information Data Format Location Object

PSAP - Public-Safety Answering Point

RSSI - Received Signal Strength Indicator

SIP - The Session Initiation Protocol

VoIP - Voice over IP

VoLTE - Voice over LTE

Bibliography

- [1] N. H. T. S. Administration, "National 911 Progress Report," 2014. [Online]. Available: <http://www.911.gov/pdf/National-911-Program-2014-ProfileDatabaseProgressReport-031315.pdf>. [Accessed 06 2016].
- [2] "FCC," TruePosition, 2013. [Online]. Available: http://transition.fcc.gov/bureaus/pshs/911/Phase%202/Workshop_11_2013/TruePosition%20FCC%20briefing%2011-18-2013.pdf. [Accessed 06 2016].
- [3] USA Today, [Online]. Available: <http://www.usatoday.com/story/news/2015/02/22/cellphone-911-lack-location-data/23570499/>. [Accessed 05 2016].
- [4] Bluetooth SIG, "Bluetooth, Specification of the Bluetooth System," [Online]. Available: <https://www.bluetooth.com/specifications/adopted-specifications>. [Accessed 02 2016].
- [5] "NENA," NENA, 2011. [Online]. Available: https://c.ymcdn.com/sites/www.nena.org/resource/resmgr/Standards/08-003_Detailed_Functional_a.pdf.
- [6] "Federal Communication Commission (FCC)," 2015. [Online]. Available: https://apps.fcc.gov/edocs_public/attachmatch/FCC-15-9A1.pdf. [Accessed 06 2016].
- [7] FCC, "Guidelines for Testing and Verifying the Accuracy of Wireless E911 Location Systems," 2000. [Online]. Available: https://transition.fcc.gov/Bureaus/Engineering_Technology/Documents/bulletins/oet71/oet71.pdf.
- [8] Particle (formerly Spark), "Particle Photon Docs," Spark, [Online]. Available: <https://docs.particle.io/datasheets/photon-datasheet/>. [Accessed 02 2016].
- [9] IIT, [Online]. Available: <https://appliedtech.iit.edu/rtc-lab/research-projects>. [Accessed 05 2016].
- [10] AltBeacon Bluetooth, [Online]. Available: https://altbeacon.github.io/android-beacon-library/distance_vs_time.html. [Accessed 05 2016].
- [11] G. Chakraborty, K. Naik and D. Chakraborty, "Analysis of the Bluetooth device discovery protocol," 2008. [Online]. Available: <http://www.shiratori.riec.tohoku.ac.jp/~deba/PAPER/Journal/WINET-onlineFinal.pdf>.
- [12] "http://transition.fcc.gov," CSRIC, 2013. [Online]. Available: http://transition.fcc.gov/bureaus/pshs/advisory/csric3/CSRIC_III_WG3_Report_March_%202013_ILTestBedReport.pdf. [Accessed 07 2016].
- [13] "TruePosition," [Online]. Available: <https://www.trueposition.com/>. [Accessed 07 2016].
- [14] J.-g. Park, B. Charrow, D. Curtis, J. Battat, E. Minkov, J. Hicks and S. Teller, "Growing an Organic Indoor Location System," 2010. [Online]. Available: <https://people.csail.mit.edu/jgpark/docs/PCC10.pdf>. [Accessed 07 2016].

- [15] A. Bekkelien, "Bluetooth Indoor Positioning," 2012. [Online]. Available: http://tam.unige.ch/assets/documents/masters/bekkelien/Bekkelien_Master_Thesis.pdf. [Accessed 07 2016].
- [16] E. Kim, "DeepBLE - Localized navigation using Low Energy," 2014. [Online]. Available: http://www.seas.upenn.edu/~cse400/CSE400_2013_2014/reports/18_report.pdf. [Accessed 07 2016].
- [17] H. Xia, X. Wang, Y. Qiao, J. Jian and Y. Chang, "Using Multiple Barometers to Detect the Floor Location of Smart Phones with Built-in Barometric Sensors for Indoor Positioning," 2015. [Online]. Available: <http://www.mdpi.com/1424-8220/15/4/7857>. [Accessed 07 2016].
- [18] Federal Communications Commision, 2013. [Online]. Available: <https://fccid.io/document.php?id=2106792>.
- [19] I. C. Society, "IEEE 802.11 Std. páginas 2689-2690," New York, 2012.
- [20] Federal Communications Commision, [Online]. Available: <https://www.fcc.gov/consumers/guides/911-wireless-services>. [Accessed 4 2016].
- [21] A. B. J. Henry Sinnreich, Internet Communications Using SIP: Delivering VoIP and Multimedia Services, 2001.
- [22] S. T. a. L. Hettick, "Network World," 2005. [Online]. Available: <http://www.networkworld.com/article/2315734/lan-wan/how-e-911-caller-locations-are-discovered.html>. [Accessed 6 2016].

Appendix A:

A1. AltBeacon Library

Below there is an explanation of the Bluetooth parsing that the AltBeacon Library uses.

For filtering a standard Bluetooth LE iBeacon we will use:

```
beaconManager.getBeaconParsers().add(new BeaconParser().setBeaconLayout("m:2-3=0215,i:4-19,i:20-21,i:22-23,p:24-24,d:25-25"));
```

This signifies that the beacon type will be decoded when an advertisement is found with 0x0215 in bytes 2-3, and a three-part identifier will be pulled out of bytes 4-19, bytes 20-21 and bytes 22-23, respectively. A signed power calibration value will be pulled out of byte 24, and a data field will be pulled out of byte 25.

m - matching byte sequence for this beacon type to parse (exactly one required)

s - ServiceUuid for this beacon type to parse (optional, only for Gatt-based beacons)

i - identifier (at least one required, multiple allowed)

p - power calibration field (exactly one required)

d - data field (optional, multiple allowed)

x - extra layout. Signifies that the layout is secondary to a primary layout with the same matching byte sequence (or ServiceUuid). Extra layouts do not require power or identifier fields and create Beacon objects without identifiers.

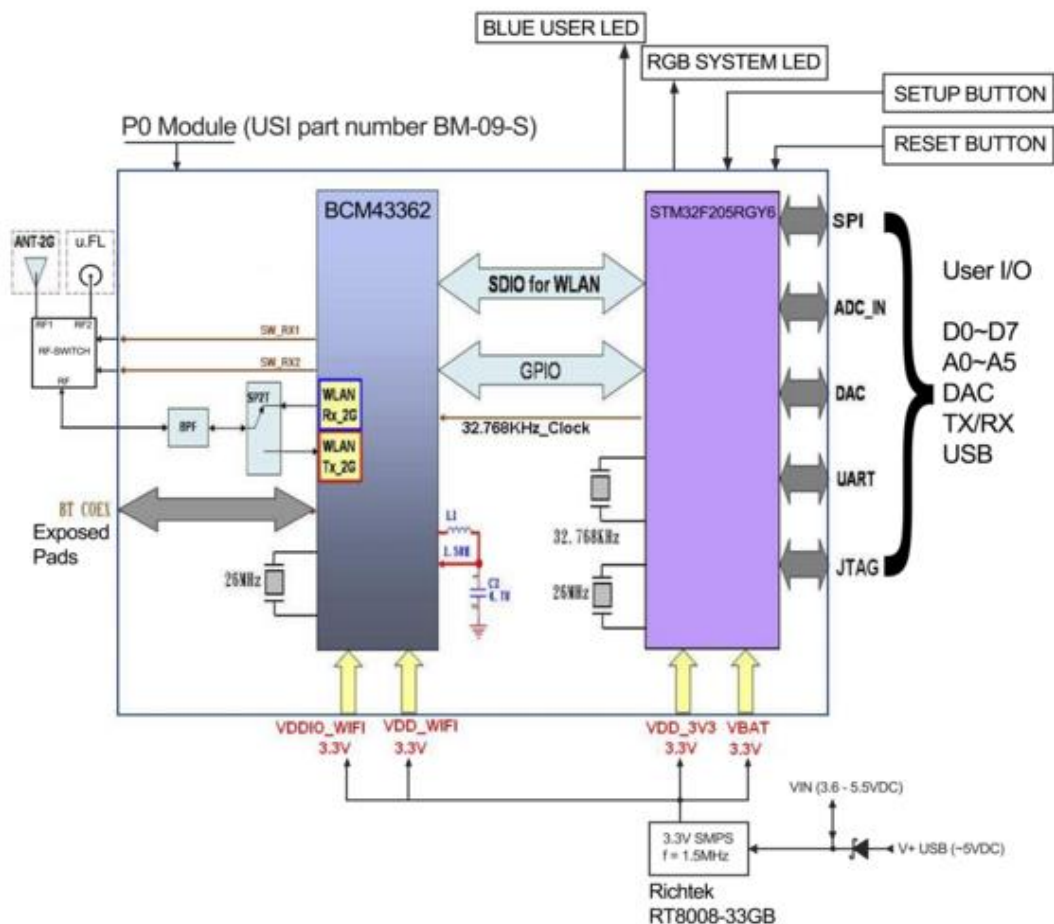
Appendix B:

B1. Nexus 5 Specifications

Screen	<ul style="list-style-type: none">• 4.95" 1920 x 1080 display (445 ppi)• Full HD IPS• Corning® Gorilla® Glass 3
Size	69.17 x 137.84 x 8.59mm
Weight	4.59 ounces (130g)
Cameras	<ul style="list-style-type: none">• 1.3 MP front facing• 8 MP rear facing with Optical Image Stabilization
Memory	<ul style="list-style-type: none">• 16GB or 32GB (actual formatted capacity will be less)• 2GB RAM
Processing	<ul style="list-style-type: none">• CPU: Qualcomm Snapdragon™ 800, 2.26GHz• GPU: Adreno 330, 450MHz
Sensors	<ul style="list-style-type: none">• GPS• Gyroscope• Accelerometer• Compass• Proximity/Ambient Light• Pressure• Hall Effect
Networks	2G/3G/4G LTE North America: GSM: 850/900/1800/1900 MHz CDMA: Band Class: 0/1/10 WCDMA: Bands: 1/2/4/5/6/8/19 LTE: Bands: 1/2/4/5/17/19/25/26/41 Rest of World: GSM: 850/900/1800/1900 MHz WCDMA: Bands: 1/2/4/5/6/8 LTE: Bands: 1/3/5/7/8/20
Wireless	<ul style="list-style-type: none">• Dual-band Wi-Fi (2.4G/5G) 802.11 a/b/g/n/ac• NFC (Android Beam)• Bluetooth 4.0 LE• Bluetooth Output power: 8.4dBm [6]
Ports & Connectors	<ul style="list-style-type: none">• microUSB• SlimPort™ enabled• 3.5mm stereo audio jack• Dual microphones• Ceramic Power & Volume buttons

Audio	<ul style="list-style-type: none"> Built-in speaker 3.5mm stereo audio connector
Battery	<ul style="list-style-type: none"> 2,300 mAh non-removable battery Standby time: up to 300 hours Talk time: up to 17 hours Internet use time: up to 8.5 hours on Wi-Fi; up to 7 hours on LTE Wireless Charging built-in
Supported Service	<ul style="list-style-type: none"> Unlocked: Use with any provider that offers service on Nexus 5 supported networks Compatible carriers include: T-Mobile, Sprint, AT&T
OS	Android 5.1 (Lollipop)

B2. Photon Particle Specifications



ABSOLUTE MAXIMUM RATINGS

Parameter	Symbol	Min	Typ	Max	Unit
Supply Input Voltage	V_{IN-MAX}			+6.5	V
Supply Output Current	$I_{IN-MAX-L}$			1	A
Supply Output Current	$I_{3V3-MAX-L}$			100	mA
Storage Temperature	T_{stg}	-40		+85	°C
Enable Voltage	V_{EN}			$V_{IN}+0.6$	V
ESD Susceptibility HBM (Human Body Mode)	V_{ESD}			2	kV

RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol	Min	Typ	Max	Unit
Supply Input Voltage	V_{IN}	+3.6		+5.5	V
Supply Input Voltage	V_{3V3}	+3.0	+3.3	+3.6	V
Supply Output Voltage	V_{IN}		+4.8		V
Supply Output Voltage	V_{3V3}		+3.3		V
Supply Input Voltage	V_{VBAT}	+1.65		+3.6	V
Supply Input Current (VBAT)	I_{VBAT}			19	uA
Operating Current (Wi-Fi on)	I_{IN-avg}		80	100	mA
Operating Current (Wi-Fi on)	I_{IN-pk}	235 ^[1]		430 ^[1]	mA
Operating Current (Wi-Fi on, w/powersave)	I_{IN-avg}		18	100 ^[2]	mA
Operating Current (Wi-Fi off)	I_{IN-avg}		30	40	mA
Sleep Current (5V @ VIN)	I_{Qs}		1	2	mA
Deep Sleep Current (5V @ VIN)	I_{Qds}		80	100	uA
Operating Temperature	T_{op}	-20		+60	°C
Humidity Range Non condensing, relative humidity				95	%

B3. Serial Bluetooth 4.0 BLE Module – iBeacon

The Bluetooth Module (HM-10, HM-11) is a highly integrated Bluetooth BLE V4.0 module, designed for low energy, short-range wireless communication in the 2.4 GHz ISM band. Combined with the Bluetooth low energy protocol stack from Texas Instruments.

Parameters

- (6) BT Version: Bluetooth Specification V4.0 BLE
- (7) Send and receive no bytes limit.
- (8) Working frequency: 2.4GHz ISM band
- (9) Modulation method: GFSK(Gaussian Frequency Shift Keying)
- (10) RF Power: -23dbm, -6dbm, 0dbm, 6dbm, can modify through AT Command AT+POWE.
- (11) Speed: Asynchronous: 6K Bytes
- (12) Synchronous: 6K Bytes
- (13) Security: Authentication and encryption
- (14) Service: Central & Peripheral UUID FFE0,FFE1
- (15) Power: +3.3VDC 50mA
- (16) Long range: Open space have 100 Meters with iphone4s
- (17) Power: In sleep mode 400uA~1.5mA, Active mode 8.5mA. Working temperature:-5 ~ +65 Centigrade Size: HM- 10 26.9mm x 13mm x 2.2 mm; HM-11 18*13.5*2.2mm

Appendix C:

C1. Access to Location Server

IP address: 64.131.109.50
server address: nead.bramsoft.com
server user: root
server password: *****

C2. Specifications of the Location Server

CPU Info

```
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 23
model name    : Intel(R) Core(TM)2 Quad CPU    Q9400  @ 2.66GHz
stepping      : 10
microcode     : 2571
cpu MHz       : 1998.000
cache size    : 3072 KB
physical id   : 0
siblings      : 4
core id       : 0
cpu cores     : 4
apicid        : 0
initial apicid : 0
fpu           : yes
fpu_exception : yes
cpuid level   : 13
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx lm constant
_tsc arch_perfmon pebs bts rep_good aperfmperf pni dtes64 monitor ds_cpl vmx
smx                      est tm2 ssse3 cx16 xtpr
pdcm sse4_1 xsave lahf_lm dts tpr_shadow vnmi flexprior
ity
bogomips      : 5303.06
clflush size   : 64
cache_alignment : 64
address sizes  : 36 bits physical, 48 bits virtual
power management:
---
processor      : 1
vendor_id     : GenuineIntel
```

```

cpu family    : 6
model         : 23
model name    : Intel(R) Core(TM)2 Quad CPU   Q9400  @ 2.66GHz
stepping      : 10
microcode     : 2571
cpu MHz       : 1998.000
cache size    : 3072 KB
physical id    : 0
siblings      : 4
core id       : 2
cpu cores     : 4
apicid        : 2
initial apicid : 2
fpu           : yes
fpu_exception : yes
cpuid level   : 13
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx lm constant
_tsc arch_perfmon pebs bts rep_good aperfmperf pni dtes64 monitor ds_cpl vmx
smx                                     est tm2 ssse3 cx16 xtpr
pdcm sse4_1 xsave lahf_lm dts tpr_shadow vnmi flexprior
ity
bogomips      : 5303.06
clflush size  : 64
cache_alignment : 64
address sizes  : 36 bits physical, 48 bits virtual
power management:
---
processor      : 2
vendor_id      : GenuineIntel
cpu family     : 6
model          : 23
model name     : Intel(R) Core(TM)2 Quad CPU   Q9400  @ 2.66GHz
stepping       : 10
microcode      : 2571
cpu MHz        : 1998.000
cache size     : 3072 KB
physical id    : 0
siblings       : 4
core id        : 1
cpu cores      : 4
apicid         : 1
initial apicid : 1
fpu            : yes
fpu_exception  : yes

```

```

cpuid level   : 13
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx lm constant
_tsc arch_perfmon pebs bts rep_good aperfmperf pni dtes64 monitor ds_cpl vmx
smx                                     est tm2 ssse3 cx16 xtpr
pdcmm sse4_1 xsave lahf_lm dts tpr_shadow vnmi flexprior
ity
bogomips     : 5303.06
clflush size : 64
cache_alignment : 64
address sizes : 36 bits physical, 48 bits virtual
power management:
---
processor     : 3
vendor_id     : GenuineIntel
cpu family    : 6
model         : 23
model name    : Intel(R) Core(TM)2 Quad CPU    Q9400  @ 2.66GHz
stepping      : 10
microcode     : 2571
cpu MHz       : 1998.000
cache size    : 3072 KB
physical id    : 0
siblings      : 4
core id       : 3
cpu cores     : 4
apicid        : 3
initial apicid : 3
fpu           : yes
fpu_exception : yes
cpuid level   : 13
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx lm constant
_tsc arch_perfmon pebs bts rep_good aperfmperf pni dtes64 monitor ds_cpl vmx
smx                                     est tm2 ssse3 cx16 xtpr
pdcmm sse4_1 xsave lahf_lm dts tpr_shadow vnmi flexprior
ity
bogomips     : 5303.06
clflush size : 64
cache_alignment : 64
address sizes : 36 bits physical, 48 bits virtual

```

File System

Filesystem

	Size	Used	Avail	Use%	Mounted on
/dev/mapper/vg_nead-lv_root	50G	2.8G	44G	6%	
tmpfs	3.9G	0	3.9G	0%	/dev/shm
/dev/sda1	477M	35M	417M	8%	/boot
/dev/mapper/vg_nead-lv_home	90G	56M	85G	1%	/home

Appendix D:

D1. NG911BL App Configuration

Set Up Instructions

For doing a call you need to do the following process:

- 1) Settings
- 2) Sip Account 1
- 3) Username: android2
- 4) Password: teamc255
- 5) Server or Proxy: 64.131.109.30 (This will lead you to the ESInet of the IIT RTC Lab)

Default Location Server: nead.bramsoft.com

However, the indoor location won't work if you don't use Bluetooth LE sensors.

For more information, contact agonza36@hawk.iit.edu

D2. ESInet Configuration

SIP Responses

	Description	Examples
1xx	Informational – Request received, continuing to process request.	180 Ringing 181 Call is Being Forwarded
2xx	Success – Action was successfully received, understood and accepted.	200 OK
3xx	Redirection – Further action needs to be taken in order to complete the request.	300 Multiple Choices 302 Moved Temporarily
4xx	Client Error – Request contains bad syntax or cannot be fulfilled at this server.	401 Unauthorized 408 Request Timeout
5xx	Server Error – Server failed to fulfill an apparently valid request.	503 Service Unavailable 505 Version Not Supported
6xx	Global Failure – Request is invalid at any server.	600 Busy Everywhere 603 Decline

Table 9.SIP Response Codes

Events connecting to the IIT ESInet:

Below we wrote down the more typical network events and procedures to solve them used during the development process.

- 1) If the following error appears in a SIP communication after sending a SIP INVITE to the BCF of the ESInet (Emergency Services IP Network):

```
#49 --> 03-24 12:47:49

SIP/2.0 100 Trying
Via: SIP/2.0/UDP
10.0.8.1:34438;received=104.194.122.23;branch=z9hG4bK85040;rport=46167
To: urn:service:sos
From: <sip:caller2@64.131.109.30>;tag=z9hG4bK99874964
Call-ID: 802004778484@10.0.8.1
CSeq: 1 INVITE

#50 --> 03-24 12:47:49

SIP/2.0 500 Internal Policy Error
Via: SIP/2.0/UDP
10.0.8.1:34438;received=104.194.122.23;branch=z9hG4bK85040;rport=46167
To: <urn:service:sos>;tag=b1CWqgEcfaZgZgfsL3oaLg
From: <sip:caller2@64.131.109.30>;tag=z9hG4bK99874964
Call-ID: 802004778484@10.0.8.1
CSeq: 1 INVITE
Date: Thu, 24 Mar 2016 16:39:21 GMT
Server: Columbia-SIP-Server 1.24 for NG9-1-1 POC (Build May 1 2011)
Content-Length: 0
```

We will need to check the authentication process and there we might find the problem. First we will check the Access-List on the SIP Source Group to be sure the call is being allowed in. Later we will create another non used username, in the SBC, and reboot the systems. After that, if the SIP INVITE message goes through the issue was caused due to an authentication error. If we still have some issues we will have to try other solution.

Also, the SIPCallTaker, the software used for retrieving the SIP calls in the call dispatcher, seems to have an unexpected behavior when the call has not been released previously. In other words, if a caller makes a call to the call dispatcher in the PSAP, this one will have to respond or release the call once it has finished. If he doesn't do it the user won't be able to call again to the PSAP.

To solve this issue we might have to close and open again the SIPCallTaker application.

- 2) If the authentication is satisfactory done and in the connection to the PSAP we don't see any package, the ESRP might be routing the SIP INVITE to the wrong PSAP.

For connecting to our PSAP or another PSAP we will have to access to the ESRP and go to the folder `ng911/sipd/perlmad/lost.pm` change the pointer to the desired PSAP.

If after that we still can't reach the RTC Lab PSAP, we might consider rebooting the SIPD (SIP proxy and location server), ESRP and the PSAP of the ESI-net. For doing it, we will have to access to the following devices and run the scripts:

- In the SIPD we will go to the folder `ng911/sipd/` and run `./sipd.sh`
- In the ESRP we will go to the folder `ng911/sipd/` and run `./sipd.sh`
- In the Columbia PSAP we will go to the folder `ng911/sipd/` and run `./sipd.sh` and then, we will access to the folder `ng911/psapd/` and run `./psap.sh`

Solved the ESI-net problem we will be able to send the Invite and establish the call with the dispatcher.

E1. IBeacon Deployment Locations Map

Basement

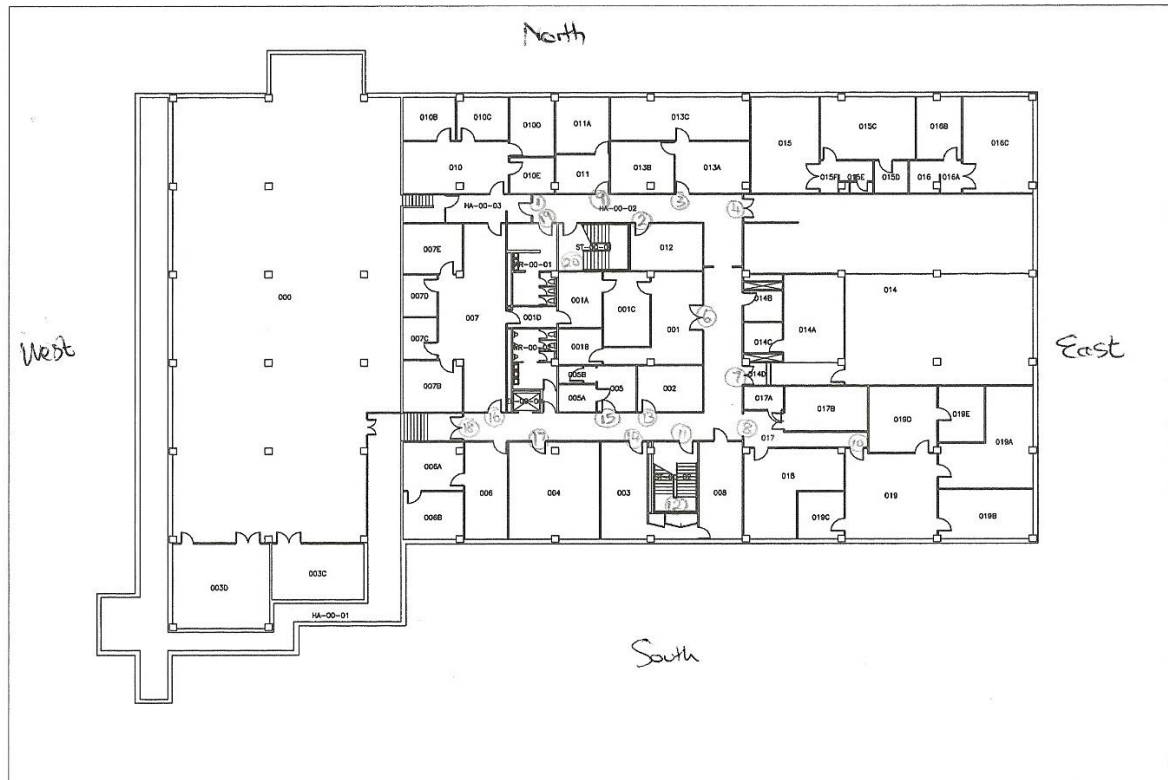


Figure 29. Stuart Building iBeacons deployment locations Basement

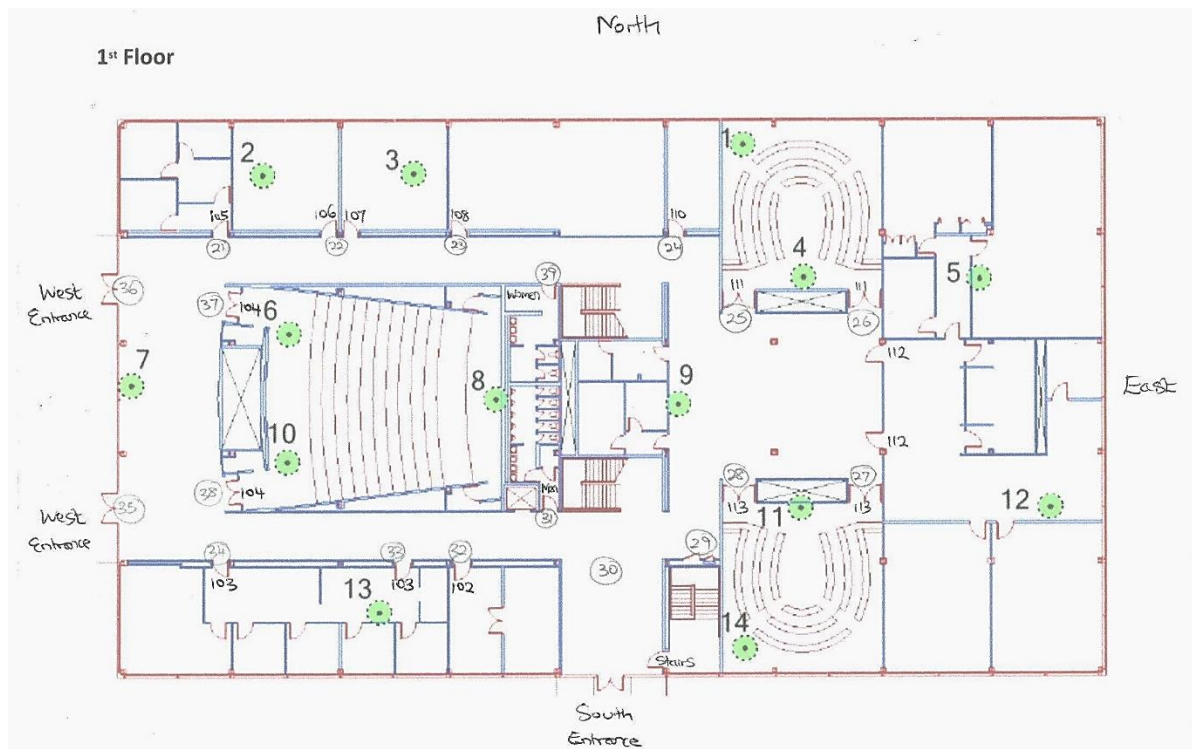


Figure 30. Stuart Building iBeacons deployment locations First Floor

Appendix F:

F1. SQL Scripts

Maximum Power Algorithm

```
SELECT t.TestLocationNum, minor, rssi FROM APDB.BluetoothTestVals t
inner Join (SELECT TestLocationNum, max(rssi) as maxrssi FROM
APDB.BluetoothTestVals
group by TestLocationNum) t2
on t.TestLocationNum=t2.TestLocationNum and t.rssi=t2.maxrssi
order by t.TestLocationNum;
```

Maximum Power Estimated Indoor Locations

```
SELECT t.TestLocationNum, minor, rssi, t3.Acronym, t3.Floor, t3.RoomType,
t3.RoomNum, t3.NearestWall FROM APDB.BluetoothTestVals t
inner Join (SELECT TestLocationNum, max(rssi) as maxrssi FROM
APDB.BluetoothTestVals
group by TestLocationNum) t2
on t.TestLocationNum=t2.TestLocationNum and t.rssi=t2.maxrssi
inner Join( Select * FROM LocationsMappingTests ) t3
on t.TestLocationNum=t3.TestLocationNum
order by t.TestLocationNum;
```

Maximum Average Power Algorithm Script

```
Select t.TestLocationNum, t.Minor, t.avg_ From (SELECT TestLocationNum, minor,
avg(power(10,rssi/10)*power(10,6)) as avg_
FROM BluetoothTestVals
group by TestLocationNum, minor
order by TestLocationNum, avg_ desc) t
GROUP BY t.TestLocationNum;
```

