ShopAssist - A Unified, Interactive, Location-Aware System for Shopping

Bruno Rodrigues Timóteo Lopes

Dissertation submitted to obtain the Master Degree in Information Systems and Computer Engineering

Supervisor: Prof. Ricardo Jorge Feliciano Lopes Pereira

Examination Committee

Chairperson: Prof. Mário Rui Fonseca dos Santos Gomes
Supervisor: Prof. Ricardo Jorge Feliciano Lopes Pereira
Member of the Committee: Prof. Miguel Filipe Leitão Pardal

May 2016
Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisor Prof. Ricardo Lopes Pereira for his guidance and endless support, regardless of the time of day. His constructive suggestions and feedback were absolutely invaluable and determinant for the accomplishment of the work present in the thesis.

Not less important, I would also like to thank my parents for the unconditional support, encouragement and love I have gotten over the years. Undoubtedly I could not have go so far without you. A special thank to my father, for his wise advices throughout my life.

A grateful acknowledgement to the Communication Networks and Mobility group at INESC-ID Lisboa, Instituto Superior Técnico.

To my closer friends that accompanied me throughout this magnificent academic journey, I express my sincere gratitude.

Thank you.

Bruno Timóteo Lopes
Lisbon, May 2016
to my parents.

I owe everything to you.
Abstract

Being able to locate and identify customers in shopping malls triggers the possibility of having a multitude of new features. On one hand, customers can appreciate features such as indoor navigation, product localization and enjoy location-based offers. On the other hand, business owners also benefit from it through the produced metrics and traffic pattern analysis, increased revenue and customer engagement.

In this document we survey the available technologies that make this possible and propose a system which is capable of providing and making use of indoor localization. A prototype of this system was implemented and evaluated. Our prototype relies on small Bluetooth devices, called Beacons, to provide indoor location, taking advantage of the fact that nowadays most people carry a smartphone.

Different indoor-location aware applications for commerce exist, but these are purposely built for a specific store. We, on the other hand, developed a generic mobile application that dynamically locates users indoors, regardless of the store, enabling several stores to make use of it. Moreover, a Content Management System was developed to abstract Business Owners from the complexity of deploying an indoor localization system. We also provide an interface for controlling the media content to be displayed on the Digital Signage, enabling targeted content to be aimed at different shoppers.

Towards validating our prototype, we performed accuracy tests in different scenarios to evaluate the accurateness of the system. Likewise, load tests were carried out to assess the performance of our server, as well as user tests to measure the usefulness of the system.

**Keywords:** Context-awareness, Indoor positioning, Targeted content, Bluetooth Low-energy, Digital Signage, Mobile consumer engagement
Resumo

Ser capaz de localizar e identificar consumidores em estabelecimentos comerciais despoleta a possibilidade de usufruir de uma multiplicidade de novas funcionalidades. Por um lado, os consumidores podem desfrutar de funcionalidades como a navegação em ambientes interiores, localização de produtos, assim como aproveitar ofertas baseadas num determinado contexto. Por outro lado, os proprietários de negócios também beneficiarão disso através das métricas produzidas e análises de padrões de tráfego, aumento de receita e envolvimento do consumidor.

Neste documento examinamos as tecnologias disponíveis que possibilitam isso e apresentamos um sistema capaz de disponibilizar e fazer uso de localização dentro de edifícios. Um protótipo deste sistema foi implementado e avaliado. O nosso protótipo baseia-se em pequenos dispositivos Bluetooth, designados de *Beacons*, para oferecer localização dentro de edifícios, tirando partido do facto de que hoje em dia a maioria das pessoas transportam um *smartphone* consigo.

Existem vários sistemas de localização para estabelecimentos comerciais, mas estes são altamente individuais, exclusivamente desenhados para uma loja específica. Nós, por outro lado, desenvolvemos uma aplicação móvel genérica capaz de localizar dinamicamente o utilizador dentro de edifícios, independentemente da loja, permitindo que várias lojas façam uso da mesma aplicação. Para além disso, um Sistema de Gestão de Conteúdos foi desenvolvido para abstrair os lojistas da complexidade de implantar um sistema de localização dentro de edifícios. Também disponibilizamos uma interface para gerir o conteúdo multimédia a ser mostrado na sinalização digital, permitindo que esse conteúdo seja dirigido para diferentes utilizadores.

No sentido de validar o nosso protótipo, realizámos testes de precisão em diferentes cenários para avaliar a exactidão do sistema. Do mesmo modo, testes de carga foram efectuados para avaliar o desempenho do nosso servidor, assim como testes com utilizadores para medir a utilidade do sistema.

**Palavras-chave:** Posicionamento interior, Conteúdo dirigido, Bluetooth Low-energy, Sinalização Digital, Context-awareness, Envolvimento do cliente móvel
# Contents

Acknowledgments iii  
Abstract vi  
Resumo vii  
List of Figures xiv  
List of Tables xv  
Acronyms xvii  

## 1 Introduction  
1.1 Motivation .................................................. 1  
1.2 Proposed Solution ........................................... 2  
1.3 Thesis Contribution .......................................... 3  
1.4 Outline ....................................................... 4  

## 2 Related Work  
2.1 Technologies .................................................. 5  
2.1.1 Global Positioning System (GPS) .......................... 5  
2.1.2 Global System for Mobile Communications (GSM) .......... 6  
2.1.3 Wireless Local Area Network (WLAN) ....................... 6  
2.1.4 Bluetooth .................................................. 7  
2.1.5 Visual Recognition .......................................... 8  
2.1.6 Ultra-Wide Band (UWB) ..................................... 8  
2.1.7 Radio-Frequency Identification (RFID) ....................... 9  
2.1.8 Infrared ..................................................... 9  
2.1.9 Near-Field-Communication (NFC) .......................... 10  
2.1.10 Inertial Navigation System (INS) .......................... 11  
2.1.11 Visible light communication (VLC) ......................... 11  
2.2 Positioning Techniques ....................................... 12  
2.2.1 Trilateration and Triangulation ............................. 12
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2</td>
<td>Fingerprinting</td>
<td>14</td>
</tr>
<tr>
<td>2.2.3</td>
<td>Dead-Reckoning</td>
<td>14</td>
</tr>
<tr>
<td>2.2.4</td>
<td>Proximity-based</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>Solutions</td>
<td>15</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Estimote</td>
<td>15</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Apple iBeacon</td>
<td>16</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Galeries LaFayette Haussmann</td>
<td>17</td>
</tr>
<tr>
<td>2.3.4</td>
<td>Low-cost third party beacons</td>
<td>17</td>
</tr>
<tr>
<td>2.3.5</td>
<td>Google Indoor</td>
<td>18</td>
</tr>
<tr>
<td>2.4</td>
<td>Critical Analysis</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>Architecture</td>
<td>22</td>
</tr>
<tr>
<td>3.1</td>
<td>Requirements</td>
<td>22</td>
</tr>
<tr>
<td>3.2</td>
<td>System Modules</td>
<td>23</td>
</tr>
<tr>
<td>3.2.1</td>
<td>ShopAssist Server</td>
<td>23</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Client</td>
<td>26</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Digital Signage</td>
<td>26</td>
</tr>
<tr>
<td>3.3</td>
<td>Privacy &amp; Security concerns</td>
<td>27</td>
</tr>
<tr>
<td>3.4</td>
<td>Chapter Summary</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>Implementation</td>
<td>28</td>
</tr>
<tr>
<td>4.1</td>
<td>Adopted Technologies</td>
<td>28</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Platforms</td>
<td>28</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Implementation choices</td>
<td>30</td>
</tr>
<tr>
<td>4.2</td>
<td>ShopAssist Server</td>
<td>33</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Content Management System (CMS)</td>
<td>35</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Database</td>
<td>41</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Positioning Engine</td>
<td>43</td>
</tr>
<tr>
<td>4.3</td>
<td>Client</td>
<td>44</td>
</tr>
<tr>
<td>4.4</td>
<td>Digital Signage</td>
<td>49</td>
</tr>
<tr>
<td>4.5</td>
<td>Chapter Summary</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Evaluation</td>
<td>52</td>
</tr>
<tr>
<td>5.1</td>
<td>Tests Objectives</td>
<td>52</td>
</tr>
<tr>
<td>5.2</td>
<td>Tests Scenarios</td>
<td>52</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Room</td>
<td>53</td>
</tr>
<tr>
<td>5.2.2</td>
<td>University Office</td>
<td>54</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Used equipment and Software</td>
<td>56</td>
</tr>
<tr>
<td>5.3</td>
<td>Test Results and Discussion</td>
<td>57</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Server Load Tests</td>
<td>58</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>5.3.2 Accuracy Tests</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>5.3.3 ShopAssist Mobile App Resource Usage</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>5.3.4 User Tests</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>5.4 Costs</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>5.5 Chapter Summary</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>6.1 Summary</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>6.2 Future Work</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>A Appendix A</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>A.1 Questionnaire</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>Bibliography</td>
<td>78</td>
<td></td>
</tr>
</tbody>
</table>
# List of Figures

1.1 Simplified system’s interaction .................................................. 3

2.1 Positioning techniques based on geometric information .................. 13

2.2 Estimote Beacon’s Hardware .................................................... 16

2.3 iBeacon packet. Based in [1] .................................................... 17

2.4 Galeries LaFayette app in action ................................................. 17

3.1 Block Diagram of the system ..................................................... 24

3.2 Positioning Engine input/output example – matching a Beacon to a Location .................................................. 25

4.1 Example of a more complex space mapped using Estimote Indoor SDK .................................................. 30

4.2 Mapping the same space using Estimote Indoor iOS application versus Estimote Indoor SDK .................................................. 31

4.3 Clarification of Estimote SDK input Fields ..................................... 34

4.4 Clarification on how to measure the space orientation. $\beta$ denotes the angle between the magnetic north (represented by the green arrow) and the location. .................................................. 34

4.5 Step 2 - Adding Location Boundary Points .................................... 36

4.6 Step 3 - Adding Beacons to a Location ........................................ 36

4.7 Step 3 - Adding Windows to a Location ........................................ 37

4.8 Location Manager - Modify Location feature .................................. 37

4.9 Manage Beacons page - lists existing beacons, allowing the user to delete or add new ones. .................................................. 38

4.10 Step 2 - Adding media content to a Location ................................ 38

4.11 Step 3 - Editing user groups ..................................................... 39

4.12 Overview of the Content Manager .............................................. 39

4.13 CMS Position Log feature – Allows filtering by user and by time interval through a date/time picker widget .................................................. 40

4.14 CMS Product Management feature ............................................. 41

4.15 UML Table Relationship Diagram of the Database ......................... 42

4.16 Communication methods between Client, Server and Digital Signage .................................................. 44

4.17 Screenshot of the mainscreen of the ShopAssist Mobile app .................. 45

4.18 Sequence Diagram resulting from the ShopAssist iOS application .......................... 47
4.19 Diagram representing the interactions between the AppDelegate, Views (Storyboard) and the corresponding ViewControllers .................................................. 48
4.20 Digital Signage running with a Raspberry Pi 3, with a beacon above. ............. 50

5.1 Illustration of the performed tests on a room ........................................... 53
5.2 ShopAssist app running on iPhone 6s within a University Office .................. 54
5.3 Two mapped offices for user testing scenario ........................................... 55
5.4 Network usage of the Server .................................................................. 58
5.5 CPU Usage of the Server ........................................................................ 59
5.6 Memory usage of the Server .................................................................... 59
5.7 RetrieveLocation WS Load Test - Response time (ms) ............................... 59
5.8 Accuracy tests on a Room ....................................................................... 60
5.9 Screenshot of the mobile application during accuracy tests being performed on a room, in proximity to beacons ......................................................... 62
5.10 Result of the accuracy tests in office ...................................................... 63
5.11 CPU Usage of the ShopAssist iOS application ....................................... 64
5.12 Network Usage of the ShopAssist iOS application ................................. 65
5.13 Energy Impact of the ShopAssist iOS application ................................... 65
5.14 RAM Usage of the ShopAssist iOS application ...................................... 66
5.15 Overview of the store ............................................................................ 68
List of Tables

2.1 Comparison of localization technologies ......................................................... 19
3.1 CMSs comparison .............................................................................................. 25
5.1 Server Hardware ................................................................................................. 56
5.2 Server Software .................................................................................................. 56
5.3 Software versions of the used equipment .......................................................... 57
5.4 iPad Mini 2 specifications ................................................................................. 57
5.5 iPhone 6s specifications ...................................................................................... 57
5.6 Accuracy test results in a room along Y-axis .................................................... 60
5.7 Accuracy test results in a room along X-axis .................................................... 61
5.8 Calculated probability parameters for room measurements ............................. 61
5.9 Accuracy test results of measurements taken on a room in proximity to beacons .......................................................................................................................... 62
5.10 Accuracy test results in an office using an iPad Mini 2 .................................... 62
5.11 Accuracy test results in an office using an iPhone 6s ....................................... 63
5.12 Calculated probability parameters for office measurements made with an iPad Mini 2 ................................................................................................................................. 63
5.13 Calculated probability parameters for office measurements made with an iPhone 6s ................................................................................................................................. 63
5.14 User tests results of task time-of-accomplishment ......................................... 66
5.15 User classification of the system in a 7-point scale, where the best scores are the lowest values .................................................................................................................. 67
5.16 Effective prototype cost .................................................................................... 68
5.17 Cost estimation for an approximately 230 sq. meter store ............................. 69
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-GPS</td>
<td>Assisted-GPS</td>
</tr>
<tr>
<td>AoA</td>
<td>Angle of Arrival</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BLE</td>
<td>Bluetooth Low-Energy</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>INS</td>
<td>Inertial Navigation System</td>
</tr>
<tr>
<td>IR</td>
<td>Infrared</td>
</tr>
<tr>
<td>LCD</td>
<td>Liquid-crystal display</td>
</tr>
<tr>
<td>LED</td>
<td>Light-emitting diode</td>
</tr>
<tr>
<td>LoS</td>
<td>Line-of-Sight</td>
</tr>
<tr>
<td>LTE</td>
<td>Long Term Evolution</td>
</tr>
<tr>
<td>NFC</td>
<td>Near-Field-Communication</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>QR</td>
<td>Quick Response</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency Identification</td>
</tr>
<tr>
<td>RSSI</td>
<td>Received Signal Strength Indicator</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>TDoA</td>
<td>Time Difference of Arrival</td>
</tr>
<tr>
<td>ToA</td>
<td>Time of Arrival</td>
</tr>
<tr>
<td>TTFF</td>
<td>Time To First Fix</td>
</tr>
<tr>
<td>UWB</td>
<td>Ultra-Wide Band</td>
</tr>
<tr>
<td>VLC</td>
<td>Visible light communication</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

Nowadays, we spend most of our time indoors, either at home, at the office or shopping. Over the course of time, given the broad range of products available, large retail spaces have essentially turned into the primary "go-to" place to buy everything. In contrast, small traditional commerce is losing its faithful customers. In this type of proximity commerce, shop assistants usually know their customers and are able to perform a personalized marketing by recommending them the products they are likely to need or buy. The same is true for online commerce, where massive databases of consumers habits can be mined for recommendation and targeted promotions, as Amazon does with the recommendation algorithms [2]. However, the same does not apply for large brick and mortar stores, where thousands of individuals cross the entrance doors per day. And in today’s society where time is at a premium, customers are looking for an efficient shopping experience, avoiding wasting time searching for goods.

Retail spaces only have access to information about the customer at the time of payment, when he finally identifies himself and makes his presence known, just right before he leaves the store! All shopping done by the customer is considered blind, thus preventing targeted recommendations to the shopper. Large companies try to circumvent this by deploying loyalty programs, where customers are regularly contacted (by mail, email or SMS) about global or specifically tailored promotions. Nonetheless, time spent in the store, browsing products, is a wasted chance for personalized marketing.

Today, there are technologies that may provide a leap forward in this context. As almost every person carries a mobile phone, these technologies are available for companies to take advantage of. Furthermore, the applications of these technologies are not confined only to retail. They could either be deployed in museums, to deliver detailed information about certain pieces of art, as well as providing self-guided tours to visitors. Also, in restaurants and pubs, they may be used to book a table and make an order by selecting the products on the mobile phone.

If we know the customer position indoors, it is possible to enrich his shopping experience. As we spend most of our time indoors, either working, shopping or eating, and having in mind that GPS performs poorly inside buildings, a problem emerges. Thus, we come up with the following question:
How can we identify and locate customers inside stores, in a reliable and non-intrusive fashion?

Currently, there are technologies that aid to overcome the problem of indoor positioning, and several systems have already been developed in this scope. However, none of them is completely reliable, as all the underlying technologies have their own drawbacks, which will be better detailed in Chapter 2.

Retail spaces can benefit from indoor positioning technologies that enable them to trigger contextual actions designed by business owners. Pushing unique and highly individualized promotions to the customer and collecting detailed metrics and analytics (heatmaps\(^1\) and dwell time\(^2\) maps) are some examples that can improve a business's operation and revenue. Customers can also benefit from these technologies to enhance their shopping experience: indoor navigation will facilitate and expedite their visits to large stores; augmented reality or contextualized information display will provide easy access to more information about the products that interest them, such as photos, videos and reviews. Some large retail spaces have begun to implement indoor localization technologies and provide their own mobile apps so that their customers can experience some of these benefits. However, each application is designed for a specific context, for a particular place, requiring customers to install a specific app for each store, thereby leading to the following question:

In what way can we overcome the need of one application per place, aiming a deployment in multiple places?

### 1.2 Proposed Solution

Our solution aims to solve the previously stated problem of individuality, as currently each different space requires a different application. Therefore, our proposal is to:

- Develop a **generic context-aware mobile application** for shoppers’ smartphones, which dynamically locates customers indoors regardless of the store, enabling several stores to make use of it.

- Implement a **flexible CMS**, where business owners are given full-control over their locations (in this context, a store), and which abstracts them from the complexity of deploying a beacon infrastructure to provide indoor localization.

- Implement the **Digital Signage** component, where targeted content is displayed. Furthermore, we enable its media content management as well as the association of advertisement campaigns to user groups or specific users.

- Offer a centralized solution, providing business owners a way to increase revenue, by presenting them shopping analytics.

- Use Bluetooth Low-energy for indoor localization.

\(^1\) Graphical representations where values are represented by colours. 
\(^2\) The amount of time a person spends in the same place
Figure 1.1 presents an eagle-eyed view of the system’s interaction, which aims to provide the aforementioned features. In here we can observe the shoppers’ smartphone receiving location information, represented by $L_{ID}$, from Bluetooth devices (Beacons), fusing it with user identification ($U_{ID}$) and sending it to the cloud. This data is combined with personal information resulting in contextual and targeted content to be displayed on both the Digital Signage and the costumers’ smartphone.

To enable indoor navigation, the map of the building is required. It is intended that new locations can be integrated in our application by business owners themselves, who will supply the corresponding location data and desired media content.

### 1.3 Thesis Contribution

Taking into account the aforementioned problems and challenges, the main goal of this thesis was to develop a system in which the customer shopping experience is enriched, bringing added value for both sides (businesses and shoppers). Therefore, the contributions of this thesis are:

- A proposal of a system that facilitates the deployment of an indoor localization system in shopping malls, while at the same time providing customers an interactive shopping experience.

- An architecture that implements the aforementioned proposal, that can be deployed in new scenarios.

- A prototype to validate the produced architecture. This prototype was intended to be the most generic possible, by developing a unified context-aware mobile application for shoppers’ smartphones, a flexible CMS for business owners and the digital signage component.

- An experimental evaluation to validate our prototype, by performing user, load and accuracy tests over our system.
1.4 Outline

The remainder of this document is organized as follows.

- **Chapter 2** presents the state of the art divided into 4 sections: in section 2.1, we survey the state-of-the-art of indoor localization technologies; section 2.2 presents the techniques used with the different technologies to calculate user's position indoors; a set of today's commercial solutions for indoor location services appear described in subsection 2.3; a discussion and critical analysis of the researched technologies can be found in subsection 2.4.

- **Chapter 3** describes the system requirements and the architecture of our system.

- **Chapter 4** describes the implementation details of our prototype and the adopted technologies.

- **Chapter 5** describes the evaluation tests performed to evaluate our system and the corresponding results.

- **Chapter 6** summarizes the work developed and future work.
Chapter 2

Related Work

In this Section, we begin by presenting the principles and state-of-the-art for indoor positioning as our project is tightly coupled with this technology. In order to improve the customer shopping experience, it is necessary to know his position inside the buildings. Significant work has already been done in the area of indoor localization. A detailed overview of the technologies, methodologies and solutions already in use will be presented. Afterwards, an analysis will be presented towards finding out which is the technology that better suits our use-cases. The technologies will be presented sorted by descending order of range, starting from the higher-range and ending in the shorter-range.

2.1 Technologies

2.1.1 GPS

A very popular approach for localization is the GPS. This well-known technology is used worldwide, having been created by the US military to overcome the limitations of previous navigation systems. It provides a 3D-position (giving latitude, longitude and altitude), typically accurate within \( \sim 1 \) meter outdoors\(^1\) with today's technology. The positioning method used by GPS receivers to calculate their position is the Trilateration, which will be better described in Section 2.2.1.

GPS performs poorly on indoor environments, since their signals are unable to reach the inside of building with sufficient strength or quality to be used [3].

An experiment made in 2010 analysed the GPS signal availability using high-sensitivity GPS receivers [4]. They could only achieve, as best result, around 5-meter precision in wood buildings. The issue here is that many buildings are not made of wood, thereby increasing the error to unattractive levels. Furthermore, using GPS indoors suffers from low signal-to-noise ratios, multipath propagation (due to the surrounding buildings) and poor satellite constellation geometries (when the angle of the received signals differs in a small degree).

\(^1\)http://www.gps.gov/systems/gps/performance/accuracy/
Assisted-GPS (A-GPS) was proposed to surpass some of the limitations of the traditional GPS. A-GPS consists of using the mobile data network to aid position acquisition by combining the data from a location server and possibly weak GPS signals [5]. It provides a shorter Time To First Fix (TTFF) by predicting which satellites are visible to the mobile device in a certain place. Although it outperforms traditional GPS, by shortening the TTFF from a cold start (i.e., without any knowledge about visible GPS constellations) and by providing superior accuracy [6], the usage of A-GPS is still a not very reliable way of indoor localization [7].

The main drawback of this technology is that it requires an unobstructed Line-of-Sight to four or more satellites - three for position acquisition, and the 4th for clock synchronization.

### 2.1.2 GSM

GSM is the most widespread cellular-based network, typically operating in the 900 MHz and 1800 MHz frequency bands, and can also be used to locate people.

Mobile networks can provide an approximate location of the user by knowing which cell the mobile phone is connected to, as each cell has its own unique identifier (Cell-ID) and known absolute position. The density of cells is higher in urban areas (providing better accuracy), whereas in rural areas the density is much lower [8].

Otsason et al. implemented and tested an indoor localization system where GSM signals were used in order to calculate indoor position [9]. The key idea in this system is the usage of a wide range of signal strength fingerprints, i.e., in addition to the six strongest GSM signals, the weakest ones were also used. This higher dimensionality brought by using weak GSM signals (which are not strong enough to provide communication) significantly improves location accuracy. Hence, using up to 29 GSM signals, a median accuracy of 5 meters was achieved.

### 2.1.3 WLAN

Wireless Local Area Network, commonly know as WiFi, is a radio communication system operating in the 2.4 GHz and 5 GHZ frequency bands that connects two or more devices providing data exchange services. It has been continuously growing, having the major strength of being deployed almost everywhere in which people live and work, although it was never designed bearing location in mind.

In 2000, a system named RADAR was presented, in which WLAN was used for user localization and tracking indoors [10]. 3 Access Points (APs) were deployed in one floor of a building. Then, a learning procedure took place, where many measurements of the pair \( \text{coordinate,Received Signal Strength Indicator (RSSI)} \) were recorded on a database.
After this, to locate a user the system matches the measured RSSI with the pre-recorded RSSI values on a database. This positioning method, called Fingerprinting, will be better described in section 2.2.2. On their experimental results a median accuracy of 3 meters was achieved, although this result is affected by the orientation of the user and number of recorded samples.

Regarding WLAN in general: as smartphones are not standardized, each phone measures WiFi RSSI in a different way, limiting the use of this technology for localization purposes [11].

2.1.4 Bluetooth

Bluetooth technology was initially created in 1994 as a wireless alternative to RS-232 data cables, also know as Serial Port, allowing data exchange over short distances.

Many attempts of using Bluetooth for indoor positioning took place. One of those systems is called TOPAZ [12]. This hybrid system uses a combination of both Bluetooth and Infrared (IR) technologies to reach an improved precision and overcome IRs limitation: as IR doesn’t cross walls, it provides a room-confined positioning.

A room accuracy of about 2 meters was achieved. The main drawback of this system is the positioning latency, which is around 10 to 30 seconds due to the prototype and algorithms’ performance.

Later on, a marketing system using Bluetooth Low-Energy (BLE) was implemented [13], where push notifications were sent to the users upon their approach to the entry of the Book Store. Also, in-store advertising was provided to the user, according to his indoor location. Analytics were captured in order to find out if the system was effective, if it was driving more people to enter the book store than usual, and thereby increasing sales. A limitation of this system is the requirement of installation of the application by the user, so he can receive the promotions on the mobile phone.

Another usage of BLE is made by Corbacho Salas [14], that studied how BLE could be used as an Indoor Positioning System, and which factors may decrease the precision. After an analysis of several Indoor Positioning techniques, the trilateration method provided the best precision. With beacons placed on the walls of a room, and after a calibration phase, an average accuracy of 1.5 meters has been achieved.

The main reported disadvantage concerning the use of Bluetooth for Indoor Positioning is related to the need of extra infrastructure investment, as Bluetooth devices are usually not installed on buildings, although nearly all mobile devices have it [15].

The wide proliferation of these chipsets fostered the development of a new specification: BLE, the 4th version of the Bluetooth Protocol. Marketed as Bluetooth Smart, it provides low power consumption along with higher bit rates, compared to previous versions2. Its deployment began in 2011 with the Apple iPhone 4S [16]. This new version eases and increases localization accuracy by including the default RSSI value in the advertisement packet. This value is measured at one-meter distance and programmed into the beacon during manufacturing, as it may vary according to the chipset vendor [17].

2http://www.bluetooth.com/Pages/Fast-Facts.aspx
More recently, Bluetooth versions 4.1 and 4.2 emerged in 2013 and 2014, respectively. The Bluetooth 4.1 update is solely a software enhancement, having three major improvements: its extension to cope with the Internet of Things[18] (by providing the foundations for IP-based connections), enhanced usability for customers (new automatic reconnection policies) as well as a better cooperation between Bluetooth and Long Term Evolution (LTE) devices.

Bluetooth 4.2 is an hardware upgrade, although some features might be implemented on older Bluetooth chipsets as firmware updates. Mainly, it features improvements in transfer speeds and in the scope of the Internet of Things by laying the support for IPv6 within the Bluetooth Stack.

2.1.5 Visual Recognition

Visual recognition consists of analysing pictures taken by video cameras for further identification of a person’s face. However, it requires high computational power to process a large amount of images, making it not a very attractive method.

The major advantage reported is the provided comfort to the user, as there’s no need to carry any tracking device with them [19].

EasyLiving is a system built for position sensing and identification of people [20]. It uses two cameras on a room for motion detection and person identification, maintaining a record of one’s location history. Although this might look promising, it requires a considerable processing power and also the accuracy cannot be assured, due to the variations in the environment.

Focken et al. explored the use of multiple cameras to identify a person, along with a dedicated computer running an algorithm to process the images and compute their precise location [21]. In this case, the algorithm removes the background from the actual image, resulting in an image of individuals or moving objects. However, the problem here is related with the correct background subtraction, in order to avoid false-alarms produced by light changes.

2.1.6 UWB

Ultra-Wide Band provides short-range but high-bandwidth wireless communications. It is designed for low-energy consumption, transmitting information over a large portion of the radio spectrum.

UWB tags present the advantage of consuming less power when compared, for example, to RFID tags [5]. Furthermore, they are almost immune to interference of other radio signals, due to the different signal type (low-powered) and wide spectrum used.

Gigl et al. implemented a localization system using UWB technology [22]. Radio transmissions were done in the frequencies between 3.1 and 10.6GHz, acting like a beacon system by sending ultra-short
pulses. After a calibration phase, the RSSI was used along with the trilateration technique to calculate the position. In the end, the achieved precision was of 1.5 meters for the 90th percentile of error.

Reported drawbacks are related with the interference caused by certain materials, such as metals and liquids. In addition, it is also necessary to carefully learn the best way to handle the received signal [22]: i.e., in UWB the majority of the signal arrives in a very short period, so if we continue integrating the energy after the last relevant part, only noise is added. Plus, on the other end (too short integration time), a problem still exists. Those are called the integration limits, an issue particularly relevant to this technology due to the low-energy density of UWB which makes the signal noise-like [23].

2.1.7 RFID

RFID is a wireless communication system composed of RFID tags and readers. The tags, containing an integrated circuit plus an antenna, can be either active or passive [5], and the accuracy of the system depends on the type of chosen tag. Active tags have a built-in battery and are capable of actively transmitting radio signals, namely their ID. They have an higher range compared to passive ones. In contrast, passive tags operate without a battery, thereby being cheaper. They reflect the received radio signal, adding some information by modulating the signal. The major strength of this technology is not requiring of Line-of-Sight (LoS) to operate [24].

LANDMARC is a system that attempts to use RFID tags and readers to locate objects inside buildings [25]. This system uses the so-called reference-tags, which act like a landmark, to help location calibration. The calculation of the position is done by comparing the signal strength of the reference tag and the tracking tag, followed by the application of the nearest neighbour method, which returns the nearest neighbour of a tracking tag.

Some problems emerged, namely the lack of signal strength indicators (only a scale of 8 power levels are given) and the long latency to detect an object, due to the scanning time and the interval for emitting two consecutive IDs from the same tag. The experiments made to locate a tracking tag reported a maximum error of 2 meters, which can be lowered by having an higher density of reference tags.

A few years later, another Indoor Positioning System using RFID tags was developed [26] to overcome some of the problems faced by LANDMARC. Unlike LANDMARC where the nearest-neighbour algorithm is used, this new proposed system uses an adapted Bayesian-based algorithm. This better-performing filtering algorithm removes the abnormal signal data caused by multipath propagation effect, decreasing the error from 1 m (LANDMARC) to 0.5 m, when considering the 60th-percentile precision. Nevertheless, this accuracy can decrease quickly as the measurement errors increase.

2.1.8 Infrared
IR is a short-range wireless communication system using light, commonly used in remote controls. Its signals cannot penetrate walls, unlike radio signals.

There are two types of IR sensors: active and passive sensors. Active sensors rely on the interruption, e.g. by a passing Human, of a signal constantly being transmitted between a sender and a receiver, thus making it more energy-consuming. On the other hand, passive sensors simply wait for a heat signal, i.e., IR thermal radiation emitted by a Human body (source of heat).

One of the first attempts to provide indoor location using IR was the Active Badge [27]. Taking place in 1992, the purpose was to build a system for localization of people inside an office. A network of IR sensors was deployed, and the correspondence between receivers and rooms was stored in a central server. Each person carried an Active Badge, acting like a beacon emitting a Unique Identifier every 15 seconds. Afterwards, the signals are captured by the network of IR-sensors spread around the building, and processed by a central server, thus identifying the room or corridor where the user was located.

The main problems faced by this approach are the requirement of an unobstructed Line-of-Sight, as well as the requirement of a high sensor density, due to the limited range of an IR signal transmission. Also, direct sunlight interferes with IR networks, in case of places with windows [10].

2.1.9 NFC

NFC is a low-cost, modern and very short-range wireless communication technology3, which supports bidirectional exchange of data among two NFC devices a few centimetres apart. NFC is based on RFID, sharing the same frequency band (13.56 MHz), and operates in 3 modes: initiator (acting like a reader), target (acting as an host), or peer-to-peer (enabling bidirectional exchange of data) — the feature that differentiates NFC from typical RFID [28].

Ozdenizci et al. developed an indoor navigation system using passive NFC tags [29]. It consists of using a mobile device with an NFC reader, and touching the NFC tags spread inside the buildings. Each time the user approaches the NFC enabled mobile phone to a NFC tag placed on the wall, the installed application recognizes the location. Hence, in order to navigate from point A to point B inside a building, it is necessary to be repeatedly touching the NFC tags, so that the mobile application recognizes the location, thus calculating the route.

One year later, Hammadi et al. proposed a more comprehensive indoor localization system using NFC technology along with Quick Response (QR) codes [30], for the non-capable NFC smartphones. It features a mobile application with indoor map guidance.

The modus operandi of this system consists of scanning QR codes4 and NFC tags spread within the building (with known positions). The scan is done through the developed mobile application, providing information to the user about his current position and consequently diverse navigation features, e.g., the shortest path to a destination (a toilet, a store, car park).

3http://nfc-forum.org/what-is-nfc/about-the-technology
42D-barcode, converted into text when scanned with a camera
Both systems have drawbacks hereunder:

- Lack of integration with different outdoor navigation systems, as they require the map of the building on a special format;

- Absence of on-the-go position information, forcing constant action by the user.

Thanks to the NFC tags fixed position, the accuracy is guaranteed. In addition, the privacy of the user is assured as it eliminates the need of a central server to calculate its position.

2.1.10 INS

INSs use a variety of sensors, such as gyroscopes, accelerometers, tilt and pressure sensors, to infer a relative position. The new position is calculated from the previous determined position, fused it with the data coming from the sensors, thus making it a system subject to increased drift over time [31].

It is used nowadays in ships, submarines and aircraft, but can also be used as an indoor positioning system. It is considered a dead-reckoning positioning method (more detailed in section 2.2.3), as the new position is always deduced.

In 2006, an hybrid of WiFi and INS was implemented and tested by Evennou et al. [32] Their INS consists of a dual-axis accelerometer, a gyroscope and a pressure sensor (to detect movement from one floor to the other). One of the main problems they faced was the cumulative error coming from the Inertial Navigation System. It is also necessary to carry this set of sensors, preferably located on the users belt, to detect his movement. Integrating this system with WiFi signal strength measures from 4 APs, resulted in a average 3 meters error. Nevertheless, it was still only reliable for a short term, mainly due to the noisy measurements from the sensors.

After that, a particle filtering algorithm was implemented to discard any impossible move, e.g., crossing a wall. For this, a map of the building, in a bitmap format, is required. With the integration of all these methods, a 1-meter accuracy could be achieved. Although this might look like a promising location system, there was still the cumulative error problem. As in other technologies (GSM, Bluetooth, UWB), the precision can be improved by increasing node density (in this case APs).

2.1.11 VLC

VLC systems rely in visible light to transmit data. Among the three most-common types of lamps, incandescent, fluorescent and Light-emitting diode (LED), the latter is the most suitable to be used as a VLC. LEDs are semiconductor efficient light sources allowing to be switched at very high speeds. By modulating its intensity, the LED acts as a strobe emitting pulses in such high-frequencies that are undetectable for humans but detectable, for example, by smartphone cameras.
As an high-frequency is necessary to prevent visible flickering, this approach brings unnecessary processing overload to receiving devices. In 2015 a system to bypass this difficulty was proposed using a polarization-based\textsuperscript{5} VLC [33]. In this design, a polarizing film was attached to a portion of the light sources. Equally, a thin polarizing film was also placed over the front camera of the mobile phone. Afterwards, the smartphone runs an application which decodes the captured images.

One major reported drawback is related with deployment issues — although inexpensive, it is necessary to attach polarizing film to the front camera of the mobile phones. Furthermore, this system causes interferences in polarized devices, such as sunglasses and Liquid-crystal displays (LCDs). The VLC system emitting can be observed through the sunglasses. As LCDs also contains a polarizer, it is possible to notice the effects of the VLC on the screen.

VLC systems also suffer from multipath effects, interferences from other lamps and require a LoS to the light source. Reported accuracies are in the order of tens of centimetres [33] [34].

\section*{2.2 Positioning Techniques}

There are several methods to estimate a user's position indoors that can be used with the above technologies. The presented techniques can be divided into four main groups: related to user's closeness (Proximity-based approaches), based on exploring geometric information about a certain place (Trilateration and Triangulation), offline mapping (Fingerprinting) and also deduced reckoning approaches (Dead-Reckoning) [35].

Some of the technologies analysed in the previous section can be used with several of these methods, while others are only suited to a particular one.

\subsection*{2.2.1 Trilateration and Triangulation}

\textbf{Triangulation:} Using triangulation to calculate the position of a node requires measuring the angle of the arrived signals (called Angle of Arrival (AoA)).

The following step consists of using a basic geometric rule, after measuring the angle between the unknown point (location of the user) and at least two fixed points, whose distance between them is known (see Figure 2.1(b)). The disadvantages are the requirement of specific and complex hardware to measure the angles, and the precision degradation as the user moves away from the known positions, thus making it not very popular [5].

\textbf{Trilateration:} Trilateration, commonly called "triangulation", is a well-known positioning method that uses distances to calculate the position of a node.

It is based on the geometric relationship between nodes; by forming centred circumferences around three nodes, we came up with one point of intersection (see Figure 2.1(a)). Three nodes are the minimum required in a 2-dimensional space. In case of using more nodes to increase precision, it gets called Multilateration, hereinafter referred as Trilateration for simplification.

\textsuperscript{5}Polarization is the orientation of the oscillating light waves. Changing it is imperceptible for humans.
In trilateration, distances are usually derived from another measure, such as RSSI values. The problem lies in the fact that the distance measurements suffer from error and may oscillate greatly due to fluctuations in RSSI values, resulting on an intersecting area instead of a single point. In order to attenuate this error, more measurements can be taken and the results averaged [36].

![Diagram showing trilateration method example](image1)

(a) Trilateration method example

![Diagram showing angulation method example](image2)

(b) Angulation method example

Figure 2.1: Positioning techniques based on geometric information

### Determining distances:
In both Trilateration and Triangulation it is necessary to make measurements, either requiring additional hardware or not. As stated before, in triangulation technique the AoA is measured with specific hardware. Below, three methods for distance estimation are presented:

- **RSSI**: Using RSSI as an estimator of distance is common, as its value is inversely proportional to the square of the distance, and has the advantage of not requiring extra-hardware. A path-loss equation (Eq. 2.1), based on the *Friis transmission formula*, is used to convert the RSSI values to an estimated distance. The distance is dependent of the path-loss coefficient ($\alpha$), antenna gains ($G_r$ and $G_t$) and naturally on the RSSI ($P_{rx}$):

\[
\frac{P_{tx}}{P_{rx}} = G_r G_t \left( \frac{\lambda}{4\pi d} \right)^2 \Leftrightarrow d = \sqrt{\left( \frac{\lambda}{4\pi} \right)^2 \frac{P_{tx} G_r G_t}{P_{rx}}} \tag{2.1}
\]

Several different path-loss models have been proposed to provide better results, relying on the fact that some reference values are known beforehand. For instance, by adding other factors, bearing in mind the multipath effects and antenna pointing, the accuracy can be improved.

The major drawback of this method lies in the existence of multipath effects in indoor environments, as well as the oscillation of RSSI values, due to changes in the environment and interferences [37].

- **Time of Arrival (ToA)**:

Given that radio signal speed is approximately the speed of light, it becomes possible to determine distances based on the propagation time — Time of Arrival. Firstly, ToA implies synchronized
clocks among the sender and receiver to calculate the distance, when the propagation speed is known. Secondly, both sender and receiver need to know the time when the transmission of the pulse starts, to allow calculating the distance. For instance, this is the way GPS calculates its position.

The ToA method is not very adequate for indoor environments (at least with radio waves) due to relatively short distances and requirement of extremely high precision clocks [38]. In case of using sound waves, whose propagation speed is lower, other problems arise, such as the humidity or temperature, which affect the propagation speed.

- **Time Difference of Arrival (TDoA):**

  To overcome the limitations of the previous technique (ToA), the TDoA can be used without requiring explicit clock synchronization. This is done by telling the receiver when the pulse will be transmitted, and then transmit it through two mediums with different propagation speeds. Later on, the receiver measures the time difference of arrival of the same pulse through those mediums, and computes the distance. Although outperforming the RSSI approach, it requires two different types of senders and receivers per node.

### 2.2.2 Fingerprinting

Fingerprinting is a positioning method divided into two parts: the offline phase and the online phase [38]. First, on the offline phase, the area of the building is divided into small areas with an associated coordinate. Afterwards, for each coordinate, RSSI samples are measured and stored into a database. Then, in the online phase, when the user is at an unknown position, the measured RSSI is compared with the values stored on the database. This is done through a matching algorithm, which returns the coordinates that better fit the fingerprint. As an example, Bahl et al. [10] successfully implemented a system for locating users indoors using this technique. Their fingerprint was the RSSI of WLAN Access Points, resulting in a 3-meter average accuracy. This value is strongly affected by the recorded number of samples.

The main obstacle regarding fingerprinting is the lack of flexibility to couple with changes in topology and signal strength.

### 2.2.3 Dead-Reckoning

Dead-Reckoning is another possible approach to locate people indoors. It consists of having sensors attached to a person, such as accelerometers, gyroscopes, pressure sensors and magnetometers, plus having the benefit of not requiring a pre-installed infrastructure in buildings [39]. For absolute posi-
tioning, the initial position must be known; then, sensors are used to estimate the relative motion. The current position is based on the previous position fix, but calculated by including all data coming from the sensors. As this is a cumulative positioning method, the errors will accumulate on each new calculation.

INSs use dead-reckoning internally. Indeed, another positioning system might be used in order to periodically correct the drift of Dead-Reckoning approach, as done by Evennou et al. [32].

Systems like Locometric⁶ also use motion sensors in a similar fashion. Roomscan by Locometric is an application for smartphones to draw floor plans in seconds. This mobile application acquires the walls orientation by touching the surface of different walls with the smartphone, which causes motion to be sensed. This data is captured and the application is able to draw a floor map.

2.2.4 Proximity-based

This approach takes advantage of the limited range of some technologies, by detecting if a node is within the range of its anchor (e.g., a Bluetooth beacon). It is only suitable for short-range Radio Frequency devices (such as Bluetooth), as with longer range technologies (e.g., WLAN or GSM) the covered area would be too large for this purpose.

Even though it only provides coarse-grain information, the company Estimote⁷ successfully implemented a commercial product using this approach. When the user is close to a beacon (a BLE device), they push notifications with commercial offers to him.

Active Badge [27] is another example of a localization system based on proximity, due to the use of IR to restrict the localization boundaries to a room.

2.3 Solutions

Below will be briefly presented some commercial solutions available nowadays for indoor localization purposes.

2.3.1 Estimote

Estimote is a producer of BLE devices, called beacons (See Figure 2.2), enabling the creation of context-aware mobile applications using the provided Software Development Kit (SDK).

By using the proximity-based positioning technique 2.2.4, they are able to push notifications to near users providing some offers to them.

Or, for example in a museum, upon approaching a mobile device to a painting or sculpture, additional and detailed information can be pushed to that device (either a Tablet or a Mobile Phone). Utilizing 4 bea-

---

cons allows indoor location and distance measurement functionality on Estimote’s mobile application. This functionality uses the trilateration positioning technique, avoiding the need of a previous calibration from the users. Still, typical problems (e.g., multipath propagation, changes in the environment) continue to affect its performance.

The Estimote’s provided SDK is fully documented and allows developers to integrate context-awareness to their own applications, or to develop new usages for indoor location information. Estimote follows iBeacon specification for broadcasting BLE advertising packets, described below. Each Estimote beacon sends its advertising packet every 200ms. The advertising interval is programmable, as well as the broadcasting power.

2.3.2 Apple iBeacon

iBeacon is name of the Apple’s approach to the new generation of location-based services. Introduced in iOS 7, it allows both iOS devices and third-party BLE transmitters to act like an iBeacon. The latter have a specific and fixed format of Bluetooth Low-energy packets which identifies themselves as iBeacons. On the iOS side, applications can recognise whether they entered or left a region around an iBeacon and display push notifications accordingly, as well estimate of the distance to the transmitter. The main advantage is that it can run in the background whilst the device is locked, only showing up when necessary. This transparent integration in iOS devices provides a great user experience.

BLE packet

Figure 2.3 represents the iBeacon format of a Bluetooth Low-energy packet. The most relevant block is the Data Block of the Protocol Data Unit (PDU) where:

- The 9-bytes iBeacon prefix is a fixed value indicating that the present BLE devices follows the iBeacon protocol;
- The Proximity UUID is an unique identifier to differentiate the iBeacons from one manufacturer to the other. In other words, it means that every beacon belonging to a company has the same UUID;
- Major is a 2 bytes unsigned identifier but used to distinguish a set of iBeacons. For instance, all beacons placed in a store will have the same Major;
- Minor is the beacon unique identifier. These two bytes are used to identify individual beacons;
Lastly, **TX Power** is the 2’s complement of the strength of signal measured at 1-meter distance in the factory, and programmed natively into the beacon. This value can’t be changed and aids in better RSSI measurements, to determine more precisely how close a device is to a beacon.

### 2.3.3 Galeries LaFayette Haussmann

Galeries LaFayette Haussmann is a representative example of a class of applications. *LaFayette Galeries*, a shopping centre situated in Paris, has recently deployed an example of an indoor positioning system using Bluetooth Low-Energy and trilateration (Figure 2.4).

By having Bluetooth Beacons spread around the mall along with the corresponding smartphone application, allows customers to locate themselves on the map, trace a route to a desired store, find facilities and access detailed information about the venue. It also pushes messages to customers, including promotions if one is within the radius defined for a particular brand.

### 2.3.4 Low-cost third party beacons

Currently there are available a plenty of beacons from different brands, such as Qualcomm, Texas Instruments, some projects in Kickstarter and countless other unbranded beacons made in China. Many
formats are also available, e.g. for key chains, thin stickers, custom formats. This demonstrates the huge bet that has been happening around Bluetooth Low-Energy, and is expected to continue increasing along with the growing demand from the market.

2.3.5 Google Indoor

Google recently made available its own indoor location service, seamlessly integrated with Google Maps. It combines a plurality of data sources to give an estimate of user location: WiFi, GPS and smartphone sensors (gyroscope, accelerometer, barometer, etc) are fused to pinpoint user location on the map. This requires a preliminary mapping (using the fingerprinting technique) in order to record essential data, thus discarding indoor navigation for non-mapped buildings. Besides Google Maps, it is possible to integrate indoor maps (through Google Maps Application Programming Interfaces (APIs)) to enhance further mobile apps.

2.4 Critical Analysis

Table 2.1 provides an overview of the technologies and body of work analyzed in this section.

Some technologies (e.g. Ultra-Wide Band, Wireless LAN), even though they are reported as not requiring Line-of-Sight, will provide higher precision when LoS is available. The Cost referred in the table is not exclusively related with monetary value, but also with time and space constraints for they installation. By Enables Network Access is meant if we can access the Internet using the same infrastructure for indoor localization.

Next, we will examine the studied technologies to determine the best for our work. The chosen technology has to guarantee:

- Low energy consumption
- Good accuracy (∼ 1 - 2 meters)
- Reduced deployment cost
- Ease of use
- Scalability

Rationale:

Although accuracy is one of the most important parameters to take into account, other criteria were also considered in the below motivation.

- An Inertial Navigation System by itself does not provide a reliable accuracy. Along with WiFi, it provides relatively good accuracy, but only for a short time once it loses WiFi, due to the cumulative error coming from the INSs;
<table>
<thead>
<tr>
<th>Technology</th>
<th>Cost</th>
<th>Accuracy/Precision</th>
<th>Need of carrying device</th>
<th>Requirement of Line-of-Sight</th>
<th>Enables Network Access</th>
<th>Energy consumption</th>
<th>Smartphone support</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPS</td>
<td>Moderate</td>
<td>~ 3 meters</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Low</td>
<td>X</td>
</tr>
<tr>
<td>GSM</td>
<td>Moderate</td>
<td>~ 5 meters</td>
<td>X</td>
<td>X</td>
<td>Moderate</td>
<td>Low</td>
<td>X</td>
</tr>
<tr>
<td>Wireless LAN</td>
<td>Low (Zero if infrastructure already in place)</td>
<td>~ 3 meters</td>
<td>X</td>
<td>X</td>
<td>Low (Moderate to High)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>Moderate (Low in case of BLE)</td>
<td>~ 1.5 meters</td>
<td>X</td>
<td>X</td>
<td>Low</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Visual Recognition</td>
<td>High - Expensive devices and high computational power</td>
<td>Variable</td>
<td>X</td>
<td>X</td>
<td>High</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Ultra-Wide Band</td>
<td>Moderate</td>
<td>~ 1.5 meters</td>
<td>X</td>
<td>X</td>
<td>Low</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>RFID</td>
<td>Moderate (Low in case of Passive tags)</td>
<td>~ 2 meters</td>
<td>X</td>
<td>X</td>
<td>Low</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>Infrared</td>
<td>Low</td>
<td>Room accuracy</td>
<td>Depends on type of IR sensor</td>
<td>X</td>
<td>Low</td>
<td>Unusual</td>
<td>x</td>
</tr>
<tr>
<td>Near-Field-Communication</td>
<td>Low</td>
<td>Depends on tag density</td>
<td></td>
<td>X</td>
<td>Low</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Inertial Navigation System</td>
<td>Moderate to high</td>
<td>Decreases with distance</td>
<td>X (Except Smartphones)</td>
<td>Moderate</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Visible Light Communication</td>
<td>Low</td>
<td>High - tens of centimeters</td>
<td>X</td>
<td>X</td>
<td>High</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 2.1: Comparison of localization technologies
• VLC systems have the highest accuracy among all possibilities, but in case the LoS to the illumination is shadowed or the phone isn’t flat (parallel to the ground), it suddenly stops working. Moreover, these systems have an high-energy consumption due to the need of keeping the smartphones’ camera turned on.

• As stated before, GSM cannot provide enough accuracy for the scope of our system. Still, this technology can be combined with others to increase precision.

• WLAN could be a good choice due to its widespread deployment, eliminating the need for installing extra-hardware, even though it was not designed with the purpose of localization. Li et al. [40] stated some problems concerning using WLAN for indoor localization, namely the penetration losses through walls, the interference from other sources of the same frequency, and the effects of signal reflections. WLAN does not provide a reliable accuracy and is easily subject to fluctuation. Also, in case of large shopping centres, the current AP density might not be enough for indoor location, thereby leading to an extra deployment cost;

• The heavyweight drawback of GPS is its inefficiency indoors because buildings block GPS signals; thus becoming a not so interesting solution;

• The accuracy of a Visual Recognition system is hard to quantify, as it depends greatly on the training data and the used algorithms [41], in addition to the high required power burden to operate and poor robustness. Likewise, the deployment cost is high due to the necessary effort to produce an initial training database, which is difficult in large malls;

• Using NFC for this project could be interesting, but the repetitive action required by the user for the system to operate is a considerable inconvenient. Also using NFC would largely restrict the number of users due to three related issues: Apple iPhone has the 2nd highest market share[8]; NFC deployment on Apple devices only started in their last model, the iPhone 6; the use of NFC on their devices is restricted to Apple Pay.

• Employing RFID for indoor localization purposes would require the user to carry an RFID reader to scan the tags, along with the need of a high dense sensors infrastructure. Another limitation is the interference from sunlight;

• Infrared has limited range, low throughput, is easily subject to interferences (e.g. sunlight). Furthermore, it became a deprecated technology in terms of mobile devices, making it an unattractive technology;

• A big handicap with respect to Ultra-Wide Band is the need of specific hardware, and the low degree of feasibility of deploying an indoor positioning system using this technology, as Smartphones do not have embedded UWB;

• Concerning **Bluetooth**, first of all these devices have a high penetration in the society, with all mobile phones and tablets being equipped with this technology by standard, although it is necessary to deploy the beacons’ infrastructure. Secondly, due to the massive production of Bluetooth chipsets, the price *per* unit is low. Also, its 4th version has an effectively low energy consumption, outperforming VLC systems. Last, but not least, it proves to be a scalable and extensible solution — scalability done simply by deploying more beacons, and coping with extensibility by adding more features to the mobile application;
Chapter 3

Architecture

In this Section we propose an architecture to fulfil the requirements of our proposal. The key idea is to take advantage of the fact that almost every person carries a mobile phone, by providing customers a context-aware application for their devices, while at the same time providing business owners a way to engage customers and increase revenue.

3.1 Requirements

The objective of this work is to provide a complete context-aware beacon-based platform. Therefore, the goals of our system are:

- **A flexible unified interface** - Our system must have a unified interface where business owners are able to manage all the information about customers, locations and media content. We want to abstract business owners from the complexity of deploying a beacon-platform on a new location.

- **A single mobile application** - Instead of each commercial space having its own application, we want to provide costumers a unique and generic application that can be used in a multitude of places. By maintaining the location data in our server, we abstract the need of storing anything on the users phone.

- **Provide indoor navigation** - The system must offer customers a way to browse for products inside a store and offer a to way locate them.

- **Low-resource usage** - Our system must avoid battery drain on the mobile devices and use the minimum bandwidth possible.

- **Changes-compliant** - The system must be capable of adapting to changes and variations in the environment, either by modifying the location floor plan, by adding more beacons or assigning new media content.

- **Present analytics** - The system must collect metrics and present analytics to the business owners, allowing for data mining in the future.


- **Increased customers engagement** - Our system must provide an interactive and individualized shopping experience.

### 3.2 System Modules

Bearing the above goals in mind, in this section we describe an architecture composed by three parts: the Server, the Client and the Digital Signage. We opted for a client-server architecture due to the need of centralization of information. A peer-to-peer architecture would not be a good choice to implement, as clients are not available to provide services – they are just data consumers (e.g., the client mobile application). Furthermore, we want to provide a unique place for statistic data gathering, and for the purposes of our prototype, scalability problems do not apply. Moreover, we aim to have all data stored in the same place, manageable in a easy fashion. Although this may raise privacy issues for users, it is valuable for business owners, as collecting this data increases the potential for an eventual data-mining.

Hence, we developed a modular architecture, in order to facilitate the extension of the functionalities or their updates. Figure 3.1 presents a detailed view of the architecture in the form of a block diagram of the entire system. In the subsections below, we describe in detail what each module is responsible for.

#### 3.2.1 ShopAssist Server

The ShopAssist Server is the core of our system. It is responsible for coordinating the inputs from clients, process them and act accordingly (e.g., by sending content to the Digital Signage display).

The ShopAssist Server is also responsible for managing the unique identification of the customers, as they can be registered on different stores. This issue will be left for future work, as many options are available: either using the email address of the customer, or using OAuth\(^1\). For the purposes of this prototype, a unique username was used to provide customer identification across all locations, requiring the user to register in our platform.

Whenever the user device senses a beacon, the Server is sent the username and BeaconID (Major and Minor values of a beacon). This information is processed and matched to a known location. After this, Location Data is sent to the client, containing all the coordinates of the location map, beacons position, location of products, windows and doors. Furthermore, this information from the client is combined with media content previously stored in the Database, and sent to the Digital Signage in order for targeted advertisement to be displayed. These interactions will be better detailed in Chapter 4 along with Figure 4.18.

#### 3.2.1.1 CMS

Our system requires a unified interface where business owners are able to manage all the information about customers, locations and media content. The CMS is the module responsible for this. Therein lies

\(^1\)Open Authentication provides users a way to login using Google, Microsoft, Facebook or Twitter accounts without exposing their password
the place where business owners can manage or add new locations, administer content to be displayed on the digital signage and consult gathered analytics. It uses a database to store this information in a persistent manner.

Estimote has its own CMS, but we found it to be very limited and could not cope with what we wanted to provide. Table 3.1 depicts a comparison between Estimote and our CMS. Essentially, Estimote Location Manager only allows the user to modify the coordinates of existing points (corners of the room). Regarding beacon management, their CMS allows to modify only a minimal number of settings, such as its assigned location, its Major/Minor values and few other parameters. We want to provide business owners a flexible platform, where they are able to fully manage all data. This way we decided to implement our own CMS.

Our CMS module interacts with the Database, where all data is stored, and exposes a Web interface with the following functionalities:

- **Location Manager** - Allows the creation of new locations from scratch, or modification of already existing ones. In here, business owners can build up the map of his space, add beacons, windows and doors (to provide better accuracy, as claimed by Estimote).
Table 3.1: CMSs comparison

<table>
<thead>
<tr>
<th></th>
<th>Estimate CMS</th>
<th>Our CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add new Locations</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Manage Floor plan</td>
<td>Limited</td>
<td>X</td>
</tr>
<tr>
<td>Manage Windows</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Manage Doors</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Manage Beacons</td>
<td>Limited</td>
<td>X</td>
</tr>
<tr>
<td>Add/Assign media content</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Position Logger</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

- **Content Manager** - Allows assigning media content to each digital signage present on the corresponding location. Furthermore, it allows to aggregate customers within user profiles, to associate advertising campaigns to them.

- **Position Log** - In here business owners can view the collected analytics. They are able to track who, when and where users have been. All this information is stored in the database.

### 3.2.1.2 Database

We needed a way to store all location data. The database module is responsible for storing customer data, advertising media and maps of locations. The database is queried by server’s positioning engine and replies accordingly.

We opted for a SQL database rather than a NoSQL one, as the authors are more familiar with SQL. Moreover, our prototype is better suited to use an SQL database, where data is stored in a relational model with rows and columns, allowing us to perform more complex queries over the dataset (eg., Natural Joins).

### 3.2.1.3 Positioning Engine

The Positioning Engine handles inputs and processes it accordingly, either querying the database, sending the data to the presentation module of the digital signage or to the client. It exposes a RESTful API for the client and the digital signage to use, detailed later in Section 4.2.3.

Figure 3.2 presents an input and output example of the Positioning Engine. It maps a Beacon to a Location by querying the Database, returning the Location Data.

Figure 3.2: Positioning Engine input/output example – matching a Beacon to a Location
3.2.2 Client

The client runs our mobile application on his smartphone, which provides indoor positioning and allows customers to browse for products within the current location. It displays a list of products or product sections (whichever the Business Owner defines) in the main screen. The selected product is then marked in the map, enabling customers to guide themselves to the destination. The application also notifies the presence of the user in the store, causing the Digital Signage to be activated. Initially other features were considered, such as providing the itinerary to the car and pre-order goods at the entrance of the store. They were abandoned mainly due to Estimote limitations, as pointed out in Section 4.1.1.

The client acquires Bluetooth data (RSSI and BeaconID) through the Bluetooth Module and passes it to the location module. The BeaconID consists of a pair of numbers (Major and Minor) which uniquely identify each beacon within a location.

The Location Module is a wrapper around the Indoor Positioning System from Estimote, which provides positioning within a space. Once a beacon is detected, we fuse its BeaconID with customer identification (UserID) and send it to the ShopAssist Server. After this, Location Data (containing the map of the space) is returned from the server. That information is used to configure the Estimote API in order to provide indoor positioning, relatively to that location.

The Presentation Module displays a map of the location where the user is located. It uses the positioning data from Estimote to draw an Avatar on the map which moves according to users movement.

Although map-caching is not implemented in our prototype, it is supported by our architecture. It is represented in the Block Diagram (Fig. 3.1) by the passing $L_{ID}$ directly to the Presentation Module. This way, if the map has been already downloaded in a previous period, the customer would be able to locate himself on it, without contacting the ShopAssist Server. For simplification, $L_{ID}$ represents a symbolic location instead of coordinates.

3.2.3 Digital Signage

Consists of having several screens spread around the shopping mall to display targeted advertisements. Upon approach of an user to the screen, it displays potentially attractive promotions based on his user profile. Privacy concerns will be issued later, in section 3.3.

The Presentation Module of the Digital Signage is responsible for displaying media content, sent by the ShopAssist Server. It consists of a screen which shows individualized promotions for the customer nearby.

There is a REST API that is called to indicate the presence of the customer in the proximity and trigger actions accordingly. Moreover, the campaigns and media content are retrieved from the database, registering how many times a user has seen them.
3.3 Privacy & Security concerns

Privacy concerns
Once a user approaches a screen, personalized content is displayed based on consumer profile. Naturally, consumers will not be delighted to see their individual tastes displayed on large screens for everyone to see. The business owner is the only responsible for the content and has to make sure it is not embarrassing nor unsuitable for the customers.

Beacon Security Attacks
Beacon spoofing is a possible attack to BLE-based systems. It consists of modifying the Universally Unique Identifier (UUID) value of Bluetooth beacons, thus leading to incorrect data being displayed on the application. Fortunately, Estimote beacons features a secureUUID mechanism that changes beacon's ID in a fixed time period to a randomized and encrypted value. Then, the mobile app recognizes the beacon by using the Estimote Cloud for translation and decryption. This requires constant connection to the Cloud, but does not necessarily represent a problem, as connectivity is required for the rest (media content, individual offers).

3.4 Chapter Summary

In this chapter we presented an overall description of our system architecture illustrated with a block diagram that responds to the defined requirements. This architecture is intended to facilitate the deployment of an indoor localization system in shopping malls, while at the same time providing customers an interactive shopping experience. The system is designed to be most generic possible, either to customers and business owners, by:

• Avoiding the need of multiple mobile applications for different stores

• Abstracting the need for a deeper knowledge from business owners to setup a location using Estimote framework. We provide a layer over Estimote Framework where business owners are guided through a set of five steps to setup their location, facilitating the deployment.

In the following Chapter will be thoroughly described the implementation details of the presented architecture.
Chapter 4

Implementation

The present chapter details how we implemented the aforementioned architecture. We will present the used technologies and platforms to accomplish our prototype, along with a detailed description on how each Module is implemented.

4.1 Adopted Technologies

In this section we will describe the adopted technologies (Platforms and Implementation Choices) to accomplish the implementation of our prototype.

4.1.1 Platforms

Apple iOS

Apple iOS was the chosen mobile operating system to develop the mobile application. Although Android\(^1\) is open-source and has an higher market share than iOS, the Estimote Indoor SDK is only compatible with Apple devices. The main reported reason by the company is that the Bluetooth chipsets and antennas are not standardized on Android devices, unlike Apple devices where the same Bluetooth Antennas are used. Each Android device would report a different Bluetooth RSSI, thus making it very complex to develop algorithms to calculate indoor position this way.

Play Framework

Play\(^2\) is a ‘full-stack’ framework written in Java and Scala allowing to develop lightweight and efficient Web applications. It follows the Model-View-Controller (MVC) architectural pattern, where the application is split into three layers: Model, View and Controller.

- The Model is responsible to manage the data and its behaviour. This is where we execute CRUD (Create, Read, Update e Delete) operations over the database.

- The Controller handles the inputs to command the Model, i.e., calls methods from the Model. It makes the bridge between incoming data from the users and the Database.

\(^1\)http://www.android.com/

\(^2\)https://www.playframework.com
• The View is the representation of the information (User Interface).

Play was the chosen framework to develop the Webapp of this project because of the aforementioned characteristics and due to its easiness of use. Diving in the framework itself, the Java language was used in the Controller, and a blend of HTML+Scala+Javascript for the View.

**Estimote**

We chose to use Estimote’s platform as a basis to our project. It offers a complete Beacon platform, providing a fully-documented SDK and API for their beacons. It is possible to purchase Dev Kit’s with 3 beacons each, for a reasonable price. Four beacons are required by Estimote to provide indoor localization, even though, theoretically, 3 would be enough to give positioning using triangulation. Moreover, it features a large support community, having an SDK for Android and iOS used by more than 40.000 developers.

*Estimote Indoor SDK* is a set of tools capable of providing indoor positioning. It uses the Proximity-based and triangulation positioning techniques over Bluetooth Low-Energy. However, this system has some limitations, which would impact our work, namely:

• **No multi-room support** – At the moment Estimote only supports mapping of rooms separately.

• **No multi-level locations** – The same way multiple rooms per space are unsupported, there is also no support for multiple floors.

• **No beacons in the middle** – Beacons can only be positioned in the perimeter of a room. This is a considerable limitation in our case, as a store may have several aisles, and placing beacons on the shelves would be useful for our objectives.

• **Does not accept non-standard shapes** – Estimote does not provide support for curved walls, only allowing for rectangular-shaped rooms.

We tried to overcome some of these limitations by mapping two rooms along with a corridor inside a single location. Figure 4.1 shows an attempt to map a more complex space using the Estimote Framework. Unfortunately this workaround has a problem: The reported position makes the avatar jump the walls between rooms.

To use Estimote Indoor Localization features, first it is necessary to map the space. This could either be done using the Estimote Indoor iOS application, or by setting the location manually using the SDK.

• **Estimote Indoor iOS application** - This quick mapping method consists of two steps: first, a beacon is placed in the centre-middle of each wall; after this, the user walks around the perimeter of the room, stopping in front of each beacon. All the data from phone sensors is being continuously collected to infer its position and movements, allowing for the creation of a map of the area. This method is subject to drift and was only used in a early stage of our prototype development for testing purposes. Quickly we found out that it did not offer enough precision.

• **Using Estimote Indoor SDK** - The *EILocationBuilder* class of the Estimote iOS Indoor SDK provides methods to setup location manually. This was the chosen way to add new locations to
Figure 4.1: Example of a more complex space mapped using Estimote Indoor SDK

our ShopAssist platform, due to the higher accuracy achieved, as it gives a greater control of the shape, dimensions and orientation of the mapped space. The available methods to setup a location are: `setLocationName`, `setLocationOrientation`, `setLocationBoundaryPoints`, `addBeaconIdentifiedByMac`, `addWindowWithLength` and `addDoorsWithLength`. Later will be presented an example of setting up a location using those methods (Listing 4.2).

The SDK accepts Windows and Doors as parameters, as they reflect and absorb radio signals in a different fashion comparing to walls. Although Estimote claims that it is advantageous, resulting in better precision, neither their Indoor iOS app nor their CMS permits to define those parameters. Additionally, we made an experiment by setting up a location using the SDK containing windows and doors, and sending it to the Estimote Cloud. In there, we can consult our mapped locations, but surprisingly windows and doors are not even shown – only the beacons along the walls.

Several attempts were made to map the space using the Indoor iOS application, but none of them provided a reasonable map of the space. Figure 4.2 depicts a comparison between mapping the same location using the Estimote Indoor iOS application versus using Estimote Indoor SDK.

Hence, we can conclude that setting up a location using the Indoor SDK is definitely better than using the Estimote’s iOS Indoor application, as it allows to specify precisely all location parameters. The hurdle here is the requirement of iOS development knowledge (Swift or Objective-C languages) to use the Estimote Framework. Therefore, we decided to implement a feature in our CMS which mitigates this major drawback, allowing business owners to set up a location without requiring any iOS development.

4.1.2 Implementation choices

Bluetooth Low-energy

Bearing in mind that there is no perfect technology, we found that Bluetooth Low-energy outperforms other assessed technologies, making it the most suitable technology. Indeed, Bluetooth is becoming a
trend for indoor positioning, as its 4\textsuperscript{th} version eases the process of localization (as mentioned in Section 2.1.4) along with its low power consumption.

**Swift**

Swift was the chosen programming language for iOS. Although its predecessor, Objective-C, was also considered, Swift is the new programming language for iOS devices, being faster and with a much cleaner and neat syntax compared to Objective-C. Also, we wanted to make our app future-proof, compatible with the newest iOS versions. Thus, our Mobile App was coded with its latest version: Swift 2.0.

**MySQL**

For this project, a simple and efficient database was required. We found that MySQL fits our needs, for its simplicity and high performance. Also, there are no requirements with large amounts of data as this is a prototype. If it was deployed in many commercial spaces, a different database or a cluster setup may have to be considered.

**JavaScript Object Notation (JSON)**

JSON was the chosen format to exchange messages between the Server and the Client (Mobile app). Extensible Markup Language (XML) was the other main possibility, but it was easily discarded since JSON is much more lightweight, flexible and easy to use. With XML, the objects are wrapped within open and close tags, adding an unnecessary overhead, contrasting with JSON where the tags are defined only once. Listing 4.1 demonstrates one of the exchanged JSON messages within this project. The referenced example is a simplified message returned by the server to the client, for the initial location setup. All distance units are in meters. This message contains all the location data to be
passed to the `ESTLocationBuilder` class of the iOS indoor SDK (better detailed in Section 4.3), namely:

- `points` is an array of `point` objects that contains the coordinates of the points to draw the shape of the location, drawn by the order represented by `posNr`. Note that a minimum of 4 `point` are required, but for simplicity's sake only one is represented;

- `doors` is an array with the data about each door of the space. The door is located in `segment` 0, distanced (`dist`) 0.15 meters from left `side`, with a `length` of 1.30 meters;

- `windows` array contains a set of objects which represent a window, with the corresponding attributes (equal to `doors`);

- `beacons` contains the beacons and their attributes. Likewise, a minimum of four beacons is required by Estimote to operate. For simplicity, we represented just one;

- `digsignage` describes where digital signages are located and their associated beacon (Major and Minor number);

- `products` lists where products (or zones - whichever the business owner previously defines in the CMS) are located.

```json
{  
    "locationId" : "Office",
    "orientation" : 90.0,
    "user" : "Bruno",
    "points" : [{
        "posNr" : 1,
        "x" : 0,
        "y" : 5.80
    }],
    "doors" : [{
        "dist" : 0.15,
        "segment" : 0,
        "length" : 1.30,
        "side" : "left"
    }],
    "windows" : [{
        "dist" : 0.15,
        "segment" : 1,
        "length" : 1.85,
        "side" : "left"
    }],
    "beacons" : [{
        "dist" : 2.90,
        "segment" : 0,
    }]
}
```
Listing 4.1: Response message with Location Data for the initial setup. Note that a minimum of 4 point objects (Lines 5 – 9) is required, but for simplicity we represented just one. The same applies to beacons (L. 22 – 27).

Segment, Length, Distance and Side will be recurrent keywords henceforth. Those are required fields for Estimote SDK to set up Beacons, Doors and Windows of a Location. A Location can be understood as a physical location, represented by a simple polygon constructed out of points. Figure 4.3 illustrates an example where the reader should imagine himself as standing in the middle of the room, looking at the first segment of wall (No. 0). We can observe that the beacon is distanced 2.0 meters from right side. In this example, the parameters for the beacon would be:

- **Segment** - 0,
- **Distance** - 2.90,
- **Side** - Right,
- **Length** - N/A (only applicable with Doors and Windows, meaning the length of it).

Listing 4.2 presents the Swift code-snippet of setting up a location object using the *EILLocationBuilder* class, for the JSON message of Listing 4.1. The orientation value is measured in degrees with respect to the magnetic north, as exemplified in Figure 4.4 denoted by $\beta$ symbol.

### 4.2 ShopAssist Server

As stated in section 3.2.1, our work assumes the existence of a platform responsible for handling and processing all input from clients and business owners. The ShopAssist Server, which was implemented using the Play Framework and uses a MySQL database, is the responsible for this feature. Below are presented the components of the server: CMS, Database and Positioning Engine.
Figure 4.3: Clarification of Estimote SDK input Fields

```
location.setLocationName("Office")
location.setLocationOrientation(90)
location.setLocationBoundaryPoints([[EILPoint.init(x:0.0, y: 5.80)])
location.addBeaconIdentifiedByMac("AABBCCDDEEFF", atBoundarySegmentIndex: 0,
inDistance: 2.90, fromSide: EILLocationBuilderSide.RightSide)
location.addDoorsWithLength(1.30, atBoundarySegmentIndex: 0, inDistance: 0.15,
fromSide: EILLocationBuilderSide.LeftSide)
location.addWindowWithLength(1.85, atBoundarySegmentIndex: 1, inDistance: 0.15,
fromSide: EILLocationBuilderSide.LeftSide)
```

Listing 4.2: Setting up a location object using `EILLocationBuilder` class.

Figure 4.4: Clarification on how to measure the space orientation. $\beta$ denotes the angle between the magnetic north (represented by the green arrow) and the location.
4.2.1 CMS

The CMS website was built using the Bootstrap framework\(^3\). This framework allows the development of responsive websites\(^4\) using HTML, CSS and JavaScript.

The index page of the CMS allows business owners to login with their username and password. A link to a registration page is also displayed. After successfully logged in, the main page of the CMS is presented.

Below are described the main features of the CMS, namely the Location Manager, the Content Manager, the Position Log and the Product Management page, where Shopkeepers are able to manage all data about locations.

4.2.1.1 Location Manager

In our platform, a Location is considered a store. Our Location Manager allows the Business Owners to have many stores associated. It has two main features, namely:

- **Add a Location** - Allows adding a new location to our platform. To do so, first the business owner needs to define two axis (X and Y) in the Cartesian coordinate system. Afterwards, the coordinates of location corners need to be mapped out relatively to the defined axis, along with the windows/doors parameters (length and relative distance to one corner). Also, the same parameters need to be mapped out for the beacons. All these values will be inserted in our CMS, which guides the user through a set of five steps to create a new location:

  1. **Set Location Attributes**: Location name and Orientation of the space (in degrees) with respect to the magnetic north.

  2. **Add Points**: This is where business owners set the location boundary points. Has two numeric input fields (X and Y) representing the coordinates of the corner of the room to be added. The user needs to add at least four points (minimum required by Estimote SDK) before moving on to the next step. As seen on Figure 4.5, the map of the space is drawn as the user inserts new points, allowing him to visualize how the location is being mapped. The map is drawn using HTML Canvas and resizes automatically so that the location always fits the box.

  3. **Add Beacons**: Asks for the beacon’s MAC Address, Major, Minor, Segment, Distance, Side and Zone (optional), as seen on Figure 4.6. Once the beacon is added, it is immediately shown in the map represented by a small green square. A minimum of four beacons is required (one per wall). The MAC address, Major and Minor numbers are identifiers to distinguish each beacon, present in the iBeacon protocol, as stated in section 2.3.2. Segment, Distance and Side identify the wall segment where the beacon is located, and its distance to one of the sides, respectively. The Zone field is required when the Business Owner wants to associate a Digital Signage with a beacon. This name will later be used when uploading media content.

\(^3\)http://getbootstrap.com/

\(^4\)A responsive website aims to adapt to the wide range of devices available (tablets, smartphones, laptops)
4. **Add Doors**: Allows adding doors to the location. Its Segment, Length, Distance and Side need to be provided. Once that information is submitted, the door is drawn on the map with a brown colour (see Fig. 4.7).

5. **Add Windows**: Requires the exact same parameters as adding a door. Figure 4.7 presents the final step of setting up a location, where the added door can be observed represented by a brown line, along with a Window drawn with a blue colour from the previous step.

- **Modify a Location** - Allows the modification of an existing location. In here users can modify all attributes of his locations. Figure 4.8 presents the Modify Location page, where is possible to:

  1. **Edit Location Attributes**, to modify location name and orientation.
2. **Edit the Floor Plan**, by removing or adding new points.

3. **Manage Windows**, to add or remove existing windows.

4. **Manage Doors**, to add or remove existing doors.

5. **Manage Beacons**, to remove or add a new beacon (Fig. 4.9).
4.2.1.2 Content Manager

The Content Manager allows assigning media content to each digital signage present on the location. For the purposes of our prototype it is only possible to use images (PNG and JPEG) as media content. It has the following features:

1. **Add Content to Location** - Allows business owners to add new media content and assign it to a beacon. One has to choose the image to upload, select a group of users (or user) to whom that content will be displayed to, and select the beacon zone from a dropdown. This dropdown menu contains the zones previously defined when beacons were inserted – see Figure 4.10.

   ![Figure 4.10: Step 2 - Adding media content to a Location](image)

2. **Manage User Groups** - Adds a new layer of abstraction by allowing the creation of new user groups and their management. Specially useful when Business Owners want to wrap users within a group (ex.: students) and display advertisements accordingly – Figures 4.11(a) and 4.11(b).
3. **Manage Existing Content** - In here it is possible to remove or edit the attributes of previously inserted content. It displays a preview of the upload image along with its attributes – **Group** defines to which user group should the image be displayed to; **Location** and **Zone** define the location and beacon zone respectively; and the last field displays the upload date. This page allows business owners to change the user group of the image, modify the belonging location and edit the zone (to assign the media content to another beacon) – see Figure 4.12.
4.2.1.3 Position Log

This feature displays the history of user recorded positions of any location belonging to the business owner. It is possible to see on the map where users have been, filtering it by time interval. The empty dates are greyed out on the calendar. This allows the business owner to analyse the most popular spots and dead zones of his store, being able to take actions accordingly, such as the rearrangement of the store’s disposition.

Each location of the user is represented on the map by a black dot. It is also possible to filter it by user, as seen on Figure 4.13.

![Figure 4.13: CMS Position Log feature – Allows filtering by user and by time interval through a date/time picker widget.](image)

This feature can be extended in the future with analytics, such as the generation of traffic pattern maps, duration of visits measurements, finding dead-spots and so on.

4.2.1.4 Product Management

This feature is used to manage the list of product categories shown in the mobile application. The business owner inserts the name (either a product or section) along with its coordinates, which can later be searched in the app for the customer to know where are the products located – See Fig. 4.14.
4.2.2 Database

A MySQL database is used to store all locations data. Below, in Figure 4.15, we present a view of the existing tables of the ShopAssist database along with a description. We structured the data in the same format the Estimote Indoor SDK handles it, i.e., with the parameters length, dist, side, instead of absolute coordinates.

- **Points table** – Stores the coordinates (x, y) of the boundary points of the space. \( \text{posNr} \) is an integer necessary to know in which order should the points be drawn. This is specially important when the mapped location has more than 4 walls. \( \text{location} \) is a Foreign Key to the Locations table allowing to know to which location it belongs to.

- **Beacons table** – Responsible for storing all data associated with a beacon, namely:
  - \( \text{location} \) - To which location it belongs
  - \( \text{major} \) - Represents the beacon's Major ID
  - \( \text{minor} \) - Likewise, represents the beacon's Minor ID
  - \( \text{mac} \) - MAC address of the beacon
  - \( \text{segment} \) - Wall segment of the space where the beacon is located
  - \( \text{dist} \) - Distance from the beacon to one of the corners of the space
  - \( \text{side} \) - A string that specifies the side for which the above distance was considered, represented by one of the following values: "left" or "right"
  - \( \text{zone} \) - An optional value which specifies the beacon zone. This value is later used in the CMS to assign media content to a beacon, i.e., only beacons with this field defined can be chosen from within the dropdown list (visible in Fig. 4.10).
This table is queried to do the matching between (Major, Minor) and a location. It maps a beacon to a location, being necessary when the mobile app requests location data once a beacon is listened.

• **Locations table** – Stores information about the locations, *ie*, it retains the basic parameters for a location: its name, orientation (in degrees represented by an integer) and the respective owner. This table could be extended in the future to add new attributes to the location (address, phone etc).

• **Positionlog table** – Stores customers data, namely: *x* and *y* coordinates to know the past positions, *user* to know who, *timestamp* to know when, and *location* to know where someone has been. This table can be mined in the future to infer traffic patterns, draw heatmaps, find deadspots and so on.

• **Doors table** – Responsible for storing the necessary information to know where doors are located. The meaning of each attribute is the same as those found in Beacons table, except for the new *length* attribute which specifies the length of the door.

• **Windows table** – In charge of storing information about where windows are located. Likewise, the
meaning of each attributes is the same as found in Doors table.

- **Content table** – Stores the content to be displayed on a Digital Signage. *image* column stores an image in a Binary Large Object (BLOB)\(^5\) data type. *group* is a Foreign Key for ID on Groups table, which specifies to which group that content belongs to. Otherwise, if this value is not set (is null), the *user* field is set, which is a Foreign Key to ID on Users table to know the user to whom that content will be displayed. *timestamp* defines the date of insertion of the content (in milliseconds). *zone* field defines the zone where the content will be shown *(eg., entrance)*, used for the Digital Signage. *filename* specifies the file name of the inserted content and *seen* field is an integer which lets us know how many times a user/group has seen that content.

- **Groups table** – This table is used to group users as the Business Owner wishes. It allows him to define the content (promotions) for user groups, instead of repeating the same content for each user. Stores a primary key ID, which is an auto-incremental integer, and the group name.

- **Users table** – This table stores the customers username, a salt and the hash of the password concatenated with the salt. This salt is generated randomly by the server upon user registration, using the `SecureRandom()` Java Class, which provides a cryptographically secure random number generator. We store a 32-byte salt on the database to avoid pre-computed SHA-256 dictionary attacks, possible if we only used the hash of the password. To perform user login, we pick up the stored salt, append it to the provided password, hash the resulting string and compare to the hash stored in the database.

- **Products table** – Contains information about where products are located. *name* stores the name to be displayed, *x* and *y* for its coordinates, and *location* is again a Foreign Key to know to which location it belongs.

### 4.2.3 Positioning Engine

As stated in Section 3.2.1.3, our Server exposes a RESTful API. The routes file of our PlayFramework server lists the routes needed by the application, translating the incoming HTTP requests to a controller method. Listing 4.3 presents the most important API methods made available.

```
1 POST /api/location controllers.Application.postLocation()
2 POST /upload controllers.Application.upload()
3 GET /images/:id controllers.Application.getImage(id : String)
4 GET /api/ws controllers.Application.webvSocket()
5 GET /drawrest controllers.Application.getDrawRest()
6 POST /storelogin controllers.Application.postStoreLogin()
```

*Listing 4.3: Part of the ShopAssist Server API*

1. Retrieves location data according to the received beacon Major and Minor. Builds the JSON message and returns it with a Content-Type\(^6\) of `application/json`.

\(^5\)An object capable of storing large amounts of data, either images, documents, audio or video files

\(^6\)Content-Type is a HTTP Header field which specifies the nature of the data enclosed in the HTTP Body.
2. Gets the image file contained inside the body of the MultipartFormData message and writes it on the database as a BLOB, along with other parameters.

3. Retrieves an image of id identifier from the database. Image bytes are returned as ‘image/jpeg’ Content-Type. Used to display the images in the Digital Signage.

4. This GET method starts the Websocket handshake\(^8\) in order to create the connection between the server and the Digital Signage.

5. Used to display the Position Log in the CMS. Queries the database and returns a JSON object containing an array of X and Y coordinates of the desired user.

6. This is where the Digital Signage identifies himself within the location. This action is performed by the business owner upon initial configuration/start-up of the Raspberry.

In Figure 4.16, it can be observed that the client (mobile application) communicates with the server through a REST API, and this, in turn, communicates with the Digital Signage through a Websocket to update its media content.

![Figure 4.16: Communication methods between Client, Server and Digital Signage](image)

### 4.3 Client

The ShopAssist iOS mobile application (Fig. 4.17) allows customers to have a richer experience while shopping. The main features of the mobile app are:

- **Indoor positioning** – By using the Estimote Indoor Location framework, it is possible to get an approximate location of the user and draw it on the map, represented by an avatar.

- **Search for Products** – Customers are able to locate products or sections of products, whichever the Business Owner defines on the CMS. The list is presented in the main screen, and upon clicking on a list item, a marker icon is shown in the map on the corresponding product location.

---

\(^7\)MultipartFormData is an encoding method for encoding the data in HTML Forms when submitting it to the server.

\(^8\)As described in RFC 6455, the Websocket protocol enables bidirectional communication between two hosts. The client starts by sending a Websocket handshake request to upgrade the HTTP protocol, and the server responds with an upgrade message to the Websocket protocol.
This way the user can easily spot where to find the desired product, and guide himself through the space.

- **Position reporting** – Every second the application reports its position (X and Y coordinates) to the ShopAssist server along with his username, as seen on the sequence diagram (Fig. 4.18). The server then adds a timestamp, and writes it all on the database, enabling to know when, who and where users have been. This real-time data collection provides business owners an wide range of added value.

![Sequence diagram showing user's location and interaction with the shop floor](image)

**Figure 4.17: Screenshot of the mainscreen of the ShopAssist Mobile app**

**Interactions**

Figure 4.18 depicts the steps and interactions resulting from the mobile application on a sequence diagram.

Firstly, the Client (mobile application) queries the Server through a REST API to perform the user authentication, and this, in turn, queries the database. The result of the authentication contains the salt and the previously calculated hash of the user’s provided password combined with the salt. Subsequently, with this information, the server calculates if the provided users credentials are valid and return a simple JSON message to the mobile client with the authentication result.
Hence, in case of a successful authentication, the application starts a loop listening for beacons, displaying a message on the mobile screen saying 'Waiting for a beacon'. Once a beacon is sensed, the loop ends and the beacon’s major and minor ID along with the logged username are sent to the server using the HTTP POST request method. Then, the server queries the database in order to retrieve the location data (its points, windows, doors, and so on). This information is used to build the JSON message to be passed to the Client, which consequently will setup the location object to be used for indoor positioning.

Then, with the above step finished, the positioning loop begins, where the user’s position is calculated by Estimote framework. These coordinates are used to draw the avatar on the map. Subsequently, the application kicks off a background task to send a POST request to the Server to store the user’s position. The server inserts these values on the database, along with the logged username on the application and the current timestamp.

In case the user is closer by less than one meter to a Digital Signage, the beacon’s major and minor ID, along with the username are sent to the server using a POST method. The server then acts accordingly, querying the database in order to retrieve contextual data (media content) for that user. Finally, it updates the Digital Signage with that content so it can be displayed to the user.

**Error handling**

The mobile application features error handling and displays a message to the user within these:

- *Wrong login* – Displays a popup accordingly.
- *Failed to connect with server* – Error thrown if the server did not answer in time, probably when it is offline.
- *Invalid location received* – Thrown when the received JSON message could not be used to set up the location.
- *Outside location* – Happens when Estimote Indoor framework could not locate the user within the received location from the Server. Waits a predefined amount of time and tries to locate again.

We use 5 libraries in our application, namely:

- *Alamofire* - Alamofire is a Swift-based asynchronous networking library which provides an interface for common HTTP request and response methods.

- *SwiftyJSON* - Handling JSON in Swift is painful, since Swift is very strict about types. Thereby, SwiftyJSON library facilitates parsing JSON files, which are an important part of our application.

- *EstimoteSDK* - EstimoteSDK arrives as a framework rather than a library, allowing the interaction with Estimote beacons.

- *EstimoteIndoorSDK* - Used to provide real-time indoor positioning using Estimote beacons.
Our iOS application is composed by two views, contained in the Main.storyboard file, and three classes: the AppDelegate, LoginViewController and the MainViewController. The ViewController classes are responsible for controlling and managing a View (in the Main.storyboard).
Figure 4.19 depicts how the AppDelegate, Views (Storyboard) and ViewControllers interact together. As may be observed, the AppDelegate is the entry point for the application. In here we request user permission to use Location Services and once the permission is granted, the LoginViewController object is created. This object is responsible for setting up the corresponding view, where the user is able to fill in his username and password. Once the login button is pressed, the input data is parsed by the LoginViewController, which sends a POST request to the ShopAssist Server with the Alamofire library. The response is returned in JSON format and parsed with SwiftyJSON. In case of a unsuccessful login, an AlertDialog is shown to the user asking him to retry; otherwise the MainViewController is instantiated and the corresponding view is presented. Although a View Controller can manage more than one view, we opted to have a more decoupled structure in our app.

Our MainViewController class adopts two important delegates\(^9\): the ESTBeaconManagerDelegate and the EILIndoorLocationManagerDelegate. The first one is responsible for monitoring Estimote beacons, and the latter is in charge of delivering position updates.

---

\(^9\)Delegation is a common design pattern in iOS development that enables a class to hand-off some of his responsibilities to another instance
of for-cycles to dynamically set the location boundary points, add beacons, add doors and windows. This information is used to build the location object using the `EILocationBuilder` class. We then start indoor positioning within the built location object, using the `startPositionUpdatesForLocation` method.

Listing 4.4 depicts a code snippet of the delegate method from `EILIndoorLocationManagerDelegate` which delivers the indoor positioning updates more than once per second. The `EIL OrientedPoint` class represents a geometrical point, containing its orientation along with X and Y coordinates. We use this data to draw the avatar on the map. The variable `positionAccuracy` delivers an approximate error value between 1.00m and 4.24m and it is also shown in the main screen of the application for debug purposes.

```swift
func indoorLocationManager(manager: EILIndoorLocationManager!,
didUpdatePosition position: EIL OrientedPoint!,
withAccuracy positionAccuracy: EILPositionAccuracy,
inLocation location: EILocation!)
```

Listing 4.4: Method header responsible for delivering position updates. This method is called more than once per second, depending on CPU performance of the device.

Regarding the data structures of our application, we maintain two arrays of `Product` and `Point` structures, respectively. `Product` structure contains the Product name, along with the corresponding coordinates (X and Y floats). This is used to setup the Table View which displays the list of products. Once a product is selected from the list, the map marker icon will move to the corresponding coordinates within the map.

`Point` structure contains Major, Minor, X and Y data fields. The array containing this data structure stores information about the beacons assigned to the Digital Signages. This allows us to calculate the distance between it and the device. If this distance is less than one meter (programmable value), a notification is shown in the iOS device saying "Look the nearest screen". Furthermore, a POST is sent to the ShopAssist server in order to change the content of the digital signage.

### 4.4 Digital Signage

Each Digital Signage has a Raspberry\(^\text{10}\) running Raspbian\(^\text{11}\) Operating System with a webpage opened in fullscreen. This page displays a continuous loop of default images (chosen by the Shopkeeper), and upon user approach it silently starts displaying targeted images for the nearby user. After a predefined time without receiving any signal from the user, the default set of images starts to be displayed again.

To identify unequivocally each Digital Signage, a login is required. On the startup of the Raspberry, a bash script is executed which opens the Midori browser in fullscreen and displays the StoreLogin page, where the Location ID and Zone fields have to be provided. Upon submission of this information, it redirects to the aforementioned page where images are shown in infinite loop. This image loop was implemented with the Bootstrap Carousel Plugin, which is a component for cycling through a set of images.

\(^{10}\)A small-sized computer, with 1GB RAM, Wi-Fi connection, Ethernet and HDMI ports.

\(^{11}\)http://www.raspbian.org
Every digital signage has a beacon associated with it, normally positioned above, as seen on picture 4.20.

To keep the content of the digital signage updated, we need an alive connection with the server. With HTTP request-response method, the digital signage module would have to periodically make GET requests to the server to know if there is new content to be displayed. The server could not send data unless the digital signage explicitly requested it.

Thus, we need a bidirectional connection, and Websockets do offer this functionality. A Websocket connection between the digital signage and the server is maintained, allowing the server to immediately send updated data once available. As explained in section 4.4, a set of images is displayed on the digital signage. Upon approach of a user to the digital signage, new images will be displayed accordingly. For this, we use the already opened socket connection and use Javascript to append a new HTML Div to the existing ones with the new image – Listing 4.5.

```
var connection = new WebSocket(wsurl('/api/ws'));
connection.onmessage = function (e) {
    $('#carousel').append('<div class="item" > <img src="' + e.data + '" alt="Image"></div>');</

Listing 4.5: Javascript code snippet executed to update the content of the Digital Signage
```

4.5 Chapter Summary

In this Chapter we started by detailing the adopted technologies to implement our prototype. Afterwards, a thorough description of the implementation details of each module was presented. We recall the main highlights of our platform:
1. A generic mobile application, which can dynamically adapt to new locations (if previously mapped in the CMS). The customer only needs to install one mobile application, which works in multiple stores. This allows business owners to reach a greater number of customers. Customers can also browse for products in the app.

2. A CMS for business owners, allowing them to target promotions for user groups or to specific users. Also, our CMS was designed to give business-owners full control over their locations.

3. Digital Signage, with varied functionalities: either welcoming customers on store entrance, or recommending customers products they are likely to buy.

We aimed to fill the gap between an indoor localization system and its widespread usage on shopping malls. Some faced issues were left for Future Work that need to be further studied in the situation of a larger deployment. On the following Chapter, our prototype will be evaluated and the results presented.
Chapter 5

Evaluation

The present Chapter presents the methodologies and results of our tests made towards evaluating our prototype. We will test the platform as a whole (Mobile app, CMS and Digital Signage), assessing the capability for a larger deployment, i.e., in a retail space. In Section 5.1 we define the objectives of the tests made. Section 5.2 details the Scenarios where the tests were conducted and in Section 5.3 we present the results along with a discussion.

5.1 Tests Objectives

As we aim to provide an accurate indoor localization platform for shopping malls, we assessed the following parameters:

- **Positioning accuracy** - To evaluate the accuracy of our prototype, we ran accuracy tests on different places and with different devices and checked how much the precision varies. We also varied the proximity to the beacons in order to evaluate the error fluctuation;

- **Server Response time** - To assess the performance of our server in case of high load, we measured the response times with a varying number of client requests per second;

- **Functionality and Usability** - User tests were made in order to assess the functionality and usability of our prototype (both the mobile application and the CMS). We also examined how contextual notifications affect customers behaviour, upon approximation to a digital signage.

5.2 Tests Scenarios

In this section we describe the two main test scenarios where our evaluation took place – an university office and a room – and the used equipment. In both places (Office and Room) we were expecting difficulties regarding the accuracy of the prototype due to the unstable radio signal propagation, presence of metal cabinets and other interference-causing objects nearby.
All beacons were configured with a broadcasting power of 4 dBm (maximum) and an advertising interval of 200 milliseconds. Those are the recommended settings by Estimote for optimal performance regarding indoor location.

5.2.1 Room

As a first deployment, a prototype of our solution was implemented in a room of an apartment, with an area of 22.02 square meters. We evaluated the accuracy achieved with six beacons on a single room. To measure the accuracy error, we marked points on the floor of known coordinates, distanced 1-meter from each other. These points were marked varying only the X-coordinates and in another test varying only the Y-coordinates, as illustrated on Figure 5.1.

![Figure 5.1: Illustration of the performed tests on a room](image)

Thirty two measurements were taken while walking along the blue lines marked on the figure and stopping on the black-doted points, registering the position reported by the app (x and y coordinates). The device used for this testing was an iPad Mini 2.

Another testing was also conducted in this space measuring the error in 6 points, situated in proximity to the 6 beacons. marked in the Figure 5.1 with a brown dot. For this testing, a total of 12 measurements were collected (two per point), with the device touching the beacon in a flat position.
5.2.2 University Office

Accuracy tests
Likewise, we made accuracy tests on an university office: a 17.50 square meter space (5.69 x 3.08 m), with six beacons placed along the walls. Figure 5.2 presents a screenshot of the application during accuracy tests, where it is possible to observe the layout of the office and the beacon’s disposition.

User tests
For user tests, we also considered and mapped the adjacent office, as seen in Figure 5.3. This allowed us to evaluate our prototype in a more similar way to real-world scenario, with a bigger area.

A screen was placed near the entrance of the left office welcoming who passes by, displaying a message on the screen with the person’s name. Another screen was placed near the coffee machine in the right sided office, which displayed a set of default images and upon approach of the user, a message was shown with the coffee price.

There was also another screen near a pair of shoes on the left office, and upon users’ approach, a message was displayed informing if their shoe size was available in stock. Although this information was hardcoded in our prototype for testing purposes, it demonstrates the wide range of possibilities to be extended and integrated with other tools.
Seven Engineering students of our university, aged between 23 and 25 years old, were asked to try out the ShopAssist mobile application running on our iPad. A set of three tasks was given them, as noted below:

- **Task 1** - Open the app and login with the following credentials: username *bruno* and password *bruno*. Find out where the Coffee Machine is located through the application. What is the price of the coffee?

- **Task 2** - Get close to the entrance of the office 2N11.5. Have you noticed any difference on the nearest screen?

- **Task 3** - Browse for the shoes through the application and approach it. Verify if your shoe size is available in stock.

Task 1 was relevant to find out if users are able to browse for products using the application and find them inside a location. Task 2 was important to analyse how contextual and individual content displayed in the Digital Signages may got unnoticed. Still, task 3 was pertinent to investigate if users will enjoy and benefit from a contextual and individual system like ours.

We registered how much time people took to conclude each task and the pointed difficulties. In the end they were asked to fill a questionnaire with six statements, which can be found in Appendix A translated to English. For each statement, the questionnaire had a 7-point *Likert scale*, where 1

---

1 *A Likert scale* is the most widely used scale in survey research.
The results and its discussion are presented in Section 5.3.2.2.

Furthermore, three students were asked to try out our CMS, accomplishing the following tasks:

1) Add a new location from scratch, by setting the location boundary points, adding beacons, windows and doors.

2) Create an user group.

3) Add users to the created group.

4) Add media content and assign it to the aforementioned user group.

5) Delete and add new beacons.

6) Modify location attributes (name and orientation).

7) Add products and its corresponding position.

Unfortunately the Position Log feature in the CMS could not be tested by the participants for the just created locations, as the shapes of the spaces consisted of randomly-inserted coordinates. Nevertheless, users experimented this feature for an already inserted location which had logs from past testing. We registered the participant's suggestions and faced difficulties.

5.2.3 Used equipment and Software

Concerning the used equipment and corresponding Software versions, we had our ShopAssist Server and its MySQL Database running on a computer with the specifications described on table 5.1 and 5.2.

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel i7-4700HQ @ 3.40 GHz, 6MB Cache</td>
<td>RAM</td>
</tr>
<tr>
<td>16GB DDR3 1600MHz</td>
<td>Disk</td>
</tr>
<tr>
<td>SSD Samsung 850 PRO @ SATA 6Gb/s</td>
<td>Network Card</td>
</tr>
<tr>
<td>Realtek PCIe Gigabit Ethernet</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: Server Hardware

<table>
<thead>
<tr>
<th></th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows 10 Pro Build 1511</td>
<td>Database</td>
</tr>
<tr>
<td>MySQL v. 5.7.9</td>
<td>Java</td>
</tr>
<tr>
<td>Oracle v. 1.8.0_65</td>
<td>PlayFramework</td>
</tr>
<tr>
<td>v. 2.4.6</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: Server Software
The tests were conducted with the server running on a Internet connection of 1 Gb/s Download and 1 Gb/s Upload.

The Server Load tests (Section 5.3.1) were performed using the online load testing service Loader.io\(^2\).

With respect to the client hardware used, we performed the tests on two different devices: iPad Mini 2 and iPhone 6s. Table 5.3 presents the running software versions on the mobile devices, along with the Estimote equipment versions. All used versions were the latest available at the moment of the testing.

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPad Mini 2 OS</td>
<td>iOS 9.3.1</td>
</tr>
<tr>
<td>iPhone 6s OS</td>
<td>iOS 9.3.1</td>
</tr>
<tr>
<td>Estimote Beacon Firmware</td>
<td>3.2.0</td>
</tr>
<tr>
<td>Estimote SDK:</td>
<td>4.0.0</td>
</tr>
<tr>
<td>Estimote Indoor SDK:</td>
<td>2.0.1</td>
</tr>
</tbody>
</table>

Table 5.3: Software versions of the used equipment

Table 5.4 and 5.5 list the relevant specifications in the scope of our project of the used devices.

<table>
<thead>
<tr>
<th>Equipment</th>
<th>v4.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluetooth chipset</td>
<td>v4.0</td>
</tr>
<tr>
<td>CPU</td>
<td>Dual-core 1.3GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>1GB</td>
</tr>
<tr>
<td>Battery</td>
<td>6470mAh</td>
</tr>
<tr>
<td>Network</td>
<td>Wi-Fi 802.11 a/b/g/n dual-band</td>
</tr>
</tbody>
</table>

Table 5.4: iPad Mini 2 specifications

<table>
<thead>
<tr>
<th>Equipment</th>
<th>v4.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluetooth chipset</td>
<td>v4.2</td>
</tr>
<tr>
<td>CPU</td>
<td>Dual-core 1.84GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>2GB</td>
</tr>
<tr>
<td>Battery</td>
<td>1715mAh</td>
</tr>
<tr>
<td>Network</td>
<td>Wi-Fi 802.11 a/b/g/n/ac dual-band</td>
</tr>
</tbody>
</table>

Table 5.5: iPhone 6s specifications

Wi-Fi was used to provide Internet connectivity both for the iPad and iPhone.

All beacons were placed exactly at 1.20m from floor, which is the average height when we stand the phone in front of us.

### 5.3 Test Results and Discussion

In this section we present the results of the conducted tests and a discussion to achieve valuable conclusions.

\(^2\)https://loader.io/
5.3.1 Server Load Tests

We tested our server under an increasing number of client requests and measured the network, memory and CPU usage of the server. We made six different tests, doubling the amount of requests. We queried the server with 1, 10, 20, 40, 80 and 160 requests per second, during one minute. In other words, these numbers mean client requests made each second. For instance, it lets us know how does our server perform when 160 clients connect every second, over a 1 minute period (totalling 9600 requests on one minute).

The most relevant and complex webservice of our API is the one which returns the location data, for a previous request (beacon Major and Minor). Therefore, it was the chosen webservice for the loading tests and will be named as RetrieveLocation henceforth.

From Figure 5.4 we can recognize that the amount of traffic exchanged increases almost linearly as the number of requests per second increases. Moreover, it is possible to observe that the amount of transmitted data is relatively low (few KB/s), confirming that using JSON for data exchange was a good choice.

Figure 5.4: Network usage of the Server

Figure 5.5 displays the CPU usage of a single-core of the processor. The blue line (Java process) represents the Play Framework which runs in Java, and the black line (MySQL) represent the MySQL Database process. It is observable that as the load increases, the CPU usage also increases, but tending to smooth its growth.

Figure 5.6 depicts the memory usage of the Server (Java process) and its Database (MySQL process), also being queried with a varying number of requests per second. We can observe that the memory usage of both Java and MySQL processes barely fluctuated, allowing us to confirm that our Java code of the Server does not have any memory leak.

Figure 5.7 depicts the response time of the performed tests increasing the number of client requests per second. As response time is the most important metric (due to its impact on customers User Experience) we made nine tests instead of six, also varying the number of requests per second and with a duration of one minute each. We queried the server with 1, 10, 15, 25, 50, 100, 150, 200 and 250 requests per second, and calculated the average response time for each.
From the plot of Fig. 5.7 a linear growth can be observed, as the number of requests increases. As having 250 users starting the ShopAssist Mobile application in the exact same is a scenario that requires the adoption of our system by a large number of users, we can conclude that our server responds in an acceptable time under heavy-load.
5.3.2 Accuracy Tests

In this section we present the results of the accuracy tests performed on two locations: a room and an university office, with two different devices but under the same conditions.

5.3.2.1 Room

Figure 5.8 depicts the results of the accuracy tests made on a room in the form of an histogram, where the frequency distribution of the measured error can be observed.

Tables 5.6 and 5.7 present the real point coordinates within the room, and the average error of the taken measurements. Considering point A the real position and point B the measured position, the error is obtained by calculating the distance between those two points, using the Euclidean distance equation (Eq. 5.1). We repeated this procedure for every point and calculated the average of the measured errors.

\[
EuclideanDistance(a, b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}
\]

<table>
<thead>
<tr>
<th>Real coordinate</th>
<th>(2.0; 1.0)</th>
<th>(2.0; 2.0)</th>
<th>(2.0; 3.0)</th>
<th>(2.0; 4.0)</th>
<th>(2.0; 5.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Error (4 measurements)</td>
<td>0.786</td>
<td>1.288</td>
<td>1.039</td>
<td>1.208</td>
<td>0.509</td>
</tr>
<tr>
<td>Total avg.:</td>
<td>0.966 m</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6: Accuracy test results in a room along Y-axis

In table 5.6 we can observe that the closer to the beacons, the smaller the error achieved. This made us perform accuracy tests in proximity to the beacons, in order to evaluate how the proximity affects the
accuracy.

<table>
<thead>
<tr>
<th>Real coordinate</th>
<th>(1.0; 2.9)</th>
<th>(2.0; 2.9)</th>
<th>(3.0; 2.9)</th>
<th>(3.7; 2.9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Error (3 measurements)</td>
<td>1.302</td>
<td>1.424</td>
<td>1.183</td>
<td>1.458</td>
</tr>
<tr>
<td>Total avg.:</td>
<td>1.342 m</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.7: Accuracy test results in a room along X-axis

From the 32 taken measurements, we estimated a probability interval (quantiles). To do so, the *t*-student distribution was chosen. This distribution is applied when the real average is unknown (only the sample average is known) and with an unknown variance. Therefore, in Table 5.8 we present the calculated values for this sample using a *t*-student distribution, for a confidence interval of 95%.

The *t*-student equation 5.2 gives us an interval for a 95% confidence ($\alpha = 0.05$), where the $s$ parameter is the standard deviation, $\bar{x}$ is the sample mean, $n$ is the number of samples and $\alpha$ is the confidence interval.

$$[\bar{x} - F^{-1}_{t(n-1)}(1 - \alpha/2) \times \frac{s}{\sqrt{n}}, \bar{x} + F^{-1}_{t(n-1)}(1 - \alpha/2) \times \frac{s}{\sqrt{n}}]$$ (5.2)

<table>
<thead>
<tr>
<th>Number of samples</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average error (m)</td>
<td>1.107</td>
</tr>
<tr>
<td>Standard Deviation (m)</td>
<td>0.459</td>
</tr>
<tr>
<td>Standard Error (m)</td>
<td>0.165</td>
</tr>
<tr>
<td>Min (95%)</td>
<td>0.942</td>
</tr>
<tr>
<td>Max (95%)</td>
<td>1.272</td>
</tr>
</tbody>
</table>

Table 5.8: Calculated probability parameters for room measurements

Thus, we can be 95% confident that our accuracy error will fall between 0.942 and 1.272 meters.

Concerning the accuracy tests performed in proximity to beacons, illustrated in Figure 5.9, we obtained a lower average error margin. In Table 5.9 we can observe the *Real coordinate* line which depicts the different beacons coordinates, and the *Avg. Error* line which in turn depicts the average error of the 2 measured values with the device touching the beacon, in a flat position.

---

395% is a reasonable and the most common choice for a confidence level.
Therefore, comparing the obtained average error (0.28 m) with the values obtained from the previous testing (0.97 m and 1.34 m from Tables 5.6 and 5.7 respectively), we can conclude that the closer the device is to a beacon, the better the accuracy. Thus, increasing the beacon density is likely to lead to better results.

<table>
<thead>
<tr>
<th>Real coordinate</th>
<th>(0.0; 1.93)</th>
<th>(0.0; 3.87)</th>
<th>(1.98; 5.50)</th>
<th>(3.97; 3.53)</th>
<th>(3.97; 1.76)</th>
<th>(1.98; 0.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Error (2 measur.)</td>
<td>0.63</td>
<td>0.43</td>
<td>0.14</td>
<td>0.12</td>
<td>0.06</td>
<td>0.28</td>
</tr>
<tr>
<td><strong>Total avg.</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.28 m</td>
</tr>
</tbody>
</table>

Table 5.9: Accuracy test results of measurements taken on a room in proximity to beacons

### 5.3.2.2 University Office

Tables 5.10 and 5.11 show the results of the accuracy tests performed in a university office with two different devices.

<table>
<thead>
<tr>
<th>Real coordinate</th>
<th>(1.5; 1.0)</th>
<th>(1.5; 2.0)</th>
<th>(1.5; 3.0)</th>
<th>(1.5; 4.0)</th>
<th>(1.5; 5.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Error (3 measurements)</td>
<td>0.747</td>
<td>0.507</td>
<td>1.05</td>
<td>0.99</td>
<td>0.849</td>
</tr>
<tr>
<td><strong>Total avg.</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.831 m</td>
</tr>
</tbody>
</table>

Table 5.10: Accuracy test results in an office using an iPad Mini 2

For each device, fifteen samples were collected in the same scenario (three measurements for each marked point). Tables 5.12 and 5.13 present the calculated parameters, also with a *t*-student distribution, for the measurements taken with the iPad and iPhone.
<table>
<thead>
<tr>
<th>Real coordinate</th>
<th>(1.5; 1.0)</th>
<th>(1.5; 2.0)</th>
<th>(1.5; 3.0)</th>
<th>(1.5; 4.0)</th>
<th>(1.5; 5.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. Error (3 measurements)</td>
<td>0.852</td>
<td>0.717</td>
<td>1.09</td>
<td>1.16</td>
<td>1.01</td>
</tr>
<tr>
<td>Total avg.: 0.965 m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.11: Accuracy test results in an office using an iPhone 6s

<table>
<thead>
<tr>
<th>Number of samples</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average error (m)</td>
<td>0.831</td>
</tr>
<tr>
<td>Standard Deviation (m)</td>
<td>0.341</td>
</tr>
<tr>
<td>Standard Error (m)</td>
<td>0.189</td>
</tr>
<tr>
<td>Min (95%)</td>
<td>0.642</td>
</tr>
<tr>
<td>Max (95%)</td>
<td>1.102</td>
</tr>
</tbody>
</table>

Table 5.12: Calculated probability parameters for office measurements made with an iPad Mini 2

<table>
<thead>
<tr>
<th>Number of samples</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average error (m)</td>
<td>0.965</td>
</tr>
<tr>
<td>Standard Deviation (m)</td>
<td>0.338</td>
</tr>
<tr>
<td>Standard Error (m)</td>
<td>0.188</td>
</tr>
<tr>
<td>Min (95%)</td>
<td>0.778</td>
</tr>
<tr>
<td>Max (95%)</td>
<td>1.152</td>
</tr>
</tbody>
</table>

Table 5.13: Calculated probability parameters for office measurements made with an iPhone 6s

Figure 5.10 presents an histogram of the error distribution, measured with both devices. We can observe that the iPad achieved a lower error margin. We were expecting to achieve a better accuracy from the iPhone 6s, as Estimote Indoor Location claims to be better tuned around iPhone’s rather than iPads. But surprisingly, the iPhone reported a lower accuracy comparing to the iPad, under the same test scenario.

Figure 5.10: Result of the accuracy tests in office
5.3.3 ShopAssist Mobile App Resource Usage

Xcode is an IDE developed by Apple for OS X and iOS development. It contains a tool to monitor the resource usage of the developed application, and was the one used by us to measure the efficiency parameters of our mobile application.

Figure 5.11 depicts the CPU usage of the ShopAssist process on the iPad. The initial peak hitting 32% is explained by the initial setup, which communicates with the server and sets the received location, as it has to parse the JSON message through a set of loops (for beacons, windows, doors, and so on). After this peak, an average 6% of CPU usage is obtained. This percentage is explained by the complex algorithms being executed in the background while Estimote Indoor framework is calculating user position. Also, some of this consumption might be explained by our position logger, which reports the user position to the server every second.

![Figure 5.11: CPU Usage of the ShopAssist iOS application](image)

Figure 5.12 shows the Network usage of the ShopAssist application. It is possible to see how the initial exchanged messages use the bandwidth. The first bar is explained by the sent username and password of the login screen, along with a small answer "telling" if the login was successful or not. The second bar is the submission of the Beacon Identification. In the third bar can observed the received JSON message. After this, a minimal upload bandwidth is used to submit periodically the current user position to the server.

![Figure 5.12: Network Usage of the ShopAssist application](image)

In figure 5.13, we can observe the energy impact that our app has on the iPad battery. The greyed-out squares represent the type of resource our application has used. Network represents the network activity occurring in response to our application. Networking brings up radios, which require power for prolonged periods. Cost represents energy use resulting from the work our app performs. Overhead represents energy use as a result of bringing up radios and other system resources required to perform that work. The used Xcode tool reported an high energy consumption from our application, along with a 59% overhead.

We can observe that our app has an high energy demand. This is mainly due to three reasons:

- **CPU Usage** - Our app does complex calculus in order to determine a position, which demands CPU usage.
Network usage - Although we upload minimal amounts of data, we are using the network very often, by reporting our position to the server every second.

Bluetooth usage - Despite the fact that we are using Bluetooth Low-Energy, it still requires energy to operate.

Cell radio is one of the biggest battery drainers of mobile devices. It is possible to reduce power usage of our application by grouping networking activity and communicate it to the server less frequently. This allows the radio of the device to stay on an Idle state more often. This explains the high overhead seen on Figure 5.13, as the radio never switches to an idle state due to the frequent network operations.

The RAM usage was also monitored but found to be minimal, as our app is more CPU and Network-hungry rather than memory-intensive.

5.3.4 User Tests

Below we present the user test results conducted, both to the mobile application and the CMS.
5.3.4.1 Mobile application

As observable in Table 5.14, the first task was the one which took more time to complete, with an average time-to-accomplish of 1 minute and 52 seconds. This value is explained by the fact that users were not familiar with the application and were experiencing their first interaction with it.

The second task achieved a lower average time-to-accomplish, with a value of 1 min and 6 seconds, as the task was simpler and users were getting used to the application. Task three, in turn, resulted in an average 58 seconds of time-to-accomplish.

We can observe on the same table an high Standard Deviation\(^4\). This might be explained by the fact that some participants felt that the system is easier to operate than the other participants.

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average (sec)</td>
<td>112.3</td>
<td>66.1</td>
<td>58.4</td>
</tr>
<tr>
<td>Standard deviation (sec)</td>
<td>22.8</td>
<td>26.2</td>
<td>21.7</td>
</tr>
</tbody>
</table>

Table 5.14: User tests results of task time-of-accomplishment

Users found the mobile application easy to use, but the unanimous reported problem was the low accuracy of the indoor positioning. The avatar on the map was moving in a pattern that did not correspond to users movements, thus confusing them. As indoor positioning is provided by Estimote system, barely nothing can be done about this, except waiting for updates of their framework or eventually increasing the number of beacons to achieve a better precision. Note that although an average error of 1 meter might look a reasonable value, on a space whose width is 3.08 m, it is a considerable amount (see Fig. 5.2).

Three out of seven users suggested to integrate a search bar to browse for products, rather than displaying a long product list. Furthermore, 2 users suggested to display the advertisements also on the application, rather than only in the Digital Signage. Another one suggested to integrate the stock of products, to avoid the situation of a customer browsing for product, walk towards it, and upon arrival observe that is not available.

\(^4\)Standard Deviation is a measure to quantify how much are the values spread, relatively to the average.
Table 5.15 presents the results of the questionnaire filled by the users who used the system. The worst classification concerns Statement 3, which we quote: "I am satisfied with system’ accuracy". This classification could be explained by:

- Users had too high expectations;
- Users were not holding the iPad in flat position (ie., parallel to the floor) as recommended by Estimote to achieve better results;
- Small Test scenario, where an error of 1.5 m is half the width of an office – a significant amount.

The users classified positively our application in terms of simplicity and ease of use (Statements 1 and 4).

Regarding Statement 2 of the questionnaire, "The application starts and locates me indoors quickly", some participants pointed that the application presented a message saying "Looks like you are outside this location". This message is shown when the Estimote Indoor Framework could not deliver position updates correctly, thus displaying that message. Although our server delivered the location data in time and the location was set up in almost instantly in the application, delivering position updates is beyond our reach, as we rely on Estimote. Unfortunately it affected users experience, leading to a delay in locating the users indoors.

<table>
<thead>
<tr>
<th>Statement</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>2.3</td>
<td>3.0</td>
<td>4.9</td>
<td>2.4</td>
<td>2.6</td>
<td>3.0</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>0.9</td>
<td>1.1</td>
<td>1.5</td>
<td>1.2</td>
<td>1.2</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 5.15: User classification of the system in a 7-point scale, where the best scores are the lowest values

5.3.4.2 CMS

The participants which tried the CMS had difficulties inserting a location, as the parameters segment, dist and side are not easy to understand at first glance. Therefore, they suggested to include an explanation for those fields, along with a dropdown for the side attribute rather than a textbox, as only two values are possible (left and right). Furthermore, it was suggested to develop an interactive map, where it would be possible to drag and drop beacons and boundary points. Another suggestion made was to include a live position logger, to observe real-time movements of the customers.

We also identified difficulties regarding the content management feature, as the participants did not understand they had to define a default set of images at first, and additionally the targeted content.

5.4 Costs

In this section we present the project costs and estimate the expense of deploying our platform on a larger space.
Table 5.16 represents the cost of a deployment made in a 36.7 square meter space (Fig. 4.1).

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimote Dev Kit (3 beacons)</td>
<td>100€</td>
<td>4</td>
<td>400€</td>
</tr>
<tr>
<td>Raspberry Pi 3</td>
<td>45€</td>
<td>2</td>
<td>90€</td>
</tr>
<tr>
<td>Samsung S24C450B Monitor</td>
<td>195€</td>
<td>2</td>
<td>390€</td>
</tr>
</tbody>
</table>

**Table 5.16: Effective prototype cost**

To estimate the costs of deployment in a larger scale, we assume a small-sized store. The average size for a small discount store is between 200 - 800 square meters[42]. Hence, we consider an area of approximately 230 sq. meters (18.50 x 12.50 m) as an acceptable value for our cost estimation. Figure 5.15 presents an overview of hypothetical the store, with a window, an entrance and exit door. A total of fifteen beacons were positioned equidistantly whenever possible. The closest beacon to each of the doors is intended to have a Digital Signage associated, hence the proximity to the doors.

Regarding the ideal number of beacons, it varies according to the user experience the business owner wants to create. *i.e.*, the higher the desired number of digital signage, the greater the number of necessary beacons. Besides that, Estimote claims that the higher the number of beacons, the better the accuracy achieved. For longer walls, the recommendation is to position a beacon roughly every 4/5 meters.

As observable in Table 5.17, our cost estimation for a small-sized store is of 980€. We find this value

---


68
<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimote Dev Kit (3 beacons)</td>
<td>100€</td>
<td>5</td>
<td>500€</td>
</tr>
<tr>
<td>Raspberry Pi 3</td>
<td>45€</td>
<td>2</td>
<td>90€</td>
</tr>
<tr>
<td>Samsung S24C450B Monitor</td>
<td>195€</td>
<td>2</td>
<td>390€</td>
</tr>
<tr>
<td><strong>Total cost:</strong></td>
<td></td>
<td></td>
<td><strong>980€</strong></td>
</tr>
</tbody>
</table>

Table 5.17: Cost estimation for an approximately 230 sq. meter store

accessible to many stores, motivating us to continue the path of this work.

### 5.5 Chapter Summary

In this chapter we detailed the testing scenario to evaluate our platform. We performed accuracy tests on different places, with different devices, to observe how the accurateness of the indoor positioning varied. We also run load tests on our server to assess its response time and resources consumption at unusually high loads. Likewise, we measured the resources consumption of our mobile application. Moreover, user tests were carried out to survey the usability of our prototype as a whole and to collect the suggested improvements.

Even with limited resources – reduced number of beacons, two mobile devices – we were able to test our platform, allowing us to undertake extrapolations for more complex scenarios (a larger space, such as a store).

From the load tests performed, we observed that our server responds in an acceptable time period under heavy load.

Furthermore, the accuracy tests carried out resulted on an average error of around 1 meter, which we find acceptable for an emerging technology in the field of indoor localization – the Bluetooth Low-Energy. If the average error remains the same in bigger locations, such as large retail spaces, the impact will be smaller. Nevertheless, it does not offer enough accuracy to find specific products within store shelves.

From the user tests conducted, we observed the main faced difficulties allowing us to understand where the system can be improved, regarding user experience. We encountered significant differences in the time-of-accomplishment of the tasks among users, which explained by the fact that some participants felt the application easier to operate than others.
Chapter 6

Conclusions

6.1 Summary

The goal of this thesis was to develop a system to identify customers and help them locate products in stores, enriching users shopping experience and bringing added value to the business owners. To achieve this, Bluetooth was the chosen underlying indoor location technology, as its 4th version made indoor localization systems cheaper and easier to procure. Despite the fact radio signal propagation indoors is not regular, it has a reasonable accuracy. Therefore, we chose the Estimote platform to provide indoor localization services. This company sells beacons which use Bluetooth Low-energy to provide indoor localization, together with a fully documented SDK.

In this document, the State-of-the-art of indoor positioning systems and corresponding technologies were analysed, as well as the underlying techniques. After evaluating the State-of-the-art, we started outlining a modular architecture that copes with the majority of the addressed problems. The system's architecture has suffered several changes during the development of the prototype, as some hurdles were faced that required us to find alternatives and workarounds.

Our system presents an interface for business owners – the Content Management System – that abstracts business owners from the complexity of deploying a beacon infrastructure to provide indoor localization services. Additionally, our prototype was also designed in tandem with the Digital Signage and Mobile platforms. The first provides users a context-aware shopping experience, with targeted content being shown in the Digital Signage. The latter adds an interactive layer over traditional shopping, allowing users to locate themselves within the store and browse for products in the mobile application.

Presently, each different retail space requires a specific application. Our main goal was to deploy a generic system where Business Owners solely need to provide the corresponding map information along with the desired media content to be displayed in Digital Signage. This way we save the customer from the unpleasantness of installing 10 different apps for 10 different stores. The customer only needs to install one mobile application, which works in multiple stores, thus increasing the chances of adoption of the app.

This document describes the implementation details of both Server, Client and Digital Signage plat-
form, that present a layer over Estimote Framework facilitating the deployment of a context-aware system, mixing advertisements with usefulness (indoor positioning, locating products, targeted advertisements). The opportunities are endless, and some that are worth mentioning were left for Future Work due to time constraints.

Our solution was evaluated using the implemented prototype in a mock up store within the University campus. Overall, we accomplished a good accuracy from our prototype, with error distances hovering around 1 meter. Since we based part of our platform on the Estimote Framework to provide indoor positioning, it is beyond our reach to improve accuracy any further. Nevertheless, we chose to use the model where the location is mapped using the Indoor SDK, as it allows to specify precisely all location parameters, providing better results than using their indoor application.

From the user tests carried out both to the mobile application and the CMS, we observed the main faced difficulties. This allowed us to survey the usability of our prototype as a whole and to understand where the system can be improved regarding user experience.

### 6.2 Future Work

Some features that we would like to provide were left for future work due to time restrictions. We would like to extend out test scenario into a more complex space, ideally a retail shop, in which we could test the real application of our platform.

Some possible ideas for further improvement are:

- Integrate OAuth for customer identification – To unequivocally identify customers within the different stores, Open Authentication is probably a better choice, as it allows users to authenticate with the same credentials they already use on third-party well-known websites. As a result, we avoid the customer registration on our platform.

- Add support for video media content – An animated and more vivid Digital Signage would attract the attention of more customers. Audio could also be considered. A proper study should be conducted to evaluate the impact and usefulness of such rich media content.

- Incorporate map-caching – Allowing an offline navigation mode for previously visited stores would dispense the need for requesting the map every time the customer visits the same store. One has to bear in mind that content might become outdated on mobile device, such as the product list.

- A more efficient way to store media content – Currently media content is stored in the database in a BLOB format, which for the purposes of this prototype it was found to be sufficient. However, in large deployment scenarios, this might lead to an excessively large and non-efficient database. Therefore, we highlight this concern for further improvement;

- Incorporate secure communications – HTTPS provides bidirectional encryption of communications. Hence, using the HTTPS protocol to encrypt the traffic being exchanged between our three
modules is a recommended improvement to prevent man-in-the-middle and eavesdropping attacks. This is specially important due to the sensitivity of the location data being transferred.

- More personalization – Although a unique and generic mobile application is advantageous for customers, in order to increase the chances of adoption by business owners, we would have to provide more personalization and flexibility, *i.e.*, displaying promotions in the mobile phone, in-app purchases, and so on.

After addressing such functionalities, one should start concerning about issues such as scalability and security of our work, even though some security features were taken into account in this prototype – such as the hashing of users password with a salt. But for large deployment scenarios, the scalability of our platform has to be addressed.
Appendix A

A.1 Questionnaire

Users punctuated each statement on a Likert-scale, from 1 to 7, where 1 corresponds to Strongly Agree and 7 corresponds to Strongly Disagree.

1. I found it simple to use the application.

2. The application starts and locates me indoors quickly.

3. I’m satisfied with the systems’ accuracy.

4. It is easy to locate products with the application.

5. The images shown in the Digital Signage are useful.

6. Overall, I’m satisfied with the application.
Bibliography


75
[11] Luo, J., Zhan, X.: Characterization of Smart Phone Received Signal Strength Indication for WLAN Indoor Positioning Accuracy Improvement. Journal of Networks 9(3) (March 2014) 739–746


