Mestrado em Engenharia Informática
Estágio
Relatório Final

Uso de *Bluetooth Smart* e de aplicações móveis para melhorar a experiência de compra dentro de superfícies comerciais

João Silva Aguiar
jaguiar@student.dei.uc.pt

Orientadores:
Professor Doutor Rui Pedro Paiva
Engenheiro Rafael Jegundo

Data: 2 de Setembro de 2015

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Mestrado em Engenharia Informática
Estágio
Relatório Final

Uso de *Bluetooth Smart* e de aplicações móveis para melhorar a experiência de compra dentro de superfícies comerciais

João Silva Aguiar
jaguiar@student.dei.uc.pt

Júri:
Professora Doutora Marília Curado
Professor Doutor Rui Pedro Paiva
Engenheiro Luís Filipe Cordeiro

Data: 2 de Setembro de 2015

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Resumo

Os sistemas de localização têm evoluído ao longo dos anos. Triangulação de antenas, uso de sistema de posicionamento global (*global positioning system* ou GPS) e outras tecnologias têm sido desenvolvidas para indicar, com precisão, a localização do utilizador. O problema destas soluções é a sua ineficácia dentro de edifícios.

Com a proliferação dos *smarthpones*, a localização dentro de edifícios tornou-se ainda mais relevante. Para solucionar este problema, várias tecnologias foram desenvolvidas, aparecendo os *beacons* usando *bluetooth* como uma solução válida e de baixo custo para solucionar este problema.

Nesta dissertação é exposto o desenvolvimento de um sistema que tira partido desta tecnologia para auxílio de clientes de grandes espaços comerciais. Com este propósito tem-se por objetivo criar um sistema capaz de indicar com precisão a localização de determinado produto, bem como indicar um caminho que o utilizador deve percorrer para conseguir recolher todos os produtos presentes na sua lista de compras.

Este documento demonstra o desenvolvimento de uma das aplicação deste sistema, que é responsável por realizar localização dentro de edifício, com uma margem de erro média de aproximadamente 1 metro, e capaz de gerar o percurso mais curto, com uma margem de erro média de 9.01% em relação ao percurso ótimo.

Com esta dissertação pode-se concluir que é possível criar um sistema de baixo custo suficientemente eficaz para guiar um utilizador dentro de um espaço fechado.

Abstract

Location systems have evolved over the years. Triangulation of GSM signals, Global Positioning System and other technologies had been developed in order to indicate accurately the location of the user. The problem with these kind of solutions is their ineffectiveness inside buildings.

With the proliferation of smartphones, the indoor location has become even more important. To solve this problem, various technologies have been developed as the bluetooth beacons. This technology is a valid and cost-effective solution to solve this problem.

In this dissertation is exposed the development of a system that takes advantage of bluetooth beacons in order to Enhanced Shopping Experience. For this purpose it was designed a system capable of accurately indicate the location of a particular product and generate and indicate the path that the user should follow in order to collect all the products present on user's shopping list.

This document shows the development of one of the applications of this system, responsible for estimate the location of the user within the building, with an average error of near 1 meter and is able to generate the shortest route with a average margin of error of 9.01% over the best route.

With this dissertation can be concluded that it is possible to create a effective low cost system to guide a user inside a building.

Agradecimentos

Este trabalho foi desenvolvido com a orientação do Professor Doutor Rui Pedro Paiva e pelo Engenheiro Rafael Jegundo, a quem gostaria de expressar os meus mais sinceros agradecimentos, pela presença e apoio dados. Gostaria também de deixar uma palavra de apreço ao Professor Doutor Luís Paquete, que apesar não estar ligado a este projeto auxiliou-me numa fase importante deste trabalho.

À Whitesmith por contribuído para minha formação profissional, apostando em mim e por mais tarde darem-me a oportunidade de realizar este estágio.

Aos meus amigos e colegas, que me brindaram e apoiaram ao longo destes anos. Em particular gostaria de agradecer a Goncalo Dias, Pedro Carmona e Luís Ramos por terem sempre presentes para todas as ocasiões.

E aos meus pais, pelo apoio, compreensão e carinho incondicional que sempre me deram.

Índice

Lista de Tabelas

Lista de Figuras

Glossário

Beacon Um *Beacon* é um dispositivo que eletrónico que emite um sinal periódico ou continuo. O conceito de *beacon* é usado em diferentes contextos na área de informática, sendo normalmente usados em sistemas de posicionamento, sendo que cada *beacon* se encontra num ponto conhecido (ponto âncora), o que permite estimar a posição do recetor..

Internet of Things A International Telecommunication Union define a *Internet of things* como sendo uma infraestrutura global para a sociedade da informação, permitindo serviços avançados através da interligação (física e virtual) de coisas baseadas na interoperabilidade existente e em evolução da informação e comunicação. [7] .

UI Kit A UI Kit é uma *framework* que permite auxiliar o programador a desenhar *views* e as reagir às suas ações. Esta *framework* é composta por vários tipos de componentes (*Views*) para fins específicos, como a UITextField no iOS, que permite ao utilizador introduzir texto, ou a UIWebView que é capaz de interpretar código HTML, CSS e JavaScript gerado através de um servidor externo ou no próprio dispositivo. Cada uma destas *views* é capaz de reagir a eventos do utilizador, contendo respostas predefinidas, como mostrar automaticamente o teclado, caso essa *view* permita a introdução de texto. Caso pretendido, tanto os eventos como o *layout* de cada *view* podem ser personalizados.

UIView UIView é uma *view*, que pode conter vários tipos de subviews. O utilizador é capaz de interagir com esta, sendo que cada interação gera um evento que podem ser tratado pelo programador.

Lista de Abreviaturas

AP *Access Point*.

API *Application Programming Interface*.

BLE *Bluetooth Smart* ou *Bluetooth Low Energy*.

DNS *Domain Name System*.

GPS *Global Positioning System* ou Sistema de Posicionamento Global.

IOT *Internet of Things*.

IRR *Inquiry Response Rate*.

JSON *JavaScript Object Notation*.

LQ *Link Quality*.

LSD *Lean Software Development*.

NFC *Near Field Communication*.

RSSI *Received Signal Strength Indication*.

SDK *Software Development Kit* ou Kit de Desenvolvimento de Software.

TI tecnologia de informação.

UI *User Interface*.

UID *Unique identifier*.

URL *Uniform resource locator*.

VCS *Version Control System*.

XML *eXtensible Markup Language*.

Capítulo 1

Introdução

Este documento expõe o estudo e desenvolvimento do projeto proposto pela empresa Whitesmith, Lda que se enquadrou com a realização do estágio curricular do Mestrado em Engenharia Informática. Este foi supervisionado pelo Mestre Rafael Jegundo, CEO da Whitesmith, Lda, e pelo Professor Doutor Rui Pedro Paiva.

Este capítulo introdutório está dividido em quatro secções. Na primeira secção é exposta uma contextualização do trabalho. Na segunda secção é explicado o propósito da realização deste projeto. Na terceira é realizada uma síntese dos objetivos e da abordagem ao desafio. No quarta secção é realizada uma síntese das contribuições do principais deste trabalho. Por fim, na última secção é explicado o propósito dos capítulos seguintes deste documento.

1.1 Contextualização

Apesar de não haver uma definição concreta de *smartphone* podemos considerar que é um dispositivo capaz de realizar o mesmo que um telemóvel, como por exemplo realizar chamadas, mas também realizar funções avançadas que apenas computadores eram capazes de o fazer [8][9]. No fundo, são telemóveis com elevado poder de processamento e com sistemas operativos complexos.

Ao longo dos últimos anos estes dispositivos tiveram uma grande evolução. Foram aumentando cada vez mais o seu poder de processamento, foram adquirindo novas funcionalidades e substituindo outros equipamentos eletrónicos (máquinas pessoais fotográficas e de filmar, sistemas de navegação, entre outros), tornando-se cada vez mais num dispositivo multiusos. Por sua vez as redes móveis foram acompanhando as necessidades, evoluções e desafios destes dispositivos aumentando cada vez mais as velocidades de acesso à Internet, tornando-os nos dispositivos mais utilizados pelas pessoas para a utilização desta em vários países [10] [11]. Uma vez que os sistemas operativos destes dispositivos estão abertos a programadores externos, milhares de aplicações são desenvolvidas anualmente, para todo o tipo de funções [12]. Com isto, estes dispositivos foram adquirindo cada vez mais consumidores, estando atualmente presentes na Europa e nos Estados Unidos da América, em 3/5 da população.[13][14] É portanto um mercado muito importante na área da tecnologia de informação (TI). Para as empresas produtoras de dispositivos eletrónicos, estas tentam entrar ou aumentar o seu mercado. Para as empresas de software compete-lhes diferenciar as suas aplicações das inumeras atualmente existentes.

1.1.1 Aplicações Móveis *In-Store*

Segundo a Google Shopper Marketing Agency Council, 79% dos utilizadores de *smartphones* nos Estados Unidos da América utilizam-nos para auxiliar as compras que realizam, o que eles designam por *smartphone shoppers*. Desse grupo, 84% utilizam os seus dispositivos dentro das lojas para auxílio das suas compras [15]. As superfícies comerciais podem tirar proveito desta utilização de forma melhorar a experiência dos clientes dentro das suas lojas. Aplicações deste tipo para além de ajudarem a melhorar as vendas, aumentam a relação entre o consumidor e a marca, tornando-os assim mais fiéis [16].

1.2 Motivação

Com a introdução dos *smartphones* foram criados novos paradigmas no comércio. Instituiu-se a cultura "There's an App for That", isto é, os utilizadores começaram a usar e preferir aplicações em vez de *websites*. Se até então era essencial para uma marca ter um *website*, agora têm de se adaptar e abraçar esta nova plataforma. Devido ao facto de milhões de aplicações estarem disponíveis atualmente, é necessário criar aplicações que se diferenciem, sendo inovadoras e indo de encontro às necessidades do consumidor.

O facto dos utilizadores de *smartphones* já estarem habituados a usar o seu pequeno computador dentro de superfícies comerciais deve ser canalizado para melhorar a sua experiência. Estudos indicam que 87% destes utilizadores gostariam de poder aceder ao mapa destas superfícies de forma a auxiliá-los a localizar produtos [17].

Observando as aplicações das grandes superfícies comerciais atualmente exis-

tentes, são poucas as que permitem realizar esta tarefa. Por isso, a Whitesmith e o estagiário decidiram abraçar o desafio destes utilizadores desenvolvendo um sistema que, além de indicar a posição dos produtos, é capaz de indicar como chegar até eles, bem como criar um percurso ótimo, baseado na lista de compras de cada utilizador, que minimize o tempo despendido nas superfícies de elevada dimensão.

1.3 Objetivos e Abordagens

Os objetivos deste trabalho podem ser divididos em três partes importantes:

- Estágio, cujo objetivo principal é o ganho de experiência e conhecimento por parte do estagiário;
- Empresa, cujo objetivo principal é validar as potencialidades do *Bluetooth Smart* ou *Bluetooth Low Energy* (BLE);
- Projeto, que corresponde à implementação de um sistema que auxilie os clientes dos hipermercados.

Estágio

Do ponto de vista do estágio, os objetivos passam pela consolidação dos conhecimentos de engenharia de software, a capacidade de enfrentar e resolver problemas/desafios e aprender e dominar a programação de aplicações para o sistema operativo móvel iOS.

Empresa

A Whitesmith foca os seus recursos na área da *Internet of Things* (IOT), considerando assim importante validar as potencialidades do BLE. Como empresa, é do seu interesse criar um produto que seja uma mais valia no mercado, o que levou à alteração do projeto inicial. Essa alteração permitiu a criação de um produto mais diferenciador no mercado do que o inicialmente proposto.

Projeto

Do ponto de vista do projeto, o objetivo é criar um sistema que auxilie os clientes dos hipermercados na realização das suas compras. Para isso, tira-se partido de uma tecnologia recente, mais concretamente o BLE que, indiretamente, permite a localização dentro de edifícios.

Este sistema é constituído por 3 aplicações móveis:

- Aplicação de gestão de lista de compras. Esta aplicação tem o objetivo de permitir ao utilizador a criação da sua lista de compras;
- Aplicação de guia dos clientes, que será usada pelos clientes dentro da superfície comercial. Permite indicar o local onde se encontram os produtos que estão na lista de compras, sendo capaz de criar um percurso de forma a minimiza-lo;
- Aplicação dos funcionários. Durante a reposição do stock, os funcionários deverão usar esta aplicação para indicar ao sistema o local onde cada produto se encontra.

Sendo este um sistema complexo, o estagiário é apenas responsável pela aplicação de guia dos clientes, devido ao facto de ser a aplicação fulcral e mais complexa deste sistema. Foi pedido ao estagiário para ser generalista na sua implementação, de forma a ser facilmente adaptável a outros contextos.

1.4 Contribuições Principais

Este projeto demonstra que é possível criar sistemas de localização dentro de edifícios, suficientemente eficazes para guiar utilizadores dentro de um local fechado, usando tecnologias atuais de baixo custo.

Neste documento é descrito como foi possível estimar a posição do utilizador com uma margem de erro de 1 metro usando BLE, validando esta tecnologia como uma boa solução neste contexto.
De forma a tentar minimizar os problemas resultantes desta margem de erro é demonstrado um mecanismo que tira partido do mapa da superfície de forma a garantir a validade da posição estimada do utilizador.

É também demonstrado um processo capaz de gerar o percurso mais curto em tempo real. Este percurso tem algumas limitações, devido á elevada complexidade de gerar um percurso ótimo. Por esta razão é mostrado um mecanismo, que gera o percurso mais curto com base na heurística, sendo este em média 9.01% maior que o percurso ótimo.

1.5 Estrutura do Documento

O documento está dividido nos seguintes capítulos:

Introdução No primeiro capítulo, foram descritos o contexto, a motivação e os objetivos do estágio;

Estado da Arte No segundo capítulo, são estudadas as aplicações das superfícies comerciais atualmente existentes no mercado, sistemas semelhantes ao proposto e tecnologias para localização dentro de edifícios;

Metodologia e Planeamento No terceiro capítulo, é descrita a metodologia usada neste projeto e o planeamento deste;

Requisitos No quarto capítulo são definidos os requisitos funcionais e não funcionais do sistema;

Arquitetura No quinto capítulo, é mostrada e defendida a arquitetura do sistema;

Implementação No sexto capítulo, é documentada a implementação e os testes realizados;

Conclusão No último capítulo, é realizada uma síntese e análise critica deste documento.

Capítulo 2

Estado da Arte

2.1 Estado atual das aplicações dos hipermercados

Para validar o valor de mercado que o sistema possa vir a ter, foram analisadas 13 aplicações de grandes superfícies (ver anexo A).
Neste capítulo são destacadas três dessas aplicações, a do Continente, Walmart e Tesco Groceries, apresenta-se uma tabela de resumo das aplicações analisadas e faz-se uma breve reflexão sobre elas. Foram destacadas estas aplicações uma vez que conseguem generalizar as aplicações atualmente existentes no mercado.

Os dados do número de *downloads* provêm do serviço xyo.net, que estima este valor, e os do nível de satisfação provêm da Apple Store e Google Play (informação extraída em Novembro de 2014).

Continente

O Continente é um dos maiores hipermercados portugueses que pertence ao grupo Sonae. O principal foco desta aplicação é o auxílio dos seus mais de três milhões de utilizadores que têm um cartão da marca [18]. As principais funcionalidades são:

- Visualização do valor que o cliente tem em cartão, cupões de desconto e últimas compras efetuadas (funcionalidades exclusivas para clientes com cartão de cliente);
- Criação de listas de compras;
- Pesquisa de um produto pelo seu código de barras;
- Alguns detalhes de produtos;
- Visualizar os folhetos do estabelecimento;
- Pesquisa do estabelecimento mais próximo do utilizador.

Número de *downloads*:

- 192 mil no sistema operativo Android;
- 55 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5 (versão 2.1.11)

Satisfação na Google Play: 3.9 em 5

Tesco Groceries

Tesco Groceries é uma aplicação mobile que pertence à Tesco Stores Ltd. sediada no Reino Unido. Foi desenvolvida para ser usada nas mais de 3300 lojas presentes no Reino Unido [19]. Enquanto a aplicação do Continente está mais orientada para os clientes com cartões da marca, esta aplicação está mais otimizada para compras online. As suas principais funcionalidades são:

- Pesquisa pelo estabelecimento mais próximo (apenas através de código postal);
- Mostrar as compras habituais;
- Pesquisa de produtos por código de barras;
- Envio dos produtos para a morada ou recolha dos produtos pelo cliente.

Número de *downloads*:

- 2 milhões no sistema operativo Android;
- 658 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5 (versão 7.5)

Satisfação na Google Play: 3.5 em 5

Walmart

A Wal-Mart Stores, Inc opera nos Estados Unidos da América como Walmart. A aplicação tem um foco especial nas promoções existentes nos seus estabelecimentos. As suas principais funcionalidades são:

- Pesquisa pelo estabelecimento mais próximo;
- Pesquisa de produtos por código de barras e qrcode;
- Variados detalhes dos produtos tais como os seus ingredientes e índices calóricos;
- Indicação do número da prateleira em que o produto se encontra.

Número de *downloads*:

- 14 milhões no sistema operativo Android;
- 5,2 milhões no sistema operativo iOS.

Satisfação na Apple Store: 4,5 em 5 (versão 5.1)
Satisfação na Google Play: 4,3 em 5

Uma vez que estas aplicações são de empresas que estão no mesmo tipo de mercado, é interessante analisar o facto de como elas tiram proveito das suas aplicações. O continente dá especial destaque aos clientes com cartões de fidelização, ao contrário da Tesco que utiliza a sua aplicação para dar a possibilidade de fazer compras através de um *smartphone*.

A aplicação Walmart é interessante pois consegue fazer um bom compromisso entre as duas vertentes. Ajuda os seus clientes indicando a prateleira onde os produtos se encontram e permite ao mesmo tempo realizar compras online.

Na análise feita às 13 aplicações no anexo A foram analisados os seguintes campos:

- Lista de compras, verifica se a aplicação tem capacidade de guardar uma lista de compras;
- Cartão de fidelização, verifica se é capaz de substituir e complementar os cartões de fidelização;
- Compras online, verifica se é possível a realização de compras online;
- Promoções, analisa a facilidade de acesso a produtos em promoção;
- Compras habituais, analisa se a aplicação é capaz de guardar as compras habituais;
- Detalhes de produtos, analisa a capacidade de pesquisa de produtos e seus detalhes (como o preço e índices nutricionais);
- Loja mais próxima, analisa a capacidade da aplicação indicar ao utilizador qual a loja mais próxima;
- Localizador de produtos, analisa a capacidade da aplicação em indicar ao utilizador onde se encontra determinado produto.

Esta análise originou a tabela 2.1. Observa-se que apenas duas permitem localizar produtos dentro da loja, indicando apenas a prateleira onde o produto se encontra.

	Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localização de produtos
Continete	✓	✓	✗	Parcial	✗	Parcial	✓	✗
Auchan	✓	✗	✓	✓	✗	✓	✓	✗
Continete	✓	✓	✗	Parcial	✗	Parcial	✓	✗
Intermarché	✓	✗	✗	✓	✗	✗	✓	✗
Tesco Groceries	✗	✓	✓	✓	✓	✓	Parcial	✗
Walmart	✗	N.A.	✓	✓	✗	✓	✓	Parcial
ASDA	✓	N.A.	✓	✓	✗	✓	✓	✗
Sainsbury's	✗	✓	✓	✓	✗	✓	✓	✗
Lidl	✗	N.A.	✗	✓	✗	Parcial	Parcial	✗
Kroger Co.	✗	✓	✓	✓	✗	✓	✓	✗
Costco	✓	✓	✓	✓	✗	✗	✓	✗
fnac.pt	✗	✗	✓	✓	✗	✓	✓	✗
Ikea	✓	✗	✗	Parcial	✗	✓	✓	Parcial
Toys "R" Us Shoping	*Wish List*	✗	✓	✓	✗	Parcial	✓	✗

N.A - Não Avaliado (A empresa não possui cartões de fidelização);
Parcial - Consegue satisfazer parcialmente o campo analisado;
✓- É capaz de realizar o campo analisado;
✗- Não é capaz de realizar o campo analisado.

Tabela 2.1: Estado da arte das aplicações de grandes superfícies

Em entrevista para a Fortune, o presidente da Toys 'R' Us disse "O que nós aprendemos com os nossos clientes é que eles precisam de ajuda na compra de presentes. Não querem entrar numa loja de 200.000 metros quadrados, onde não conseguem encontrar o caminho de volta, eles querem que nós o tornemos mais fácil" [20].

Uma vez que nenhuma destas grandes superfícies tem um sistema de localização dentro dos seus estabelecimentos e estes sabem da importância deles, o sistema proposto pode ser uma mais valia no mercado.

2.2 Algoritmos de Localização

Nesta secção são apresentados várias algoritmos usados para estimar a localização de um dispositivo.

2.2.1 Lateração ou *Lateration*

A lateração é um dos algoritmos mais antigos e usados para determinar/estimar a localização de um dispositivo. Como requisito desde algoritmo é necessária a distância entre o dispositivo e, no mínimo, três pontos de conhecimento prévio (pontos âncora), sendo estes os pontos onde os *beacons* estão posicionados. Esta distância é habitualmente induzida a partir do parâmetro *Received Signal Strength Indication* (RSSI), criando uma relação entre a força do sinal com a distância a que o utilizador se encontra [1] [21] [22].

O funcionamento do algoritmo inicia-se com a uso da equação da circunferência 2.1. São criadas circunferências com centro nos pontos âncora (pontos conhecidos), e com o raio de valor igual à distância previamente calculada a esse ponto.

$$(x - x_i)^2 + (y - y_i)^2 = r_i^2 \tag{2.1}$$

Os parâmetros x_i e y_i devem ser substituídos pelas posições dos pontos âncora, e r_i pelas distâncias previamente calculadas a cada um. Como descrito anteriormente, serão necessários, no mínimo, três pontos e distâncias a ele para garantir o funcionamento deste algoritmo. Assim sendo, tem-se um sistema com, no mínimo, 3 equações. O resultado do sistema de equações é o valor da interceção dos círculos que, por sua vez, é o valor da posição do *smartphone* (figura 2.1).

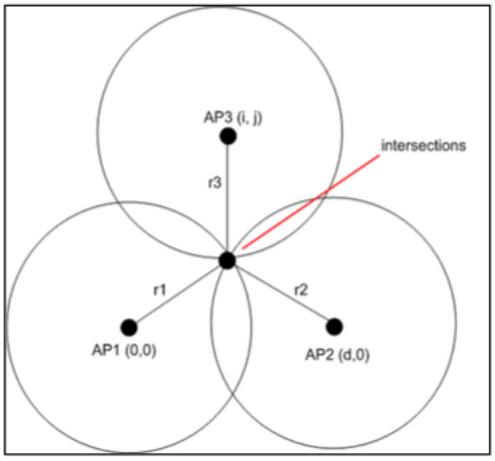


Figura 2.1: Funcionamento do método Lateração [1]

2.2.2 *Cell Based*

Este método funciona a partir dos *beacons* visíveis para determinar a posição do dispositivo. Para isso são usados vários *beacons* com curto alcance de forma a que cada um cubra uma área específica. Assim, um utilizador em movimento, irá encontrar áreas cobertas por diferentes *beacons* num curto espaço de tempo.

O funcionamento deste método começa com uma fase de treino, onde é feito um ”scan“ dos *beacons* visíveis em vários segmentos da superfície sendo esta informação guardada numa base de dados. A descoberta da posição do utilizador inicia-se com a procura de todos os *beacons* visíveis em determinado ponto, sendo guardado o seu *Unique identifier* (UID). Estes são depois comparados com os valores guardados na base de dados anteriormente sendo a melhor correspondência selecionada como a posição do *smartphone*. (figura 2.2)

Nesta abordagem, os parâmetros do sinal podem ser utilizados para desambiguar segmentos que sejam abrangidos pelo mesmos *beacons* [2][23].

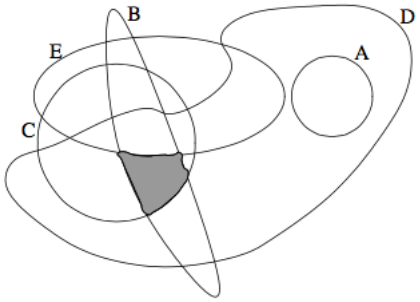


Figura 2.2: Um Recetor que está no alcance dos *beacons* B,C e D mas não de A e E deve estar localizado na região sombreada[2]

2.2.3 *Fingerprinting*

Nesta abordagem o mapa é dividido em vários segmentos ou grelhas. Nesta abordagem são igualmente usadas duas fases: *online* ou treino e *offline*. Esta abordagem inicia-se com a fase de treino, onde é associado a cada segmento atributos únicos. No contexto das redes sem fios estes atributos podem ser o RSSI, *Inquiry Response Rate* (IRR) ou *Link Quality* (LQ). Estes são medidos e atribuídos a cada segmento. Por exemplo, aplicando ao sistema proposto, poderíamos usar o RSSI como atributo e fazer uma média dos valores lidos em cada segmento. Cada *fingerprint* é constituída por várias médias dos valores de RSSI e da posição do segmento no mapa, guardando-a numa base de dados.

Na segunda fase, fase *online*, é onde é estimada a posição do *smartphone*. Para isso, este dispositivo deve recolher a informação do(s) mesmo(s) parâmetro(s) analisados na fase de treino e compará-los com os valores recolhidos. A posição será a associada à *fingerprint* com melhor correspondência aos valores lidos naquele momento [24] [25].

2.2.4 Angulação

Nos métodos baseados na angulação, a posição de determinado objeto é induzida tirando partido da geometria.

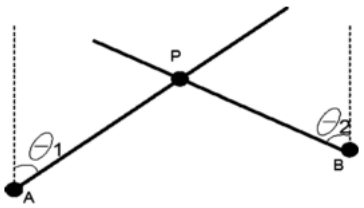


Figura 2.3: Posicionamento baseado na angulação [3]

Como se pode observar na Figura 2.3, são necessários pelo menos dois pontos âncora, representados como A e B, e a medição dos ângulos entre o objeto e estes pontos (θ_1 e θ_2). Com estes valores é possível calcular a posição deste objeto usando as equações 2.2 e 2.3 [26].

De forma a permitir estimar os ângulos necessários para o funcionamento deste algoritmo, são usualmente usadas antenas direcionais.

$$\begin{aligned} tan(\theta_1) &= \frac{y - y_1}{x - x_1} \\ tan(\theta_2) &= \frac{y - y_2}{x - x_2} \end{aligned} \tag{2.2}$$

$$y_i - x_i tan(\theta_i) = y - x tan(\theta_i) \tag{2.3}$$

2.2.5 Dead reckoning

Nesta abordagem a descoberta da posição do utilizador é efetuada a partir da posição previamente calculada. Sabendo o ponto anterior, a distância percorrida e o ângulo do movimento, conseguimos calcular a posição atual (Observar equações 2.4 e figura 2.4). [4]

$$\begin{aligned} x_1 &= d_1 cos(\theta) \\ y_1 &= d_1 sin(\theta) \\ x_2 &= x_1 + d_1 cos(\alpha) \\ y_2 &= y_1 + d_1 sin(\alpha) \end{aligned} \tag{2.4}$$

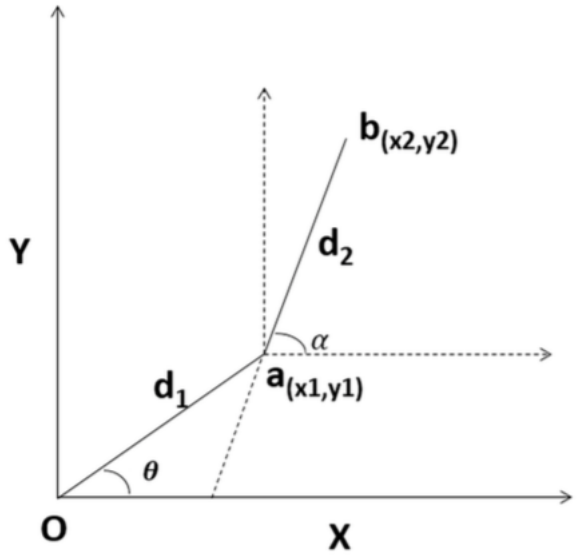


Figura 2.4: Funcionamento do algoritmo *Dead reckoning* [4]

O problema desta abordagem são os erros acumulativo, isto é, sempre que uma variável (ângulo ou distância) for mal calculada, o erro vai-se propagando ao longo do algoritmo. Para colmatar esta falha, devem ser inseridos vários

pontos de referência, de forma a reiniciar o algoritmo.

2.3 Tecnologias capazes de localizar *smartphones* dentro de edifícios

Para ser possível indicar qual o percurso que o cliente deve percorrer necessitamos de saber qual a sua localização exata em todo os os momentos. Visto tratarem-se de grandes superfícies e estas estarem cobertas, o uso de *Global Positioning System* ou Sistema de Posicionamento Global (GPS) não é uma solução. Será por isso feita um breve descrição das tecnologias que estão atualmente presentes nos *smartphones* que podem permitir realizar esta tarefa.

2.3.1 Magnetómetro

O magnetómetro é um dos sensores atualmente presentes nos *smarthpones*. Este é normalmente usado para indicar a direção para que o dispositivo está virado.

Dentro de edifícios existem oscilações nas leituras deste sensor, essencialmente porque a construção dos edifícios é à base de betão e ferro, o que interfere com a precisão desta tecnologia. Assim, alguns investigadores [27] [28], propuseram aproveitar estas anomalias para realizar localização dentro de edifícios. Para isso é usado um algoritmo de *fingerprint* onde se mapeia estas anomalias usando-as para estimar a posição do utilizador.

O problema desta abordagem é a influência de outros objetos eletrónicos ou com base em ferro produzem, que por sua vez aumentam as margens de erro

desta solução. Com esta abordagem é possível estimar a localização do utilizador com uma margem de erro média de 4.7 metros [27].

2.3.2 *Near Field Communication* (NFC)

O *Near Field Communication* (NFC) é uma tecnologia sem fios de ligação de curto alcance (tipicamente 4 cm[29]) de baixo custo. Esta tecnologia permite troca de informação entre dois dispositivos ou entre um dispositivo e uma etiqueta (*tag*) NFC. É atualmente usada para variadas aplicações, como a realização de pagamentos, identificação ou partilha de informação. Opera na gama dos 13.56 MHz com velocidades de transmissão até 424 kbps.

Alguns investigadores implementaram um sistema de localização simples e de baixo custo [30][31]. Para o funcionamento destes sistemas, são inicialmente espalhadas várias etiquetas NFC dentro do edifício em posições específicas. O utilizador quando chega perto de uma, deve aproximar o telemóvel dela, sendo esta interpretada pela aplicação. Através desta informação, a aplicação é capaz de indicar onde o utilizador se encontra, e indica para onde o utilizador deve prosseguir. No final dessa rota, encontrará outra tag e o processo anterior é repetido até chegar ao ponto pretendido. Desta forma a margem de erro da posicionamento da pessoa no local é nula.

Uma limitação desta tecnologia é que nem todos os sistemas operativos móveis permitem tirar proveito desta tecnologia como o caso do iOS.

2.3.3 Rede de telecomunicações

A função básica de um telemóvel é a capacidade de realizar e receber chamadas. Para isso estabelece uma ligação contínua com a célula através de ondas eletromagnéticas. Quando um telemóvel é ligado este interceta a onda e inicia a comunicação, sendo esta ligação multidirecional que é mantida com a célula. Uma vez que cada célula tem um número de ligações limitadas, em zonas com elevada densidade de utilização, várias células são instaladas.

Um exemplo do uso desta tecnologia para estimar a posição do utilizador dentro de edifícios foi proposto no artigo "Accurate GSM Indoor Localization" [32]. Para isso foi usado um algoritmo de *fingerprinting*, onde são mapeadas as 6 células mais fortes em cada 1.5 metros. Segundo o autor, este teve uma margem de erro média de aproximadamente 5 metros.

Apesar de ser possível utilizar este método no Android, no iOS não é possível aceder á informação da força do sinal das redes de telecomunicações, não sendo portanto possível utilizar este método neste sistema operativo.

2.3.4 Giroscópio e Acelerómetro

O giroscópio e o acelerómetro são normalmente usados em conjunto apesar de terem funções distintas. O giroscópio permite-nos indicar qual a inclinação do *smartphone* segundo o eixo dos x,y e z. O acelerómetro serve para medir a aceleração não gravitacional, de cada movimento.

Trein, Singh e Maddila realizaram um sistema de localização dentro de edifícios tirando partido destes dois sensores [4]. Estes usaram um algoritmo *dead*

reckoning que, como descrito no capítulo anterior, necessita da distância percorrida, e do ângulo em que esta distância foi efetuada.
Este ângulo é inferido a partir do valor medido pelo magnetómetro que é validado com o valores do giroscópio. Se ambos os valores coincidirem este ângulo é usado, caso contrário é ignorado, sendo usado o último valor validado.
A distância é inferida a partir de cada passo dado. Para isso os autores começam por identificar cada passo usando uma fase treino. Com este intuito, uma pessoa percorre determinado percurso e conta o número de passos, sendo a informação do acelerómetro guardada. Esta informação é depois analisada de forma a ser possível indicar inequivocamente cada um destes passos dados. Um valor da distância percorrida por cada passo é definido previamente, de acordo com a altura do utilizador. Desta forma são induzidas a distância e o ângulo necessários para o algoritmo *dead reckoning*.

Os resultados desta experiência mostraram que é possível identificar os passos de cada pessoa com uma taxa de erro de 1.055% e saber o ângulo de cada passo com uma margem de erro de aproximadamente 20º. Mas o algoritmo não foi capaz de identificar o caminho total percorrido por cada utilizador. Apesar de não ser indicada qual a margem de erro real desta abordagem, é afirmada a necessidade de outras tecnologias como *Bluetooth*, para ultrapassar limitações desta abordagem.

Ladetto e Merminod criaram um sistema semelhante tendo os seus resultados mostrado uma margem erro média de aproximadamente 10 metros [33].

2.3.5 Sensor de Imagem

Atualmente todos os *smartphones* estão equipados com sensores de imagem (câmaras). Por esta razão existe o interesse de usar esta tecnologia para estimar a posição do utilizador dentro de edifícios.

Um dos sistemas promissores é o uso de comunicação de luz visível. Essencialmente utilizam as frequências das lâmpadas para transmitir informação que, apesar de o olho humano não ser capaz de as observar, estes sensores são. É assim usada uma aplicação no telemóvel capaz de interpretar esta informação, que é essencialmente um ID da lâmpada, para ser usado como ponto âncora. É também analisado o ângulo que a câmara faz com luz emissora. Com estes dados, é aplicado um método baseado na angulação para estimar a posição do utilizador, sendo possível uma margem de erro média de 1.5 metros [34].

2.3.6 Wi-Fi e Bluetooth

Atualmente todos os *smartphones* têm placas de Bluetooth e Wi-fi, sendo por isso uma tecnologia apreciável para tentar resolver o problema de localização dentro de edifícios. Neste âmbito, vários sistemas foram desenvolvidos usando Wi-FI [35][36][37][1] e usando Bluetooth [38][39][40][2][23]. O seu funcionamento passa por espalhar estes dispositivos no espaço, criando uma malha de rede. Após isto, valores como a força do sinal (RSSI) são usadas para identificar a posição do utilizador. O problema de ambas tecnologias, é a variação do sinal, isto é, estando exatamente no mesmo ponto vários valores RSSI são lidos. Outro problema é a refração, reflexão e absorção do sinal, sendo esta agravada dentro de edifícios devido ao facto de existirem múltiplos objetos

entre o emissor e o recetor. Formas de minimizar o erro produzido por estes fatores é ainda uma área em estudo.

Wi-Fi

O IEEE 802.11, vulgarmente conhecido por Wi-Fi, ao longo dos anos foi substituindo a necessidade de ligar um cabo Ethernet aos nossos dispositivos eletrónicos. Atualmente os *smartphones* estão equipados com placas compatíveis com protocolo 802.11n, sendo que os mais atuais já cumprem a norma 802.11ac. A primeira norma opera no espectro de 2.4GHz e/ou 5GHz sendo que a segunda apenas opera no de 5GHz. Estas evoluções procuraram o aumento do alcance, velocidade de transferência e eficiência energética.

Apesar de ser uma tecnologia banalizada, atualmente presente na maioria dos edifícios, para a implementação de um sistema de localização, é necessário colocar vários *Access Point* (AP) em posições estratégicas. Usando esta tecnologia é expectável um erro entre 2-6 metros dependendo do algoritmo utilizado [3].

Ao contrário do Android, existe a limitação que no iOS não é possível aceder informações das redes Wi-Fi envolventes, sendo apenas possível usando uma API privada, infringindo assim uma regra da Apple Store, levando a que uma possível aplicação seja rejeitada. Para ultrapassar esta limitação, deverá ser o AP a estimar a posição do utilizador.

Bluetooth

O Bluetooth é atualmente usado para a criação de redes pessoais, usualmente para troca de informação. O Bluetooth usa a espectro de frequência 2.4GHz. A

sua ultima versão, 4.0, conhecida como Bluetooth Low Energy (BLE) ou Bluetooth Smart, distingue-se essencialmente pelas melhorias a nível energético. Esta nova versão consome entre 1/2 a 1/100 em comparação com o anterior versão [41]. Isto permite a dispositivos que utilizem esta tecnologia terem uma autonomia de anos quando alimentadas por pilhas usualmente presentes nos relógios de pulso [42].

Como referido anteriormente, para o uso desta tecnologia para estimar a posição do utilizador são espalhados *Beacon*s que usam *bluetooth* pelo espaço. Com esta tecnologia é expectável uma margem de erro média entre 2-4 metros. [40][43]

Sendo esta uma tecnologia barata e eficaz para o efeito, houve o interesse de criar um produto comercial, mas ao mesmo tempo era necessário que os sistemas operativos móveis integrassem-se com esse produto.

A Apple decidiu então desenvolver o seu protocolo de comunicação, que estes*Beacon*s devem cumprir, de forma ser possível comunicar com os seus sistemas operativos, a que apelidou *iBeacon*

iBeacons

A Apple com a ambição de ligar os seus dispositivos eletrónicos ao mundo externo decidiu criar um protocolo que apelidou *iBeacon*. O seu intuito é criar regiões à volta do *Beacon*s, que podem estimar a proximidade do utilizador a este e por sua vez a um produto (Figura 2.5). Apesar de ser um protocolo desenvolvida pela Apple, qualquer dispositivo móvel com uma placa BLE é capaz de receber e interpretar estes pacotes.

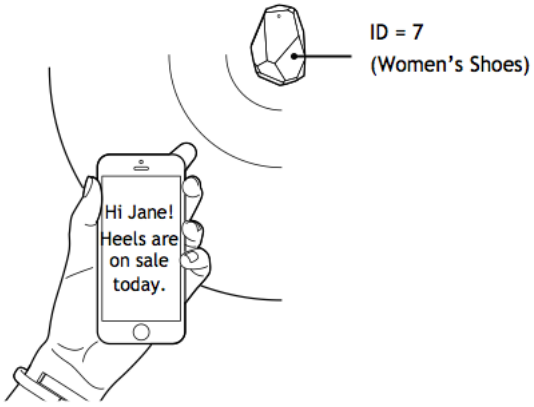


Figura 2.5: Uso de beacons [5]

Esta tecnologia tira partido do *advertisement channel* do BLE, emitindo constantemente pacotes, sendo que cada um transporta até um máximo de 31 bytes. Na figura 2.6 pode-se observar a informação enviada em cada pacote usando este protocolo [44][45][46].

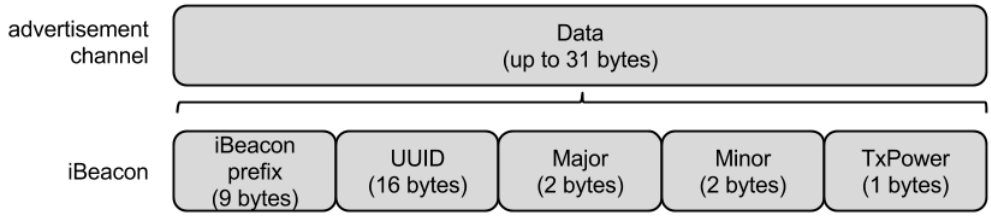


Figura 2.6: Informação enviada em cada pacote

iBeacon prefix

Este prefixo indica que o pacote provem de um iBeacon

UUID, Major, Minor

Servem para indicar inequivocamente a que beacon pertence o pacote. Por exemplo, o campo UUID pode ser partilhado por uma cadeia de lojas. O campo Major pode indicar qual a loja em questão e o Minor qual a

secção. Assim conseguimos inferir que o utilizador está em determinada loja numa determinada secção.

TX power

Indica a força do sinal medido a 1 metro, Este valor é usado para inferir a distância ao beacon.

2.3.7 Análise das tecnologias

Nesta secção é realizada uma analise das vários tecnologias documentadas nesta secção. Para isso são analisados os campos:

Margem de erro A margem de erro média da posição estimada;

Compatibilidade Compatibilidade com os sistemas operativos móveis atuais;

Escalabilidade A reação do sistema em relação a espaços diferentes;

Usabilidade Comodidade para o utilizador. Deverá ser possível estimar a posição do utilizador, com o *smartphone* no bolso. Este campo é importante, devido á introdução dos *smartwatches*, que devido ás suas limitações tem que ser o *smartphone* a estimar a posição do utilizador.

A tabela 2.2 demonstra a análise feita em relação a estes campos.

Como referido anteriormente nesta secção, tanto o NFC como o uso de redes de telecomunicações não são compatíveis com o iOS e como tal não satisfazem o campo compatibilidade.

Tecnologia	Margem de erro (metros)	Compatibilidade	Escalabilidade	Usabilidade
Bluetooth	2	✓	✓	✓
Wi-Fi	2	✓*	✓	✓
Câmara	1.5	✓	✓	✗
Giroscópio e Acelerómetro	10	✓	✓	✓
Rede de telecomunicações	5	✗	✗	✓
NFC	-	✗	✓	✗
Magnetómetro	4.7	✓	✗	✓

* Se o AP for capaz de estimar a distância ao utilizador;
✓- Satisfaz o campo analisado
✗- Não satisfaz o campo analisado.

Tabela 2.2: Analise das tecnologias capazes de localizar *smartphones* dentro de edifícios

No caso da escalabilidade para o uso das redes de comunicação e o Magnetómetro para estimar a posição do utilizador, é necessário usar técnicas de *fingerprinting*. Como visto na secção 2.2.3, este algoritmo necessita de uma fase de aprendizagem, onde é necessário retirar múltiplas amostras em múltiplos pontos, que apesar de sua realização ser simples, é demoroso. Uma vez que alterações no espaço vão influenciar os dados previamente gravados, será necessária realizar esta fase sempre que houver uma alteração no espaço.

Em relação á usabilidade, tanto o NFC como o uso da câmara, não satisfazem este campo. No caso da câmara, porque è necessário o *smartphone* estar na mão do utilizador apontado para cima. No caso do NFC porque este não é capaz de estimar a posição do utilizador de forma continua, necessitando que o utilizador se aproxime de uma tag NFC,

2.4 Sistemas Relacionados

Nesta secção é realizado um levantamento de sistemas atualmente presentes no mercado ligados à localização dentro de edifícios para superfícies comerciais. Uma vez que estes sistemas são proprietários, a informação técnica existente é muito limitada sendo informação levantada nesta secção proveniente dos *websites* de cada produto e de emails trocados com estes.

2.4.1 Senionlab

A SenionLab fornece um sistema de localização dentro de edifícios tirando partido das tecnologias Wi-Fi e/ou Bluetooth. Estes comercializam o dispositivo SenionBeacon que usa a tecnologia BLE, que deve ser instalado em variados pontos da área onde pretendemos implementar o sistema. Após a instalação destes dispositivos, o cliente envia o mapa para esta empresa. Esta empresa gera um percurso que o cliente deve percorrer de forma a recolher informação do espaço. Esta informação é depois analisada pela empresa de forma a permitir indicar a posição do utilizador.

Apesar de não existir informação que o confirme, é possível especular que utilizam um algoritmo de localização baseado no *fingerprinting*. Estes afirmam ter uma precisão entre 1 a 5 metros. Segundo emails trocados com a empresa, o kit de avaliação tem o preço de 1500 euros com 10 beacons acrescido do custo de cada beacon extra (40 Euros). Estes acrescentam que para uma instalação definitiva, existe o custo de instalação e um custo de licenciamento mensal baseado no tamanho da superfície comercial, apesar de não relevarem os possíveis custos.

2.4.2 Indoors

O Indoors é um sistema muito semelhante ao SenionLab. Este diferencia-se por ser um sistema mais maduro, tendo variadas ferramentas complementares já desenvolvidas como ferramentas para gerar mapas. Ao contrario da SenionLab que apenas permite o uso do SenionBeacon, este disponibiliza uma lista de dispositivos *bluetooth* de terceiros compatíveis.

Para realizar localização dentro de edifícios este sistema usa uma abordagem baseada no *fingerprinting*. Segundo os próprios, o sistema tem uma precisão de 5 metros em 95% dos casos. Estes indicam que o valor pode ser menor quanto maior o número de beacons, Wi-Fi ou Bluetooth, forem introduzidos. A versão base deste sistema tem o custo de 500$ por mês mais o custo dos beacons necessários.

2.4.3 Quuppa

A Quuppa desenvolveu um sistema proprietário para realização de localização dentro de edifícios. Para isso, estes desenvolveram um recetor que permite estimar o ângulo de determinado pacote *bluetooth* recebido. Com este(s) ângulo(s) (dependendo do numero de recetores) é aplicado um algoritmo baseado na angulação para estimar a posição do dispositivo. Para isto, os dispositivos têm que possuir um placa BLE e correr um software proprietário. Uma vez que em muitos dispositivos não é possível instalar software de terceiros, estes conceberam uma tag para este fim. Afirmam ser possível uma precisão inferior a 1 metro com o seu sistema. Não foi possível saber o custo deste sistema.

2.4.4 StoreMode

A StoreMode é uma plataforma desenvolvida pela empresa americana Point Inside. Com esta plataforma é possível criar mapas interativos, gerir a lista de compras, calcular o caminho mais eficiente de acordo com lista de compras e realizar variados tipos de pesquisas de produtos. Do ponto de vista da superfície comercial esta plataforma realiza análise do comportamento dos utilizadores recomendando ações. No entanto, esta plataforma é apenas um *backend*, e não uma solução completa como o aisle411.

2.4.5 aisle411

A aisle411 dedica-se à melhoria da experiência de utilizadores dentro de superfícies comerciais. Estes criaram um sistema que permite aos clientes da superfície comercial criar uma lista de compras e mapear os itens presentes nesta lista e pesquisa por outros.

Em relação aos métodos usados para realizar localização dentro de edifícios, estes usam diferentes tecnologias consoante o espaço e as necessidades do cliente. Segundo [47], em 2013, estes usavam um algoritmo de *fingerprinting* usando Wi-Fi, existindo a limitação que apenas era possível estimar a localização do utilizador em *smartphones* com o sistema operativo Android. Atualmente estes têm parcerias com empresas como a IndoorAtlas, Estimote e Cisco, empresas com várias soluções nesta área, tento essa limitação sido ultrapassada. Estes afirmam que o seu sistema tem uma precisão de 2 metros, apesar de não especificarem com que tecnologia ou conjunto de tecnologias. Os custos de implementação deste sistema podem rondar os 250 mil dólares.

Sendo este sistema semelhante ao proposto, foi realizada uma análise na tabela 2.3

Sistema	aisle411	Whitesmith
Pesquisa de produtos	✓	✓
Lista de compras	✓	✓
Geração do percurso mais curto	✓	✓
indoor location	✓	✓
Analise de dados	✓	✓*
Check-out	✓	✓*
Custo	Elevado	Baixo
Margem de Erro	2 metros	1 metro

*Planeado para a próxima iteração do sistema.
✓- Satisfaz o campo analisado.

Tabela 2.3: Comparação entre o sistema ailse411 e Whitesmith

Com está é possível observar que os sistemas apresentam semelhanças, no entanto o sistema da Whitesmith destaca-se pela positiva nos pontos "Margem de erro" e "Custo".

2.5 Desenvolvimento de Aplicações Móveis

Neste capítulo são abordados os vários tipos de aplicações que são possíveis desenvolver para os *smartphones*, fazendo um levantamento das suas vantagens e desvantagens.

Para desenvolvimento de aplicações móveis existem três formas distintas de o fazer:

Aplicações Nativas

As aplicações nativas são desenvolvidas especificamente para cada sistema operativo móvel, usando *Software Development Kit* ou Kit de Desenvolvimento

de Software (SDK) e *Application Programming Interface* (API) de cada plataforma e as suas linguagens de programação (Java em Android, Ojective-C ou Swift em iOS e C# no caso de Windows Phone). Estes foram desenvolvidos de forma a permitir ao programador tirar o máximo partido das capacidades do *smarphone* e do seu sistema operativo, tanto em termos de funcionalidades como de performance.

A necessidade de criar diferentes aplicações para os diferentes sistemas operativos é o maior problema desta abordagem, onde o código não pode ser reaproveitado entre diferentes sistemas operativos. Por isso, existem algumas ferramentas que nos permitem partilhar algum código entre as diferentes aplicações e/ou usar diferentes linguagens de programação, sendo este código compilado para a arquitetura em questão. Alguns exemplos destas ferramentas são o RubyMotion, que se programa na linguagem Ruby e permite a criação de aplicações compatíveis com iOS e Android, ou Xamarin onde se programa na linguagem C# e permite desenvolver para iOS, Android e Windowns Phone.

Aplicações Web

Ao aplicações web ou *Web Apps* são desenvolvidas usando tecnologias como o HTML5, CSS e javascript, acedidas a partir dos *browsers*, sendo assim necessária uma ligação à Internet. A vantagem em relação às aplicações nativas, são o facto de estas funcionarem em todos os sistemas operativos móveis atuais mas terem acesso limitado às funcionalidades do telemóvel, como aceder aos contactos ou correr em *background*. Ao contrário das aplicações nativas, em que cada ação pode ter um efeito imediato no estado da aplicação como uma alteração de *view*, neste tipo de aplicações isto não é possível uma vez que, usualmente, cada ação implica um pedido ao servidor e interpretação da

resposta.

Aplicações Híbridas

Os SDK dos vários sistemas operativos móveis contêm na sua *User Interface* (UI) *framework* uma *Web View*, que permite mostrar conteúdo HTML a partir de um servidor remoto ou localmente. As aplicações híbridas tiram proveito destas, sendo essencialmente aplicações web a correr numa aplicação nativa com numa *Web View*. Isto permite aceder a APIs nativas que, normalmente, não estão disponíveis para aplicações web, como a câmara, contactos ou correr em *background*. Uma vez que o UI é desenvolvido em HTML/CSS, este é compatível com todos os sistemas operativos, mas as chamadas a API nativas têm que ser alteradas consoante os vários sistemas operativos. Com o intuito de colmatar esta falha, existem *frameworks* como PhoneGap, Ionic, Trigger.io ou Cordova que funcionam como uma ponte entre a aplicação web e a aplicação nativa. Para isso, incluem código em JavaScript que permite comunicar assincronamente entre a aplicação nativa e a *web view*. Estas *frameworks*, interpretam estas mensagens, e realizam as chamadas às API nativas para o sistema operativo em que está a ser executado.

Vantagens/Desvantagens

Em seguida é realizada uma comparação entre estes vários tipos de aplicação, onde são comparados segundo os campos de custo, portabilidade de código, capacidade de acesso às funcionalidades do *smartphone*, consistência da UI, distribuição e *performance* [48][49].

Custo As aplicações nativas têm, normalmente, um maior custo devido às várias linguagens de programação para cada ecossistema, caso se pretenda desenvolver para as várias plataformas. Estas necessitam de equipas com conhecimentos específicos para cada ecossistema. As aplicações Web são as de menor custo, devido ao facto de um projeto funcionar automaticamente com todas as plataformas, e do facto de apenas ser necessário conhecimentos em desenvolvimento para a web. Nas aplicações multi-plataforma (aplicações híbridas e nativas usando *frameworks* de terceiros) é possível a partilha de parte do código ao longo dos vários ecossistemas, apresentando-se igualmente com um baixo custo de desenvolvimento.

Apesar disso, o custo está dependente de outros fatores, por isso as aplicações nativas podem nem sempre ser as mais dispendiosas de desenvolver. Principalmente no caso onde um elevado nível de personalização é envolvido, uma vez que as APIs das *frameworks* de terceiros, apenas contêm acessos genéricos/comuns às APIs do *smartphone*.

Portabilidade A portabilidade ou partilha de código, é a maior fraqueza das aplicações nativas, uma vez que estas não permitem partilha de qualquer tipo de código. Já no caso das aplicações web um projeto funciona em todas as plataformas, sendo a compatibilidade com os diferentes *browsers* a única preocupação. No caso das aplicações multi-plataforma, grande parte do código pode ser partilhado pelas várias plataformas. É usualmente necessário, alterações no *layout* das aplicações, para ficarem em conformidade com as normas de cada ecossistema e as suas lojas móveis. Outras alterações podem ter que ser necessárias realizar, consoante a ferramenta de desenvolvimento utilizada.

Capacidade de acesso ás funcionalidades do *smartphone* As aplicações web apenas conseguem aceder a algumas APIs do *smartphone* como ao GPS, mas são ainda muito limitadas. A W3C está atualmente a trabalhar em novos *standards* de forma a aumentar o número de APIs disponíveis para este tipo de aplicações. As aplicações nativas são as com maior acesso aos *smartphone*, devido a serem mantidas pelos proprietários de cada sistema operativo móvel. Assim, sempre que uma nova funcionalidade é introduzida num novo *smartphone*, estes criam APIs de forma a dar acesso aos programadores, caso assim o pretendam. Já no caso das aplicações multi-plataforma, está usualmente dependente do suporte da ferramenta utilizada.

Consistência na UI As aplicações nativas tiram partido dos UI Kits desenvolvidos pelos responsáveis por cada sistema operativo, garantindo consistência entre as aplicações de terceiros com o seu sistema operativo. Já no caso das aplicações web e multi-plataforma as UIs são desenvolvidos usando HTML, CSS e Javascript tentando assemelhar-se às aplicações nativas, mas pormenores como efeitos gráficos, não conseguem ser simulados. O facto de não usarem os UI Kits de cada plataforma, levanta um problema no caso de uma atualização no *layout* do sistema operativo. Aquando destas atualizações, os UI Kits são igualmente atualizados, passando as aplicações a serem consistentes com essa versão do sistema operativo, com um pequeno ou mesmo nenhum esforço do lado do programador. Já no caso das aplicações web e híbridas, onde o *layout* da aplicação é independente destes UI Kits nativos, este nível de atualizações torna-se mais complexo de realizar.

Distribuição As aplicações nativas e híbridas, são instaladas no sistema ope-

rativo do utilizador, sendo por isso usualmente usadas as lojas móveis. No caso do iOS apenas aplicações na Apple Store podem ser instaladas neste sistema operativo. No caso do Android, não existe este bloqueio, podendo o utilizador instalar aplicações provenientes ou de lojas de terceiros ou através do ficheiro de instalação (.apk). Mas a Google Play Store, loja da Google para este sistema operativo, é a loja com maior número de *downloads* para o Android, caso seja removido o mercado chinês, onde este loja não está presente atualmente[50]. Por esta razão torna-se importante ter presença nestas duas lojas móveis, apesar de ambas terem variados tipos de diretrizes que os programadores têm que cumprir, tanto ao nível do tipo de aplicações permitidas como ao nível do design da aplicação. Estas lojas garantem ao utilizador segurança e comodidade para encontrar e instalar aplicações. Apesar das aplicações Web não terem qualquer tipo de restrição e custo associado (as lojas móveis cobram aproximadamente 30% do custo da aplicação), também não têm uma forma tão fácil de chegar até aos seus consumidores.

Performance As aplicações nativas comunicam diretamente com as APIs do sistema e são programadas diretamente na linguagem em que o sistema está otimizado para interpretar/compilar. São estas as mais capazes de extrair a maior performance possível de cada *smartphone*. As aplicações multi-plataforma nativas, sofrem do problema das várias *layers* de abstração existentes, para ser possível, por exemplo, partilhar *views*, entre as várias plataformas. Uma vez que nas aplicações web e híbridas, todo o tipo de renderização como botões, animações ou transições de *views* são emulados usando HTML, CSS e JavaScript, não é tirado diretamente partido do *hardware* do telemóvel, existindo ainda uma maior perda de performance. Nas aplicações web acresce o facto de depender

diretamente da ligação à rede.

Na tabela 2.4 pode-se observar um resumo dos tópicos em cima referidos, e classificados como vantagem, desvantagem e neutro (quando determinado tópico analisado para determinada aplicação não se apresenta como uma vantagem ou desvantagem).

Como se pode observar não existe uma melhor abordagem, dependendo sempre dos requisitos e restrições de determinado projeto, cabendo à equipa de desenvolvimento decidir o que melhor se adapta às suas necessidades.

	Nativa	Multi-plataforma	Web
Custo	O mais elevado, caso se pretenda desenvolver para as várias plataformas	Semelhante às aplicações web mas um conjunto extra de conhecimentos é requerido.	O mais baixo dos três, devido ao facto de um projeto funcionar em todas as plataformas e de apenas ser necessário conhecimentos na área de desenvolvimento Web.
Portatibilidade	O código para um plataforma apenas funciona nessa plataforma	As ferramentas para desenvolvimento destas aplicações permitem a portabilidade de código para os vários sistemas operativos móveis.	Compatibilidade com os diversos *browsers* e performance são as únicas preocupações
Acesso às funcionalidades do *smartphone*	As plataformas de desenvolvimento para cada sistema operativo permitem aceder a todas as APIs do *smartphone*.	Várias APIs do *smartphone* podem ser acedidas, mas depende da ferramenta utilizada.	Poucas APIs podem ser acedidas
Consistência da UI	A plataforma de desenvolvimento contém componentes UI uniformizados com o sistema operativo em questão.	As frameworks de UI permitem alcançar um aspeto semelhante ao nativo	As *frameworks* de UI permitem alcançar um aspeto semelhante ao nativo
Distribuição	As lojas de aplicações móveis proporcionam benefícios de *marketing* mas têm requisitos, restrições e custos.	As lojas de aplicações móveis proporcionam benefícios de *marketing* mas têm requisitos, restrições e custos.	Sem restrições, mas não se tem os benefícios das lojas móveis.
Performance	Código nativo acede diretamente às funcionalidades da plataforma, resultando numa melhor experiência	Para aplicações complexas, as camadas de abstração impedem uma performance como as aplicações nativas.	Performance é baseada no browser e na ligação de Internet.

Legenda: Ponto a favor; Ponto neutro; Ponto contra.

Tabela 2.4: Aplicações Nativas vs Multi-plataforma vs Web

Capítulo 3

Metodologia e Planeamento

3.1 Metodologia

Para este projeto foi escolhida uma metodologia ágil. Mais concretamente foi escolhida *Lean Software Development* (LSD). Esta metodologia foi escolhida uma vez que se enquadra tanto com a filosofia da empresa como do estagiário. Os princípios desta metodologia são:

- Eliminar desperdício;
- Aumentar o conhecimento;
- Decidir o mais tarde possível;
- Entregar o mais rápido possível;
- Dar poder à equipa;
- Construir integridade (Integridade de perceção e conceptual);
- Ver o inteiro.

Eliminar desperdício, como burocracias e outros artefactos menores que outras metodologias requerem e dar poder de decisão ao estagiário, leva a ciclos mais rápidos de desenvolvimento assim como ao mesmo tempo melhora a aprendizagem da equipa, podendo levar à construção de um melhor sistema e ao aumento do conhecimento do estagiário, satisfazendo assim o objetivo do estágio.
Olhar para o sistema como um todo juntamente com o princípio de decidir o mais tarde possível permite uma rápida adaptação do sistema a novos requisitos.

Para manter o controlo das tarefas estabelecidas e ao mesmo tempo da evolução do sistema, foi usada a ferramenta Trello [1]. Esta ferramenta permite organizar tarefas por secções personalizadas. Estas secções foram inspiradas no quadro de Kanban. Estão assim divididas da seguinte forma:

- *Could Do*, inserimos neste estado as tarefas que se podem realizar caso seja possível.
- *To Do*, as tarefas inseridas neste estado são as que temos obrigatoriamente de realizar.
- *Doing*, as tarefas após iniciadas são postas neste estado.
- *Done*, quando a tarefa é concluída passamo-la para este estado.

Assim, olhando para este quadro conseguimos facilmente identificar o estado em que cada tarefa se encontra em cada momento.

Para controlo das versões geradas foi usado um *Version Control System* (VCS). A escolha recaiu sobre o Git, devido aos conhecimentos do estagiário e devido

[1] Trelo: https://trello.com/

ao facto de ser usado pela empresa. O repositório Git é sincronizado com uma versão online hospedada pelo GitHub [2], o serviço usado pela empresa.

3.2 Planeamento

3.2.1 Primeiro Semestre

Durante o primeiro semestre foram estabelecidas as seguintes tarefas:

- Estado da arte;
- Definição da arquitetura do sistema;
- Definição dos requisitos do sistema;
- Início do desenvolvimento;

Para além das tarefas estabelecidas o estagiário teve a necessidade de estudar algumas tecnologias para o desenvolvimento deste projeto:

- Swift, nova linguagem introduzida pela Apple;
- Interação dos *beacons* com iOS;
- *Inter-App Communication*;
- Comunicação eficiente com o servidor em iOS.

Por isso, foi efetuado um planeamento do semestre que originou o diagrama de Gant presente na figura 3.1,

[2]GitHub: https://github.com/

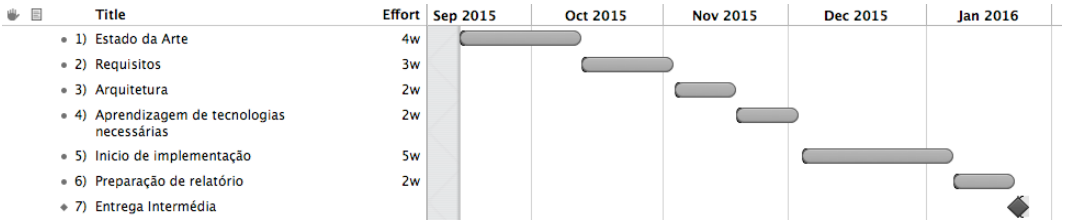


Figura 3.1: Diagrama de Gant inicial relativo ao primeiro semestre

Inicialmente o estágio tinha por objetivo aproveitar as capacidades do BLE para o contexto da *smart home*. Mas, com o início da realização do estado da arte, foram encontradas diversas soluções que tinham sido recentemente lançadas para o efeito, e com o propósito de manter um carácter inovador no estágio, foi proposta a sua alteração. Com isto, houve um desvio de três semanas em relação ao plano inicial.

O estudo das tecnologias necessárias demorou mais uma semana do que o previsto, essencialmente pela necessidade do estudo do funcionamento dos *beacons* e a sua integração com o sistema operativo iOS.

Com estes desvios, o tempo de implementação foi reduzido em três semanas.

O diagrama de Gant presente na figura 3.2 mostra a real distribuição temporal das tarefas ao longo do primeiro semestre.

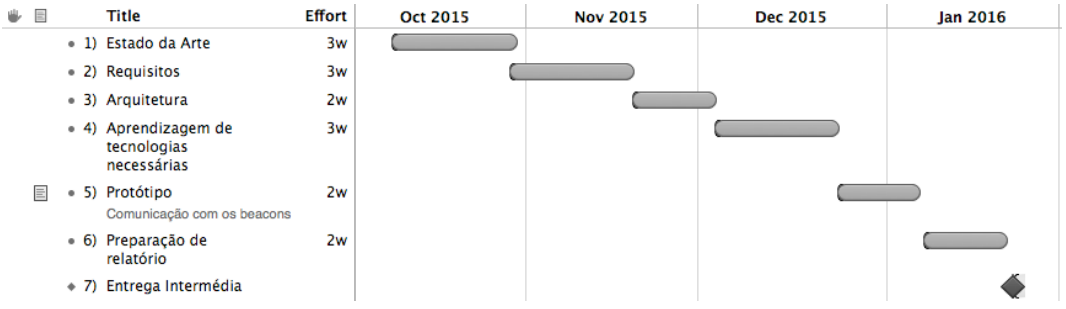


Figura 3.2: Diagrama de Gant final relativo ao primeiro semestre

Pode-se observar que foi iniciado o desenvolvimento de um protótipo. Este protótipo tinha por objetivo validar os objetivos deste estágio, sendo por isso desenvolvido um algoritmo simples de localização dentro de edifícios.

3.2.2 Segundo Semestre

Após a discussão na defesa intermédia, houve uma necessidade de ajustar os objetivos deste estágio, por este ser complexo e extenso. Inicialmente este estágio tinha por objetivo a criação das três diferentes aplicações do sistema (figura 3.3). Este objetivo foi redefinido, passando o estagiário a ser apenas responsável pelo desenvolvimento da aplicação de guia dos clientes (figura 3.4), tal como sugerido na defesa intermédia. No anexo B estão presentes ambos os diagramas detalhados.

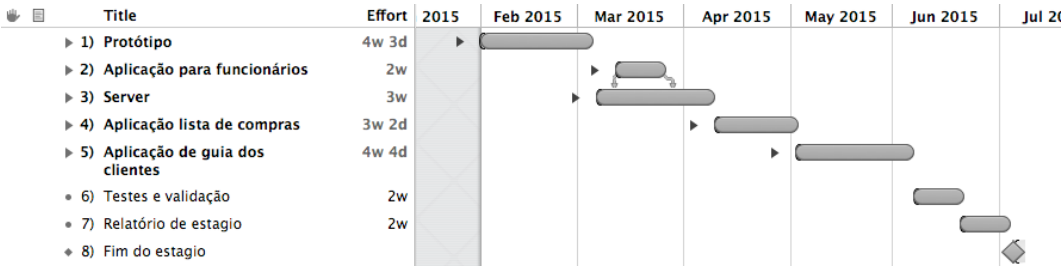


Figura 3.3: Diagrama de Gant planeado para o segundo semestre

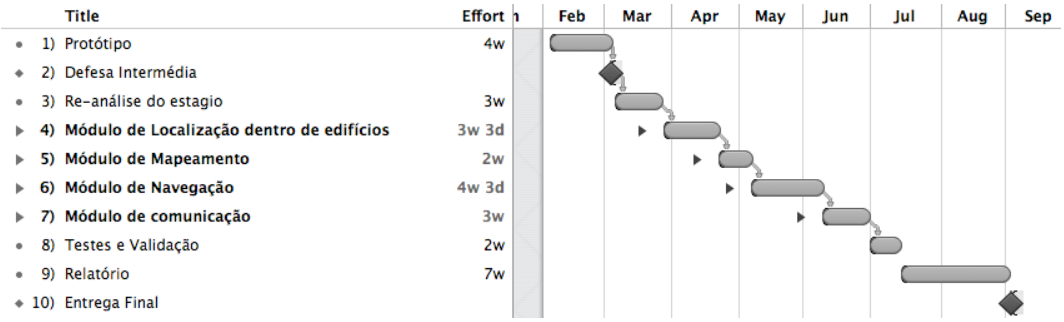


Figura 3.4: Diagrama de Gant real do segundo semestre

Após a finalização do protótipo, com resultados promissores, iniciou-se o desenvolvimento da aplicação de localização dento de edifícios.

Esta aplicação é composta por 4 diferentes módulos:

- **Módulo de localização dentro de edifícios** responsável por estimar a posição do utilizador;
- **Módulo de mapeamento** responsável pelo mapeamento da espaço;
- **Módulo de navegação** responsável pela geração do percurso mais curto e de indicar ao utilizador quais as direções que deve seguir;
- **Módulo de comunicação** responsável pela comunicação com o servidor e com outras aplicações.

Cada um destes módulos vão ser vistos em mais detalhe nos capítulos seguintes.

Capítulo 4

Requisitos

Neste capítulo são abordados os requisitos do sistema. Para isso é feito um levantamento dos *stakeholders*, casos de uso, requisitos funcionais e de qualidade deste projeto.

4.1 *Stakeholders*

Os *stakeholders* são agentes que interagem de forma direta ou indereta com o sistema, que englobam [51]:

- Financiadores do produto;
- Responsáveis pelo desenvolvimento;
- Utilizadores do produto.

Na tabela 4.1 estão referenciados os *stakeholders* deste sistema.

Stakeholders	Descrição	Responsabilidade
Whitesmith	Empresa proprietária do projeto	Comunicar necessidades e validar o sistema
Coordenador do Projeto	Eng. Rafael Jegundo	Monitorizar os processos de desenvolvimento e utilização
Utilizador	Clientes da superfície comercial que utilizam a aplicação	-
Superfície comercial	Superfície comercial onde o sistema está implementado	Responsável por manter o sistema, comunicando alterações de espaço e de produtos

Tabela 4.1: *Stakeholders* - Descrição e Responsabilidades

4.2 Casos de Uso

Nesta secção são mostrados os casos de uso deste sistema dividido pelas diferentes aplicações.

Aplicação de Gestão Lista de Compras

- **Lista de compras**

  Descrição Geral: O utilizador quer ser capaz de ver a sua lista de compras

  Entidades Envolvidas: Utilizador

  Pré-Condição: O Utilizador está autenticado

  Fluxo de eventos: 1. Entra na aplicação

  2. É mostrada a lista de compras

- **Inserir produto na lista de compras**

  Descrição Geral: O utilizador quer ser capaz de inserir produtos na sua lista de compras.

  Entidades Envolvidas: Utilizador

  Pré-Condição: O utilizador está autenticado

  Fluxo de eventos: 1. Entra na aplicação
  2. Clica no botão de inserir
  3. Insere o nome do produto e quantidade.

  Pós-Condições: 1. O sistema é capaz de identificar o produto e a quantidade inserida.
  2. O produto fica inserido na lista de compras.

- **Eliminar/Editar um produto inserido**

  Descrição Geral: O utilizador quer ser capaz de eliminar ou editar determinado produto.

  Entidades Envolvidas: Utilizador

  Pré-Condição: 1. O utilizador está autenticado
  2. O utilizador tem produtos na lista

  Fluxo de eventos: 1. Entra na aplicação
  2. Clica no produto
  3. A linha do produto fica editável sendo possível apagar arrastando o dedo.

  Pós-Condições: O produto é editado/eliminado

- **Enviar lista de compras para a aplicação de guia dos clientes**

  Descrição Geral: O utilizador quer ser capaz de enviar a sua lista para a de guia de clientes de forma a ser possível indicar o percurso mais curto

  Entidades Envolvidas: Utilizador

  Pré-Condição: 1. O utilizador está autenticado
  2. A aplicação de guia dos clientes está instalada

  Fluxo de eventos: 1. Entra na aplicação
  2. Clica no botão de exportar para a aplicação de guia dos clientes

  Pós-Condições: A lista é enviada para a aplicação de guia dos clientes

Aplicação dos Funcionários

- **Pesquisar por produtos**

  Descrição Geral: O funcionário deve ser capaz de pesquisar por produtos existentes na superfície comercial.

  Entidades Envolvidas: Funcionário

  Pré-Condição: O funcionário está autenticado

  Fluxo de eventos: 1. Entra na aplicação
  2. Clica no botão de pesquisa
  3. É mostrada uma lista categorizada dos produtos e um local onde é possível inserir o nome do produto em questão
  4. É mostrada uma lista de produtos de acordo com a seleção anterior

5. Clica num produto
6. São mostrados detalhes do produto

- **Pesquisar por produtos por código de barras**

  Descrição Geral: O funcionário deve ser capaz de pesquisar produtos através do código de barras.

  Entidades Envolvidas: Funcionário

  Pré-Condição: O funcionário está autenticado

  Fluxo de eventos:
  1. Entra na aplicação;
  2. Clica no botão de pesquisa;
  3. Clica no botão de pesquisa por código de barras;
  4. Aponta a câmara para o código de barras;
  5. São mostrados detalhes do produto

- **Editar produto**

  Descrição Geral: O funcionário deve ser capaz de editar informação de um produto.

  Entidades Envolvidas: Funcionário

  Pré-Condição:
  1. O funcionário está autenticado
  2. Pesquisou pelo produto que quer editar

  Fluxo de eventos:
  1. (Pesquisa pelo produto);
  2. Clica no botão editar;
  3. Edita um produto.

Pós-Condições: O produto fica com os valores por este introduzido

- **Adicionar um novo produto**

  Descrição Geral: O funcionário deve ser capaz de adicionar um novo produto.

  Entidades Envolvidas: Funcionário

  Pré-Condição: 1. O funcionário está autenticado

  2. Pesquisou pelo produto e este não foi encontrado no sistema

  Fluxo de eventos: 1. Clica em adicionar produto;

  2. Insere dados relativos aos produtos;

  Pós-Condições: O produto é introduzido no sistema

- **Reposição de *stock***

  Descrição Geral: O funcionário deve atualizar informação quando repõe um produto.

  Entidades Envolvidas: Funcionário

  Pré-Condição: 1. O funcionário está autenticado

  2. Pesquisou pelo produto que está a repor

  Fluxo de eventos: 1. (Pesquisa pelo produto);

  2. Clica em repor;

  3. Indica quantas unidades do produto está a repor;

  4. O sistema verifica se está na posição habitual do produto;

  5. Caso não esteja, é pedido ao funcionário para introduzir a nova posição.

Pós-Condições: O sistema guarda esta nova informação inserida

Aplicação de Guia dos Clientes

- **Pesquisa e detalhes de produtos**

  Descrição Geral: O cliente deve ser capaz de pesquisar e ver detalhes dos produtos comercializados pela superfície comercial

  Entidades Envolvidas: Cliente

  Fluxo de eventos: 1. Entra na aplicação;
  2. Clica no botão de pesquisa;
  3. É mostrada uma lista categorizada dos produtos e um local onde é possível inserir o nome do produto em questão;
  4. É mostrada uma lista de produtos de acordo com a seleção anterior
  5. Clica num produto;
  6. São mostrados detalhes desse produto.

- **Indicar o percurso para determinado produto**

  Descrição Geral: A aplicação deve ser capaz de indicar ao cliente como chegar até determinado produto

  Entidades Envolvidas: Cliente

  Pré-Condição: O cliente pesquisa por um produto

  Fluxo de eventos: 1. (clica no produto)
  2. Clica no botão de navegação
  3. É gerado o percurso até esse produto

4. O percurso é mostrado
5. A aplicação vai indicando o percurso a percorrer.

- **Receber a lista de compras**

  Descrição Geral: O sistema deve ser capaz de receber uma lista de compras proveniente de outra aplicação.

  Entidades Envolvidas: 1. Cliente
  2. Aplicação de lista de compras

  Fluxo de eventos: Recebe lista de compras proveniente de outra aplicação

  Pós-Condições: A lista é interpretada

- **Gerar percurso de acordo com a lista de compras**

  Descrição Geral: O sistema deve ser capaz de gerar o percurso mais curto, consoante a lista de compras do cliente.

  Entidades Envolvidas: 1. Cliente
  2. Aplicação de lista de compras

  Pré-Condição: A aplicação interpretou a lista de compras

  Fluxo de eventos: 1. É gerado o percurso mais curto
  2. O percurso é mostrado
  3. A aplicação vai indicando o percuso a percorrer.

4.3 Requisitos Funcionais

Com base nos casos de uso previamente descritos foram levantados os requisitos funcionais descritos nesta secção.

Aplicação de Gestão Lista de Compras

- ID: RF-LC1

  Título: Autenticação

  Entidade: Utilizador

  Descrição: Utilizador deve poder-se autenticar na aplicação (*username* e *password*)

  Racional: Por forma a ter uma conta com os seus dados e preferências.

- ID: RF-LC2

  Título: Mostrar lista de compras

  Entidade: Sistema

  Descrição: O utilizador quando abre a aplicação deve ter acesso imediato à sua lista de compras

  Racional: Uma vez que a funcionalidade principal desta aplicação é a gestão da lista de compras do utilizador, assim que este abre a aplicação deve ser mostrada uma lista com todos os produtos que o utilizador pretende comprar.

  Dependência: RF-LC1, RF-LC5

- ID: RF-LC3

  Título: Inserir produtos na lista

  Entidade: Utilizador e Sistema

Descrição: O utilizador deve ser capaz de introduzir o que pretende comprar,

Racional: O utilizador deve ser capaz de introduzir o que produtos na sua lista de compras de forma simplificada.

Dependência: RF-LC1

- ID: RF-LC4

  Título: Gestão da Lista de Compras

  Entidade: Utilizador

  Descrição: O utilizador deve ser capaz de adicionar/remover/editar a sua lista de compras.

  Racional: Por forma a permitir a gestão da sua lista de compras.

  Dependência: RF-LC2, RF-LC4

- ID: RF-LC5

  Título: Sincronização

  Entidade: Sistema

  Descrição: O sistema deve sincronizar a lista de compras.

  Racional: Por forma a permitir a utilização de vários dispositivos para gestão da lista de compras pelo mesmo utilizador.

  Dependência: RF-LC1

- ID: RF-LC6

  Título: Exportar lista de compras

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de enviar a lista de compras.

  Racional: De forma a permitir á aplicação de guia dos clientes gerar o percurso mais curto.

  Dependência: RF-LC1

Aplicação para os Funcionários

- ID: RF-F1

  Título: Autenticação

  Entidade: Superfície comercial

  Descrição: Utilizador deve poder-se autenticar na aplicação (*username* e *password*)

  Racional: Por forma a ter acesso às funcionalidades de administração.

- ID: RF-F2

  Título: Listar produtos

  Entidade: Sistema

  Descrição: A aplicação deve ser capaz de listar e categorizar todos os artigos comercializados pela superfície comercial.

Racional: Por forma a possibilitar aos funcionários visualizar todos os produtos comercializados pela superfície comercial.

Dependência: RF-F1

- ID: RF-F3

  Título: Editar produtos

  Entidade: Sistema

  Descrição: A aplicação deve ser capaz de editar a informação dos produtos

  Racional: Por forma a possibilitar aos funcionários editar a informação de um dado produto.

  Dependência: RF-F1,RF-F2

- ID: RF-F4

  Título: Adicionar produtos

  Entidade: Superfície Comercial

  Descrição: A aplicação deve permitir a adição de novos produtos.

  Racional: Por forma a possibilitar aos funcionários adicionar novos produtos e informação correspondente a ele, incluindo o seu código de barras.

  Dependência: RF-F1

- ID: RF-F5

  Título: Pesquisa de Produtos

  Entidade: Superfície Comercial

  Descrição: O funcionário da superfície comercial deve ser capaz de pesquisar por produtos por nome e por código de barras.

  Racional: Por forma a facilitar a pesquisa de produtos.

  Dependência: RF-F1, RF-F2

- ID: RF-F6

  Título: Localização

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de identificar a posição do *smartphone*

  Racional: De forma a auxiliar a introdução da posição do produto no sistema.

Aplicação de Guia dos Clientes

- ID: RF-GC1

  Título: Mapa

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de mostrar o mapa da superfície

  Racional: Por forma ao utilizador poder observar o mapa da superfície

- ID: RF-GC2

  Título: Posição

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de localizar a posição do utilizador no mapa da superfície.

  Racional: Por forma a orientar o utilizador.

  Dependência: RF-GC1

- ID: RF-GC3[1]

  Título: Piso

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de localizar o piso do utilizador

  Racional: Por forma a mostrar o mapa indicado para cada piso

  Dependência: RF-GC1, RF-GC2

- ID: RF-GC4

  Título: Pesquisa de Produtos

  Entidade: Utilizador

  Descrição: O utilizador deve ser capaz de pesquisar por produtos presentes na superfície comercial.

  Racional: Por forma a poder ter detalhes do produto e a sua posição.

[1]No âmbito deste estágio, assumiu-se que todas as superfícies comerciais só possuem um piso

Dependência: RF-GC1

- ID: RF-GC5

  Título: Localização de Produtos

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de identificar a posição de determinado produto

  Racional: Por forma a obter a localização desse produto para criar um percurso e mostrar ao utilizador.

  Dependência: RF-GC2

- ID: RF-GC6

  Título: Percurso

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de gerar um percurso para determinado produto

  Racional: Por forma a poder mostrar o percurso adequado ao utilizador

  Dependência: RF-GC1,F-GC2,RF-GC4, RF-GC5

- ID: RF-GC7

  Título: Guia

  Entidade: Sistema

Descrição: O sistema deve ser capaz de indicar as direções que o utilizador deve tomar

Racional: Por forma auxiliar o utilizador a encontrar o(s) produto(s)

Dependência: RF-GC6

- ID: RF-GC8

  Título: Lista de Compras

  Entidade: Sistema

  Descrição: O sistema deve ser capaz de receber e interpretar a lista de compras do utilizador.

  Racional: Por forma gerar o percurso que o utilizador deve efetuar de forma a minimizar o espaço percorrido

  Dependência: RF-GC6

4.4 Requisitos Não Funcionais

4.4.1 Atributos de qualidade

Os atributos de qualidade descrevem o comportamento esperado pelo sistema.

- ID: AQ1

  Título: Interface intuitiva;

Descrição: A interface deve ser o mais intuitiva possível por forma a permitir uma aprendizagem rápida por parte dos clientes. As funções principias de cada aplicação devem necessitar de apenas 3 toques para as realizar (Adicionar um produto á lista de compras, iniciar o modo de guia e adicionar a posição de um produto no sistema)

Finalidade: Por ser utilizado por pessoas sem formação a interface deve ser o mais simples e intuitiva possível, por forma a melhorar o experiência dos utilizadores ao utilizar a aplicação;

Tipo: Usabilidade.

- ID: AQ2

  Título: Contextualização espacial;

  Descrição: A aplicação não deve mostrar funcionalidades se a superfície comercial não a suporta;

  Finalidade: De forma a criar uma experiência melhorada, evitando mensagens de aviso/erro;

  Tipo: Usabilidade.

- ID: AQ3

  Título: Eficiência energética;

  Descrição: O sistema de localização deve ser tão ou mais eficiente energicamente quando comparado com sistemas de localização como o GPS;

Finalidade: Devido às baterias limitadas dos *smarthpones* atuais, o sistema não deve consumir demasiada energia, sendo o GPS um bom *benchmark*;

Tipo: Performance.

- ID: AQ4

  Título: Suporte;

  Descrição: O código fonte, as configurações dos sistemas e as configurações de desenvolvimento devem estar documentadas;

  Finalidade: De forma a facilitar a novos programadores continuar/contribuirem para o projeto.

  Tipo: Manutenção.

- ID: AQ5

  Título: Portabilidade do sistema.

  Descrição: As tecnologias e algoritmos usadas para estimar a posição do utilizador devem ter em conta os vários sistemas operativos móveis.

  Finalidade: De forma a permitir a portabilidade do sistema para outros sistemas operativos móveis.

  Tipo: Portabilidade.

- ID: AQ6

  Título: Precisão do sistema de localização.

Descrição: O erro não deverá exceder os 5 metros em 90% dos casos.

Finalidade: De forma ser possível criar uma experiência imersiva para o utilizador.

Tipo: Performance.

- ID: AQ7

  Título: Segurança na posição do utilizador.

  Descrição: A posição do utilizador nunca deve ser guardada em memória persistente.

  Finalidade: De forma garantir ao utilizador total privacidade na utilização do serviço.

  Tipo: Segurança.

4.4.2 Restrições

As seguintes restrições foram identificadas a partir das diretrizes da empresa:

- ID: R1

  Título: Tamanho máximo da aplicação;

  Descrição: A aplicação não deverá exceder os 100MB. Devido às limitações impostas por algumas lojas móveis para realização de *downloads* via rede móvel. A aplicação não deverá assim exceder este limite, permitindo aos utilizadores realizar o *download* da aplicação em qualquer local;

Tipo: Técnica.

- ID: R2 [2]

  Título: Sistema de localização com BLE;

  Descrição: Deve ser usado BLE para localização dentro de edificios;

  Tipo: Stackehoder.

- ID: R3

  Título: *Smartphones* com placas BLE;

  Descrição: *Smartphones* têm que conter uma recetor compatível BLE;

  Tipo: Técnica;

  Racional: Por forma a estimar a localização do utilizador é necessário que o *smartphone* contenha uma placa compatível com o *standard* BLE.

[2] Apesar de imposta pela empresa, não contradiz as conclusões da análise do estado da arte

Capítulo 5

Arquitetura do Sistema

Neste capítulo é definida a arquitetura do sistema. O sistema é essencialmente composto por 4 partes:

1. A aplicação de gestão da lista de compras;
2. A aplicação de guia dos clientes;
3. A aplicação dos funcionários;
4. O servidor.

Importante referir que apesar de ter sido definida uma arquitetura para as diferentes aplicações, o estagiário apenas foi responsável por desenvolver a aplicação de guia dos clientes, sendo que esta tem um especial foco nesta secção.

5.1 Vista de Alto nível

Na figura 5.1 está representada uma vista de alto nível do sistema.

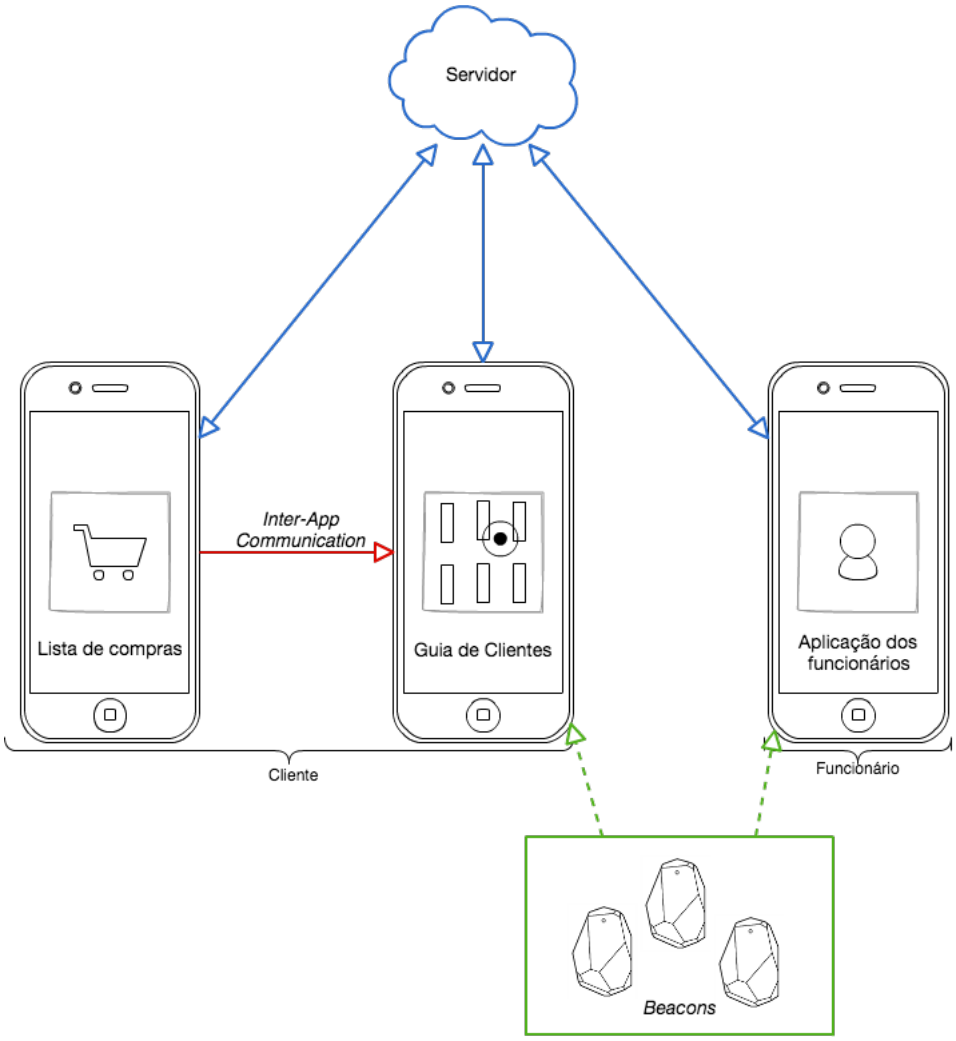


Figura 5.1: Arquitetura de alto nível do sistema

A divisão das aplicações pelo público alvo dos utilizadores é intuitiva, uma vez que os funcionários têm funcionalidades distintas dos clientes das superfícies comerciais. Mas a decisão da criação de duas aplicações diferentes para gestão de lista de compras e de *indoor location* está assente na premissa que cada aplicação móvel deve ter um foco específico, tornando as aplicações simples,

intuitivas e fáceis de usar. Ao mesmo tempo, isto cria a possibilidade dos utilizadores poderem utilizar aplicações de terceiros para gerirem a sua lista de compras, desde que estas implementem os protocolos necessários para comunicar com a aplicação de gestão de lista de compras. Do ponto do vista dos utilizadores estes não terão que realizar nenhum tipo de configuração.

Uma vez que grandes superfícies comerciais têm complexos sistema de gestão de produtos, onde sabem exatamente a posição de cada produto, não faz sentido para estes usarem a aplicação dos funcionários. Com esta arquitetura, é possível vender apenas a de guia de clientes, sem que tenham que usar a aplicação dos funcionários. Por outro lado, cadeias mais pequenas, que possuem sistemas de gestão mais simples, podem usufruir do sistema como um todo, usando as diferentes aplicações.

Como se pode observar na figura 5.1, a aplicação de lista de compras e a de *indoor location* têm o objetivo de serem usadas pelos clientes da superfície comercial. Devido ao facto da aplicação de guia dos clientes necessitar de ter acesso à lista de compras, é criada uma ligação entre as duas aplicações, usando mecanismos de *inter-app communication* (ligação vermelha).
Tanto a aplicação do funcionário como a de guia dos clientes necessitam de ter acesso ao local onde se encontra o *smartphone*. Com esta finalidade, ambas as aplicações analisam os sinais emitidos pelos *beacons* (ligação verde).

O servidor tem por objetivo trocar informação com as aplicações, enviando dados necessários para o seu funcionamento e guardar dados de cada uma delas, de forma garantir a sincronização de dados (ligação azul).

5.2 Arquitetura da Aplicação de Gestão da Lista de Compras

Na figura 5.2 podem observar-se alguns detalhes da aplicação responsável pela gestão de listas de compras.

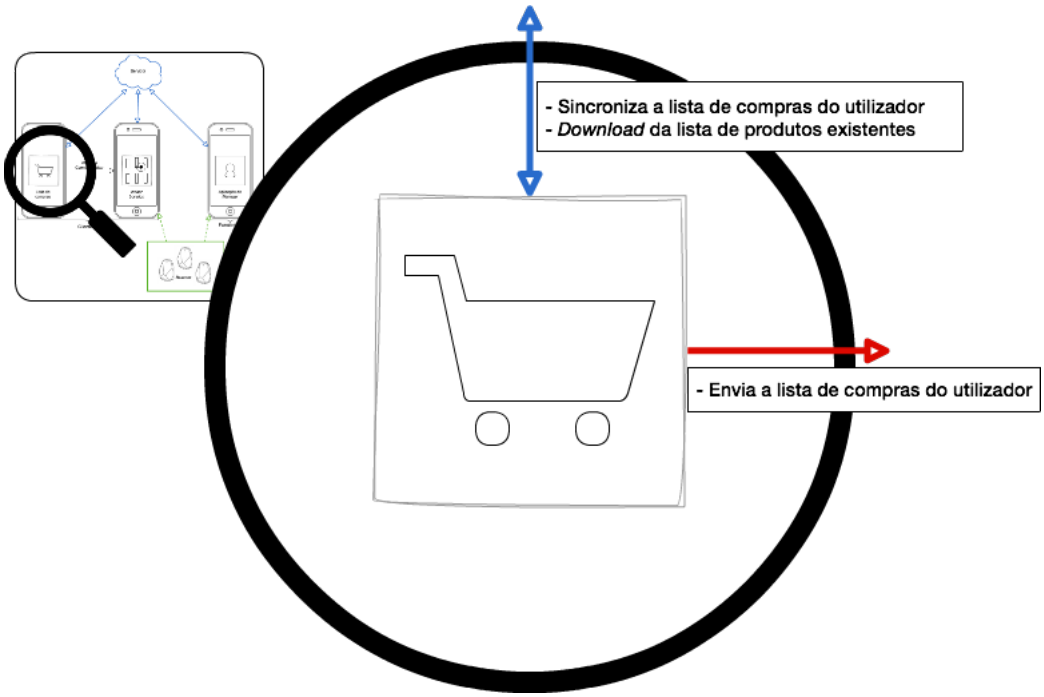


Figura 5.2: Arquitetura da aplicação da lista de compras

Esta aplicação apenas interage com o servidor (ligação azul) e com a aplicação de guida dos clientes (ligação vermelha).

Esta comunica com o servidor para ser capaz de sincronizar a base de dados de produtos de modo a que, como gestores do sistema, termos a capacidade de adicionar produtos sem termos que fazer atualizações à aplicação. As escolhas do utilizador também são enviadas para o servidor para permitir o acesso à nossa lista de compras a partir de vários dispositivos. Ao mesmo tempo possibilita uma análise de dados de consumo dos utilizadores.

Uma vez que é necessário enviar a lista de compras para a aplicação de guia dos clientes deve ser usado um mecanismo de *inter-app communication*, que se resume, na implementação de um protocolo de comunicação entre as duas aplicações. A informação a enviar consiste num identificador único do produto e da quantidade que o utilizador pretende comprar.

5.3 Arquitetura da Aplicação de Guia dos Clientes

Na figura 5.3 podem observar-se alguns detalhes da aplicação que é responsável por guiar os clientes pela superfície comercial.

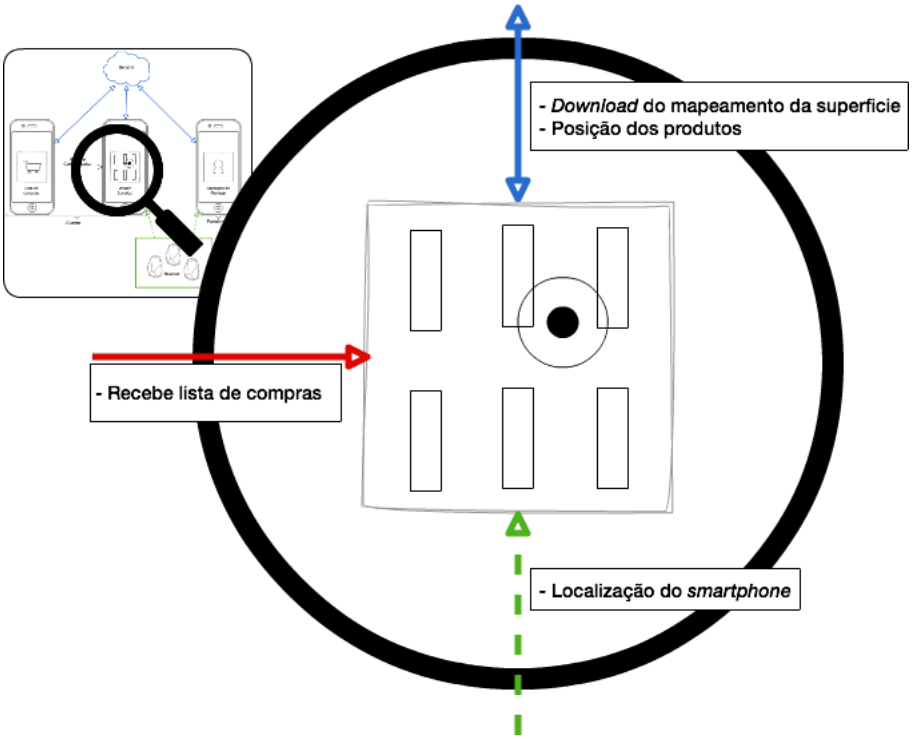


Figura 5.3: Arquitetura de alto nível da aplicação de guia dos clientes

Como referido anteriormente, esta aplicação necessita da informação relativa

à lista de compras, para ser capaz de indicar os produtos que o utilizador quer adquirir e calcular o caminho mais curto que o cliente deve percorrer, tirando partido de *inter-app communication* (ligação vermelha). Assim sendo esta aplicação recebe o id do produto e a quantidade de cada produto que o cliente pretende adquirir.

O servidor (ligação azul), é usado para fazer download do mapeamento do local, bem como das posições dos produtos presentes da lista de compras, de forma a ser possível calcular o caminho mais curto no *smartphone*.

Para rastrear o local onde o *smartphone* se encontra é utilizado o sinal emitido pelos *beacons*(ligação verde).

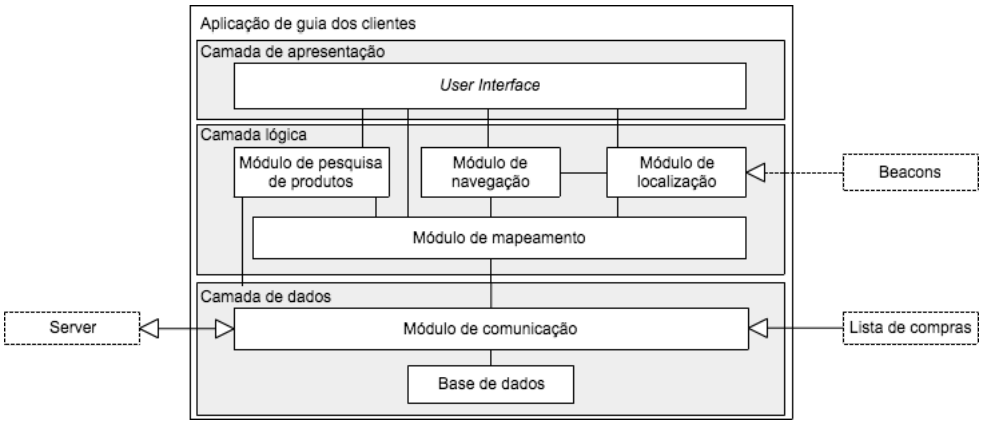


Figura 5.4: Arquitetura detalhada da aplicação de guia dos clientes

Para o funcionamento correto desta aplicação 5 diferentes módulos são necessários, como se pode observar na figura 5.4.

Módulo de mapeamento Este é responsável pelo mapeamento do local como produtos, prateleiras e posicionamento dos *beacons*.

Módulo de localização Este é responsável por estimar a posição do utilizador. Este está constantemente à escuta de *beacons* envolventes, de forma

a estimar a posição do utilizador. Esta informação é depois trocada com o módulo de navegação e com a *user interface*, de forma a mostrar a posição do utilizador.

Módulo de navegação Este módulo gere a navegação do utilizador, calculando o percurso mais curto e indicando qual a direção que o utilizador deve tomar. Este comunica com o módulo de localização de forma saber qual a posição do utilizador em cada momento.

Módulo de comunicação Este módulo é responsável por gerir informação.

Sempre que a lista de compras é enviada para a aplicação, é este o módulo responsável por analisa-la. A informação recebida deve conter a quantidade e o id de cada produto presente na lista de compras do utilizador, sendo feito um *resquest* ao servidor de forma a obter a posição do produto no mapa.

Informação em relação ao mapeamento do local deve ser guardada na base de dados de forma a minimizar o uso de Internet. Assim, quando a aplicação é iniciada apenas é feito um *request* ao servidor, que retorna a data da última alteração, de forma definir se a informação existente na base de dados é válida. Caso não seja, é iniciado um processo de sincronização.

Módulo de pesquisa de produtos Este é usado quando o utilizador pretende pesquisar e saber detalhes de um produto. Este é o principal módulo da aplicação dos funcionários, não tendo sido desenvolvida neste estágio.

5.4 Arquitetura da Aplicação dos Funcionários

Observando a figura 5.5 podem observar-se alguns detalhes da arquitetura da aplicação dos funcionários.

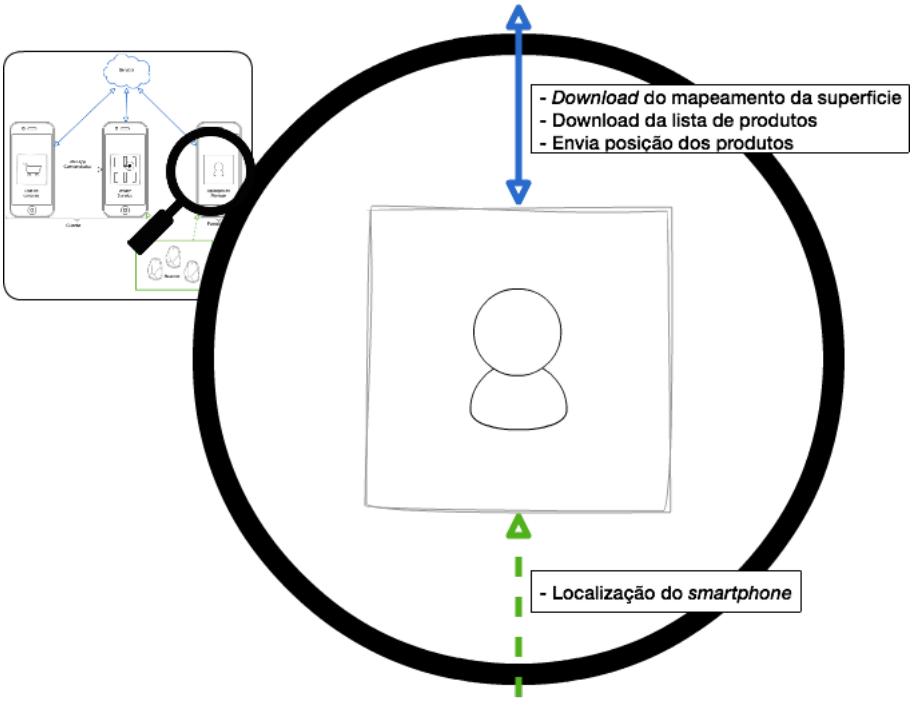


Figura 5.5: Arquitetura do sistema da aplicação dos funcionários

Esta aplicação é responsável por introduzir no sistema as posições dos produtos. Para isso é necessário ter acesso à posição em que o *smartphone* se encontra a cada momento. Com esta finalidade, é efetuado *download* do mapa da superfície alojado no servidor (ligação azul) e este é sobreposto com a posição do utilizador estimada a partir dos sinais dos *beacons* (ligação verde) tal como acontece na aplicação de guia dos clientes, podendo o módulo de localização e mapeamento ser partilhado.

Uma vez que necessitamos de conseguir identificar produtos, é efetuado um *download* do servidor da lista de produtos, com detalhes do código de barras

de cada item, que permite identificar produtos facilmente. Após os identificar, a localização é enviada para o servidor para posteriormente ser usada pela aplicação de guia dos clientes.

5.5 Servidor

O servidor servirá como *backend* das aplicações móveis. Como visto anteriormente este será responsável por:

- Enviar o mapeamento do local
- Guardar e enviar a lista de compras do utilizador
- Guardar e enviar informação dos produtos

Apesar de estar fora do âmbito do estágio o desenvolvimento do servidor, é importante definir qual o formato dos dados que o servidor deve responder. Para facilitar o interpretação da informação foram apenas consideradas formatos que têm suporte nos vários sistemas operativos móveis. Isto restringe as escolha das possíveis soluções entre *JavaScript Object Notation* (JSON) e *eXtensible Markup Language* (XML).

Foi definido o uso de JSON por ter menos *overhead* de informação quando comparado com *XML*. Assim menos tráfego será gasto, e mais rápida será a troca de informação com o servidor.

Capítulo 6

Implementação

A responsabilidade do estagiário é a implementação da aplicação de guia dos clientes, estando nas secções seguintes documentada a implementação realizada.

6.1 Aplicação Móvel

Após o estudo efetuado no estado da arte dos diferentes tipos de aplicações móveis existentes (nativas, híbridas e web), é importante definir qual o tipo de aplicação que será desenvolvida e para que plataforma. Como referido no estado de arte, não existe uma melhor abordagem em relação a outra, dependendo sempre do projeto e dos conhecimentos da equipa de desenvolvimento.

No caso especifico deste projeto, é necessário aceder aos pacotes recepcionados pelo *bluetooth*.
As aplicações Web não são apropriadas para este projeto, uma vez que não é possível aceder estes pacotes, devido ao facto de não haver uma API standard

que o permita fazer a partir do browser,

No caso das aplicações híbridas, uma vez que são instaladas no *smartphone*, está limitação pode ser ultrapassada, dependendo da ferramenta utilizada. Uma das ferramentas mais populares para este tipo de aplicações é o Cordova por ser gratuita, open-source e ser utilizada como o núcleo de funcionamento de outras ferramentas para o mesmo efeito, como Ionic e PhoneGap. Apesar de nenhum destes suportarem nativamente o uso de *bluetooth*, o Cordova permite a instalação de plugins de terceiros, que são suportados pela comunidade e que nos permite aceder às APIs de Bluetooth.

Nas aplicações nativas, se usarmos os SDKs de cada um, não existe qualquer tipo de limitação, sendo que, tanto Apple como o Android, dão suporte providenciando APIs para o efeito. No caso da Apple estes têm uma *framework* que auxilia o uso da informação enviada pelos iBeacons, sendo facilitado o acesso ao valores como o UUID, *Major Value* e *Minor Value*, que nos permitem identificar o *beacon* em questão. Já no caso do Android, a extração desta informação tem que ser realizada pelo programador, uma vez que não é dado suporte até à versão atual (Android 5.1).

Devido à complexidade dos cálculos necessários tanto para estimar a localização como para a geração do percurso mais curto, devemos optar pela solução que nos garante melhor performance e eficiência na gestão de memória, campos onde o uso dos SDK nativos são a melhor solução. Por isso optou-se pelo uso de aplicações nativas em detrimento das aplicações híbridas, desenvolvida pelos SDKs nativos.

Uma vez que há um limite temporal para o desenvolvimento desta aplicação, o primeiro protótipo desta aplicação, será desenvolvido para iOS, por ter suporte para os iBeacons, tendo as APIs e *frameworks* para auxiliar o seu uso.

6.2 Linguagem de Programação

Para desenvolver para iOS usando o SDK nativo, podemos programar em Objective-C ou Swift.

O Objective-C é uma linguagem baseada em C mas orientada a objetos. Esta foi usada e licenciada pela NeXT, que após ser comprada pela Apple, tornou-se a base do operativo Mac OS X. Assim, desde 1996, este tem sido a linguagem usada para se desenvolver para o Mac OS X, e atualmente para o iOS. Ao longo deste tempo estes foram-na evoluindo adicionando-lhe novas *features* como *closures* e *Automatic Reference Counting* (ARC). Em Junho de 2014, a Apple anunciou o Swift, como sendo "o melhor do C e do Objective-C mas sem as limitações de compatibilidade do C". Isto permitiu criar uma linguagem de raiz, com o intuito de ser mais rápida, mais resistente a erros e fácil de programar. Uma vez que ambas as linguagens usam o compilador LLVM, permite-nos correr código Swift, Objective-C, C e C++, no mesmo programa sem qualquer problema [52]. Por esta razão, não existe qualquer limitação em relação às APIs atuais, ou a bibliotecas de terceiros, programadas em Objective-C, sendo que estas são compatíveis com a nova linguagem.

Apesar de não ser expectável o fim das aplicações escritas em Objective-C nos próximos anos, devido às inúmeras aplicações atualmente escritas com esta, é expectável que Apple diminua o suporte para estes, forçando-os a portar as suas aplicações.

Devido ao facto do swift ser uma nova linguagem com características interessantes e sendo a mais promissora olhando para o futuro, optou-se por programar a aplicação nesta linguagem.

6.3 Módulo de mapeamento

Este módulo é o responsável pela informação em relação ao espaço físico, tendo várias responsabilidades.

6.3.1 Gerar o Mapa

Este módulo interage com o de comunicação de forma a obter a posicionamento das prateleiras, e o tamanho do espaço. Com esta informação este gera uma UIView, onde cada prateleira é outra UIView. Sempre que o utilizador toca numa destas views, é despertado um evento onde é mostrada informação relativa a esta prateleira.

6.3.2 Mapear produtos

Semelhante à forma como é gerado um mapa, sempre que é importada a lista de compras do utilizador, este módulo cria uma UIImageView para cada produto. Esta classe herda todas as propriedades do UIView mas acrescenta a possibilidade de mostrar imagens. Existe igualmente um tratamento de eventos para quando um utilizador pressiona num produto, sendo mostrado o nome do produto mas dando a possibilidade do utilizador eliminá-lo. Caso o utilizador decida eliminar um produto, é enviado este evento para o módulo de navegação, de forma a adaptar o percurso de acordo com este novo evento.

6.3.3 Grafo

De forma a ajudar a estimar a posição do utilizador e de forma a permitir a geração do percurso mais curto, é gerado uma grafo. Este grafo representa o espaço onde o utilizador pode andar.

As componentes deste grafo, como a posição dos nós, e os nós a que cada ligação liga, provém do módulo de comunicação que por sua vez provém do servidor. Uma vez que esta informação é estática, no sentido em que apenas quando houver uma alteração ao mapa será necessário refazer o grafo, não há necessidade de ser o *smartphone* a realizar este cálculo.

Na figura 6.1 pode ver-se o grafo criado.

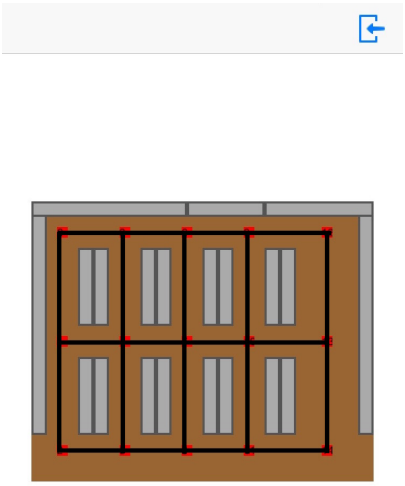


Figura 6.1: Grafo do local

6.4 Módulo de Localização

O módulo de localização é responsável por estimar a posição do utilizador na súperficie comercial. Nesta secção está documentado o seu funcionamento e implementação.

6.4.1 Decisões

Tecnologia

Segundo a analise da secção 2.3.7, pode-se observar que apenas o Wi-Fi e o *bluetooth* são os únicos que satisfazem todos os paramentos analisados.

Estes são muito semelhantes, em relação à margem de erro média (2 metros) e implementação. Para o uso destas tecnologias para realização de localização dentro de edifícios é tirado partido das sinais emitidos por cada um. Daí advém o problema de quanto maior for o número de utilizadores numa determinada secção, maior será o erro, devido à perturbação no sinal por parte de cada um. Apesar das suas semelhanças existem alguma diferenças e limitações. Como descrito em 2.3.6, o Wi-Fi tem o problema de o iOS não permitir aceder a informação das redes Wi-Fi envolventes, tornando uma implementação deste tipo menos escalável, devido à necessidade de ter um servidor responsável por estimar a posição do utilizador. Em relação à precisão, segundo [53] o BLE é mais preciso em 27% quando comparado com o Wi-Fi.

O BLE apresenta-se como a melhor solução para realizar *indoor location*. É dos mais precisod, cómodo de usar e é compatível com os vários sistemas operativos móveis, sendo apenas ligeiramente menos escalável quanto comparado com o

uso de sensores existentes no *smartphone*. Este ponto poder ser debatível, devido ao facto de existirem vários tipos/marcas de sensores em cada telemóvel com erros variáveis. Assim sendo pode ser necessário haver uma calibração para os diferentes dispositivos.

Apesar de ser um requisito o uso de BLE para realização de localização dentro de edifícios, como visto no estado da arte, este apresenta-se como uma solução para realizar esta tarefa.

Algoritmo

Estando definida a tecnologia que será usada, é importante definir qual o algoritmo indicado para estimar a posição do utilizador. Com este intuito, pode-se considerar os algoritmos de *fingerprinting*, lateração e *Cell Based*.

Uma vez o *Cell Based* é apenas capaz de identificar uma zona, e não uma posição, este não vai de encontro ás necessidades do sistema, sendo portanto excluído.

Para escolha do algoritmo mais adequado para localização deve-se ter em conta:

Escalabilidade de utilizadores Como é que o sistema reage em relação ao número de utilizadores que simultaneamente usam a plataforma

Escalabilidade de espaço Quão complexa é a implementação em espaços de diferentes dimensões.

Precisão Quão preciso é o algoritmo.

Performace Eficiência a nível energético

Em relação à escalabilidade de utilizadores, estas abordagens são independentes do número de utilizadores a usar o sistema na medida em que o processo de identificação da posição é efetuada no dispositivo do utilizador. Mas ambos os algoritmos sofrem do problema de quanto maior o número de pessoas num mesmo espaço, maior é a degradação do sinal e por consequência maior o erro da posição estimada.

Em algoritmos com fase de aprendizagem, quanto maior for o espaço físico de implementação maior será o número de *fingerprints* necessárias, que apesar de fácil é demoroso. No caso da lateração apenas é necessário a introdução das posições dos *beacons* no sistema.

Numa abordagem baseada em lateração é expectável um erro de cerca de 3 metros [54]. Por sua vez uma abordagem baseada em *fingerprint* o erro é 2-3 metros [25]. Mas para ser possível ter margens de erros nesta gama, foram efetuadas 200 *fingerprints* para um espaço com 600 metros quadrados. Se tivermos conta que uma superfície comercial tem 14 000 metros quadrados, facilmente percebe-se, que é difícil manter este tipo de precisão, na medida em que são necessárias milhares de *fingerprints*.

Algoritmo baseados em *fingerprinting* tendem a ter uma maior complexidade computacional e espacial quando comparados com algoritmos baseados na lateração [6][24], que por consequência, aumenta o gasto de energia. Isto porque os cálculos necessários são relativos ao número de fingerprints, ao contrário da lateração que está simplesmente relacionada com o número de beacons que o *smartphone* deteta em determinada posição.

A tabela 6.1 resume a análise comparativa feita nesta secção.

Após esta análise, optou-se pelo uso da lateração, pois apresenta ser o algoritmo

Campo	Lateração	*Fingerprint*
Escalabilidade de utilizadores	✓	✓
Escalabilidade de espaço	✓	
Precisão		✓
Performace	✓	

✓- O melhor para o campo analisado.

Tabela 6.1: Comparação entre a Lateração e *Fingerprinting*

mais adequado.

6.4.2 Implementação da Lateração

Como dito anteriormente, o método escolhido para estimar a posição do utilizador foi a lateração. Como explicado em 2.2.1, para o uso deste algoritmo necessitamos de estimar a distância que os dispositivos se encontram dos pontos âncora, que neste caso são os *beacons*. Para isso foi usado a framework CLLocation que estima a distância aos *beacons*. Esta framework tira partido do protocolo *iBeacon*, usando o TX Power de cada mensagem para estimar a distância. Este retorna um valor em metros a que o beacon se encontra do dispositivo.

Existem duas formas de realizar lateração, fazendo trilateração ou multilateração. A trilateração, tem apenas em conta os valores em relação a 3 pontos âncora, e usa a interceção dos círculos com centro nos beacons e raio igual à distância estimada a eles, como sendo a posição do utilizador, tal como mostrado e explicado em 2.2.1. Mas uma vez que a estimativa da distância aos beacons tem um erro envolvido, é possível que os círculos não se intercetam só num único ponto, como em 6.2b, resultando em infinitas soluções diferentes.

Para resolver este problema tira-se proveito da multilateração onde podem ser usados um número ilimitado de *beacons* para estimar a posição do utilizador,

onde o objetivo é minimizar o resíduo 6.2b.

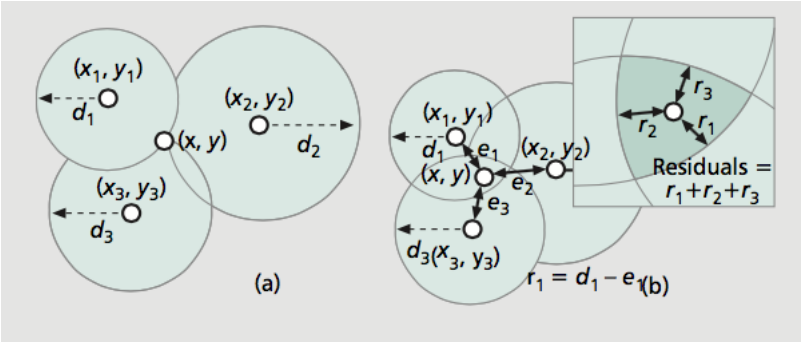


Figura 6.2: Triangulação vs Multilatereção [6]

Para estimar a posição do utilizador é usado um método de mínimos quadrados (least squares), que tem por objetivo resolver sistemas de equações sobre determinados. Existem duas categorias deste método, linear ou não linear. Yang, Jie e Chen, Yingying indicam que a solução não linear é 29 % [54] mais eficiente que a linear, tendo sido este o algoritmo escolhido.

Dada a distância estimada aos beacons d_i e as posições de cada um (x_i, y_i), a posição do utilizador pode ser estimada, usando o método não linear dos mínimos quadrados, descobrindo $(\hat{x}, \hat{y})$ que satisfaça:

$$(\hat{x}, \hat{y}) = \arg\min_{x,y} \sum_{i=1}^{N} [\sqrt{(x_i - x)^2 - (y_i - y)^2} - d_i]^2$$

Para resolver este sistema foi usada uma livraria chamada Eigen [1]. Esta biblioteca foi desenvolvida em c++, sendo possível compilar em iOS.

[1]Eigen: `http://eigen.tuxfamily.org/`

Testes

De forma a avaliar a margem de erro do algoritmo de localização foram realizados testes em dois ambientes distintos:

1. Local com variados obstáculos e tipos de interferências (redes Wi-Fi, e outros dispositivos eletrónicos)
2. O armazém dos Serviços de Ação Social da Universidade de Coimbra (SASUC).

Para a realização destes testes foram espalhados 6 *beacons* pela superfície e escolhidos 3 pontos de forma aleatória. O objetivo destes testes é analisar a exatidão (*accuracy*) e precisão (*precision*) do algoritmo, isto é, qual a distância do ponto estimado ao ponto esperado e quão dispersos estão as posições estimadas. Com este intuito, em cada ponto, foram realizadas 101 estimativas da posição do utilizador. No caso do teste 1 o espaço total mapeado foi de 43 metros quadrados, no teste 2 foi 60 metros quadrados.

As figuras 6.3 e 6.4 mostram os testes realizados relativos á exatidão do algoritmo algoritmo no local 1 e 2, respetivamente. No teste 1 foram usadas as posições (1,4), (2,2) e (6,5), obteve-se uma exatidão média de 0.66m, 1.59m e 1.09m e um desvio padrão de 0.18m, 0.23m e 0.29m respetivamente. No global a exatidão do algoritmo para este espaço é de 1.11m com um desvio padrão de 0.45m e com um erro máximo de 1.92m.
Para o teste 2 foram usadas as posições (1,5), (3,1) e (4,5) onde se obteve uma precisão média de 0,55m, 0.74m e 0.43m com um desvio padrão 0.41m, 0.42m e 0.2m respetivamente. No global a exatidão média é de 0.61m com um desvio padrão de 0.37m, com um erro máximo neste espaço foi de 2.27m.

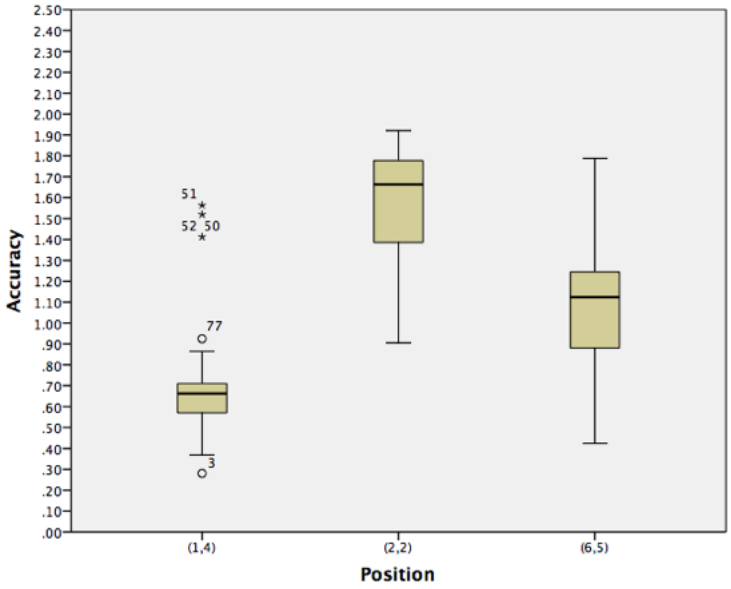


Figura 6.3: Teste 1 - Exatidão

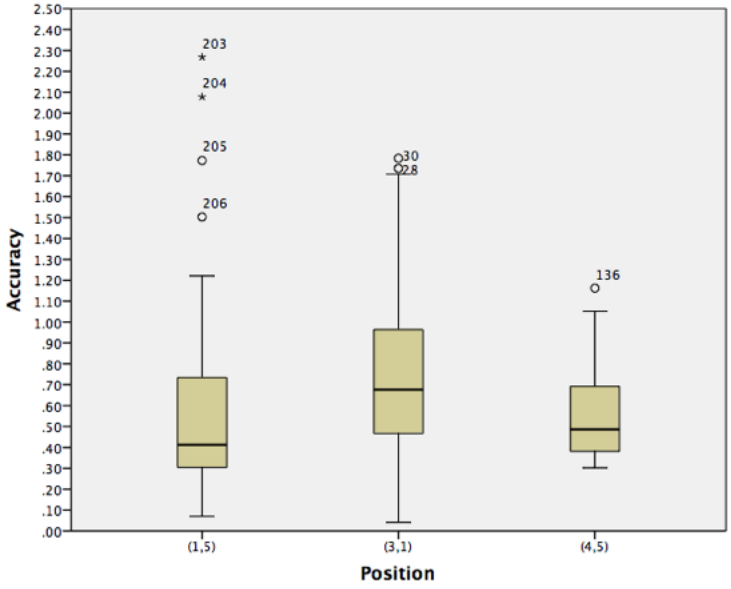


Figura 6.4: Teste 2 - Exatidão

Como se pode observar, no teste 2 obteve-se um baixo erro médio (0.61), muito inferior ao teste 1 (1.11). Existem duas razões para esta diferença:

- O espaço 2 é um espaço amplo, onde o *multipath* é mais baixo, isto é, existe menos reflexão e refração dos sinais emitidos pelos beacons.
- No espaço 1, existe um maior interferência de outras redes tanto Wi-Fi como *Bluetooth*.

Num espaço real é expectável uma margem de erro mais próximas do espaço 1, pois existem várias prateleiras e pessoas a interferir com o sinal.

As figuras 6.5 e 6.6, mostram os testes realizados relativos á precisão do algoritmo algoritmo no local 1 e 2.

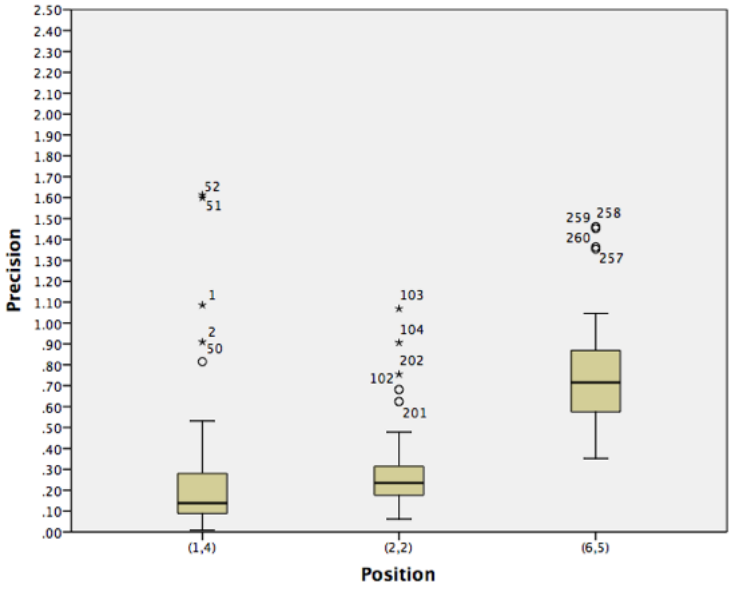


Figura 6.5: Teste 1 - Precisão

No teste 1 para as posições (1,4), (2,2) e (6,5), obteve-se uma precisão média de 0.23m, 0.27m e 0.74m e um desvio padrão de 0.26m, 0.16m e 0.22m respetivamente. No global a precisão do algoritmo para este espaço é de 0.41m com

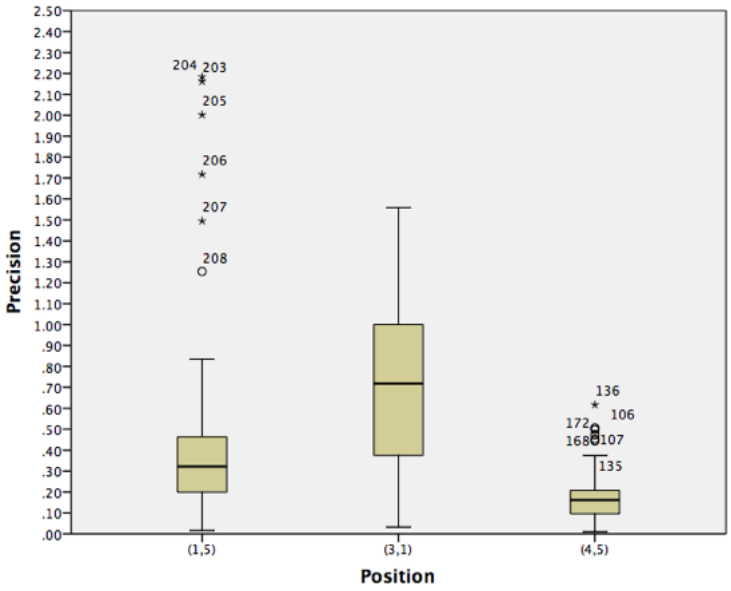


Figura 6.6: Teste 2 - Precisão

um desvio padrão de 0.32m.

Para o teste 2 foram usadas as posições (1,4), (2,2) e (6,5), onde obteve-se uma precisão média de 0.42m, 0.71m e 0.17m e um desvio padrão de 0.41m, 0.40m e 0.17m respetivamente. No global, a precisão do algoritmo para este espaço é de 0.44m com um desvio padrão de 0.40m.

Pode-se observar que este algoritmo, tende a ter a ser preciso (aproximadamente 40cm). Isto indica que, no mesmo sitio, não vai haver grandes oscilações na posição.

No global este valores mostram ser promissores, e validam que o BLE, pode ser usado como um tecnologia para realização de localização dentro de edifícios.

6.4.3 Otimização

O algoritmo anteriormente descrito devolve um valor provável da posição onde o utilizador se encontra. Mas a margem de erro pode chegar a 2 metros o

que pode levar a indicar que o utilizador está numa posição inválida como por exemplo em cima de uma prateleira.

O estagiário debateu-se com este problema de forma a tentar minimiza-lo.

Uma solução simples, passaria simplesmente por descartar uma posição que corresponde-se à prateleira, mas os testes mostraram que as leituras tendem a ser precisas mas não exatas, isto é, caso o dispositivo esteja no mesmo sitio, este tende sempre a mostrar a mesma posição, apesar de incorreta. Isto leva a que fosse muito provável que várias leituras consecutivas fossem descartadas.

Assim uma solução desse género não será a mais indicada. A solução desenvolvida parte do princípio que é pouco relevante para o utilizador saber se está mais à esquerda ou à direita dentro de um determinado corredor, este apenas está interessado em saber qual a sua posição no espaço global.

A solução passa por mapear a posição do utilizador no grafo que corresponde ao espaço em que o utilizador pode estar.

Na figura 6.7, está representado um grafo e a posição do utilizador representada pelo círculo "U", sendo A e B as ligações aos nós 4-5 e 2-5 respetivamente.

São realizados os seguintes passos:

1. Começa por encontrar o nó mais perto, sendo que neste caso é o nó 5.
2. Após encontrar este nó, projeta a posição do utilizador nos ligações desse nó. Este pontos estão representados por A' e B'
3. É calculada a distância entre a posição estimada e o utilizador (circulo U).

4. Calcula-se a distância do nó a A' e B'. O que tiver um valor mais próximo do valor calculado em 3) é a posição escolhida (circulo a tracejado).

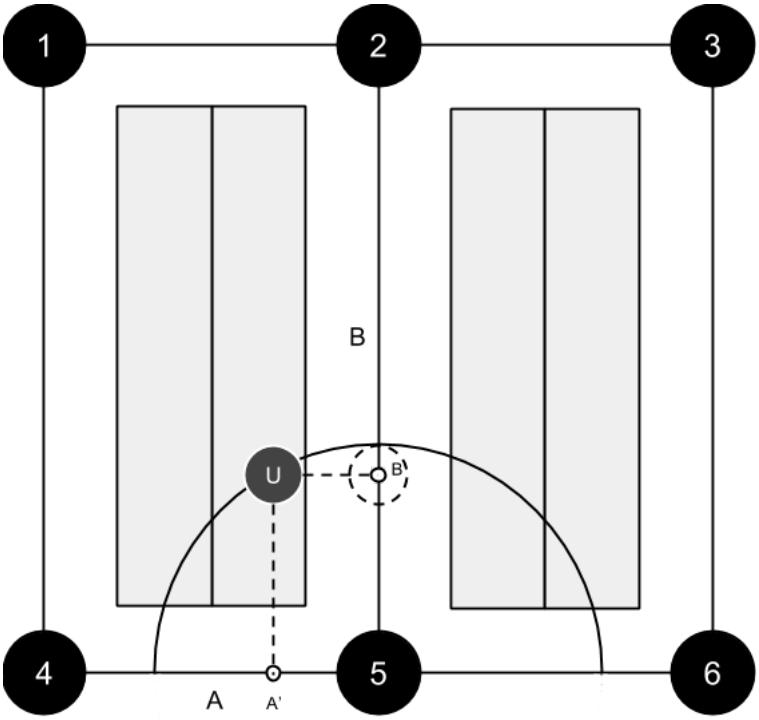


Figura 6.7: Posição do utilizador no grafo

6.5 Módulo de Navegação

Este módulo é responsável por gerar e guiar o utilizador pelo percurso que deve percorrer de forma a recolher todos os produtos percorrendo o menor espaço possível.

6.5.1 Introduzir produtos no grafo

Este módulo começa por introduzir os produtos da lista de compras no grafo construido no módulo de mapeamento. Para realizar esta ação são realizados

os seguintes passos:

1. Cada produto tem uma referência da prateleira onde se encontra. Por sua vez cada ligação no grafo, têm uma referência às prateleiras que lhe estão adjacentes. Desta forma temos acesso à conexão adjacente a determinado produto.

2. Calcula-se a direção da conexão anterior, de forma a saber se a conexão se encontra na vertical ou horizontal.

3. Segundo essa direção é calculada a posição esperada do novo nó, de forma a ficar entre os nós da conexão.

   (a) Caso não exista nenhum nó nessa posição:

      i. Cria-se um novo nó

      ii. Elimina-se a ligação

      iii. Criam-se novas ligações entre o novo nó, aos nós anteriormente ligados pela conexão (figura 6.8.A).

   (b) Caso exista um nó nesta posição: É apenas adicionada a referência ao produto nesse nó.

   (c) Caso essa posição esteja num canto: É adicionado o produto no nó adjacente (figura 6.8.B).

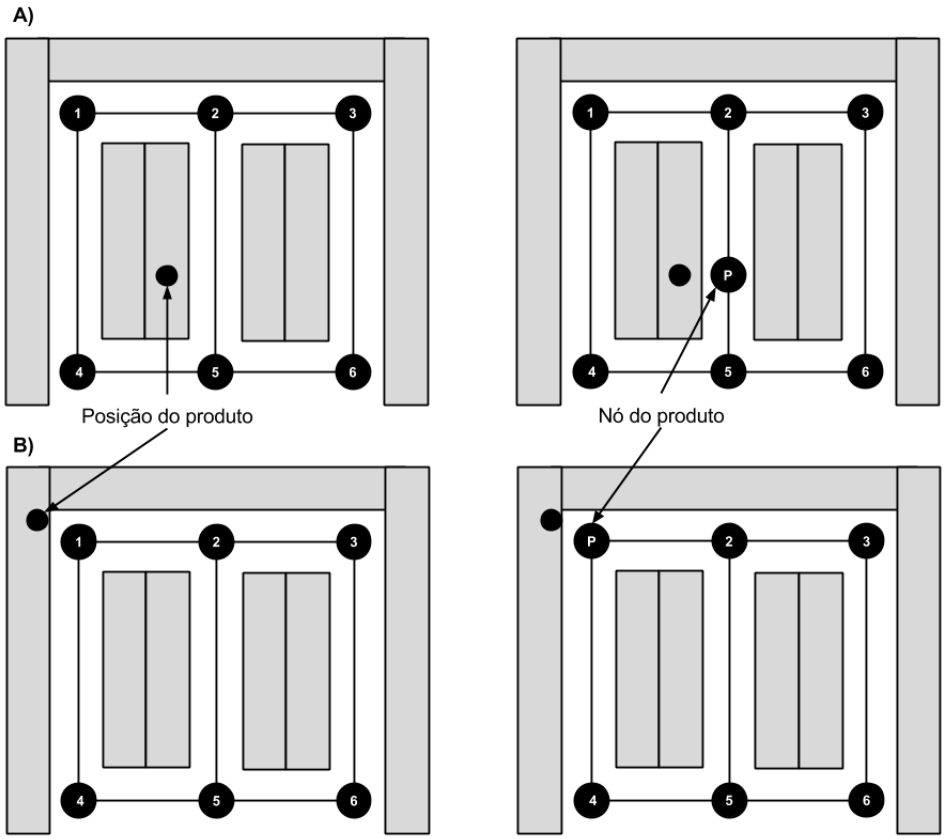


Figura 6.8: Adicionar produto no grafo

6.5.2 Ordem a recolher os produtos

Para a geração do percurso mais curto, é necessário saber qual a ordem pela qual devemos recolher os produtos presentes na lista de compras.

Este problema é semelhante ao problema do caixeiro viajante. Aqui tenta-se resolver o problema: Dado uma lista de cidades, qual é o percurso mais curto possível de forma a visitar uma cidade uma e apenas uma vez?

Para a resolução deste problema, podem ser considerados dois tipos de algoritmos, os algoritmos exatos e os algoritmos baseados na heurística.

Para os algoritmos exatos, necessitamos de saber a distância real entre os vários

produtos. Isto implicaria usar um algoritmo de forma a descobrir o caminho mais curto entre os vários produtos, o que se torna impossível de realizar, num curto espaço de tempo. Imaginemos que temos 15 produtos na lista de compras, necessitamos de calcular 15^2 percursos mais curtos. E para calcular cada percurso teríamos que usar um algoritmo como o de Dijkstra que tem uma complexidade temporal de $O(V^2)$.

Assim sendo, optou-se por um algoritmo baseado na heurística. Mais precisamente no *nearest neighbor algorithm*. Optou-se por este algoritmo devido à baixa complexidade temporal, $O(log(V))$, sendo V o número de vértices, neste caso o número de produtos na lista de compras.

O funcionamento deste algoritmo é simples:

1. Inicia-se no ponto onde o utilizador se encontra (x,y)
2. Descobre qual o nó não visitado mais perto desse ponto
3. Avança para esse nó
4. Marca o nó como visitado
5. Volta ao ponto 2, até todos os nós terem sido marcados como vistos.

A heurística usada, é a distância em linha reta entre os nós.

Testes

Facilmente se observa que este algoritmo não é ótimo, isto é, não é possível garantir que a solução deste algoritmo será a melhor.

Para testar qual a margem de erro médio deste algoritmo, foi programado uma algoritmo *Bruteforce*, de forma a testar todas as combinações possíveis, e ordem ótima baseada ma baseada na heurística.

Foram realizados 3 testes onde se insere 5, 10 e 15 produtos de forma aleatória. Cada teste foi corrido 30 vezes.

Em média o *nearest neighbor algorithm* indicou um caminho 8% maior que a solução do *bruteforce*, com um desvio padrão de 8.9%.

O tempo de execução para o teste mais pesado (15 nós) o *nearest neighbor algorithm* executa em 0.0003 segundos com contra os 3948 segundos (aproximadamente 1h e 06 minutos) do *Bruteforce*, apesar de algumas otimizações efetuadas.

Apesar destes valores terem relevância relativa pelos baixo número de testes realizados é possível perceber que a margem de erro será relativamente baixa, para a maioria dos casos, tendo um bom compromisso entre a margem de erro e o tempo de execução.

6.5.3 Geração do percurso

Após saber a ordem com que os produtos devem ser recolhidos, é necessário saber o caminho de como chegar a estes.

O algoritmo escolhido para realizar este trabalho foi o A*, por ser dos algoritmos mais eficientes para descobrir o caminho mais curto [55]. É por esta razão muito usado em aplicações como videojogos.

Este algoritmo é ótimo, caso a heurística seja admissível. Uma vez que a nossa heurística é a distância em linha reta entre nós, podemos garantir que

a heurística é admissível e por sua vez, este retorna sempre o percurso mais curto, entre 2 pontos.

O pseudocódigo do algoritmo implementado pode ser visto na figura 6.9

```
// f = heurística + custo
Inicializa-se uma lista vazia de nós abertos
Inicializa-se uma lista vazia de nós fechados
Adiciona-se o nó inicial aos nós abertos

Enquanto a lista de nos abertos não estiver vazia{

    nóAtual = nó com menos f na lista de nós abertos
    remove-se nóAtual da lista de nós abertos

    Se o nóAtual == nó final{
        fim (reconstruir caminho)
    }

    Para todos os nós vizinhos ao nóAtual{
        sucessor = vizinho
        sucessor.custo = nóAtual.custo + distância
          entre o vizinhos e o nóAtual
        sucessor.heurística = heurística ao nó final
        sucessor.pai = nóAtual

        Se a lista de nós abertos ou a lista dos
        fechados tiver um nó na mesma posição e
        com menor f que o nó sucessor)
            passa este sucessor

        Adiciona sucessor a lista de nós abertos

    }
    adiciona o nóAtual á lista de nós fechados
}
```

Figura 6.9: Pseudocódigo do Algoritmo A*

6.5.4 Otimização do percurso

Com o uso do *nearest neighbor algorithm* e usando a heurística como sendo a distância em linha reta entre os produtos, existem várias limitações.

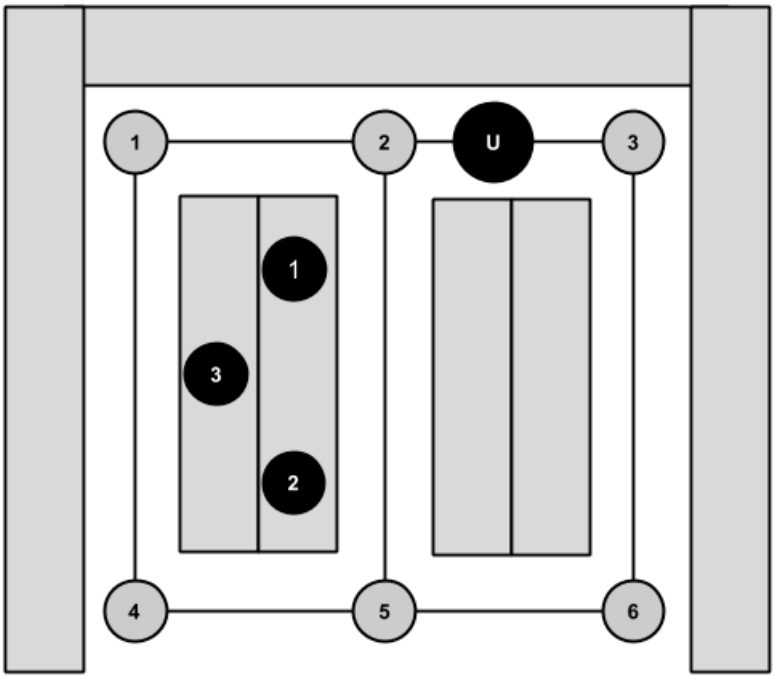


Figura 6.10: Cenário 1

Na figura 6.10 está representada uma superfície comercial onde o utilizador esta marcado com o circulo "U" e os produtos com círculos pretos. Se aplicarmos o algoritmo *nearest neighbor algorithm* nesse cenário, este gera o percurso 1-3-2, uma vez que a distância em linha reta entre 1 e 3 é menor que a de 1 e 2, o que significa estar numa secção ter que ir a outra, e volta á secção anterior.

De forma a mitigar este problema, sempre que um nó for selecionado é realizada uma pesquisa em profundidade, para descobrir a existência de mais nós nessa secção. Esse produtos são ordenados por ordem crescente, segundo a sua posição, e é comparado a distância do mais pequeno e do maior, de forma a

descobrir se é mais vantajoso ir para o nó que está mais em baixo na prateleira ou em cima.

Isto por si, já melhora o comportamento do percurso gerado, mas continua com alguns problemas. Se aplicarmos este algoritmo, no cenário da figura 6.11, a ordem gerada será 1-3-2, por o 1 estar mais próximo do 3 e este ser o último da secção. Para corrigir está falha, sempre que um percurso gerado passar por um produto que ainda não tenha sido recolhido pelo utilizador, será gerado o percurso apenas até este. Sendo que neste caso ficaria o percurso 1-2-3.

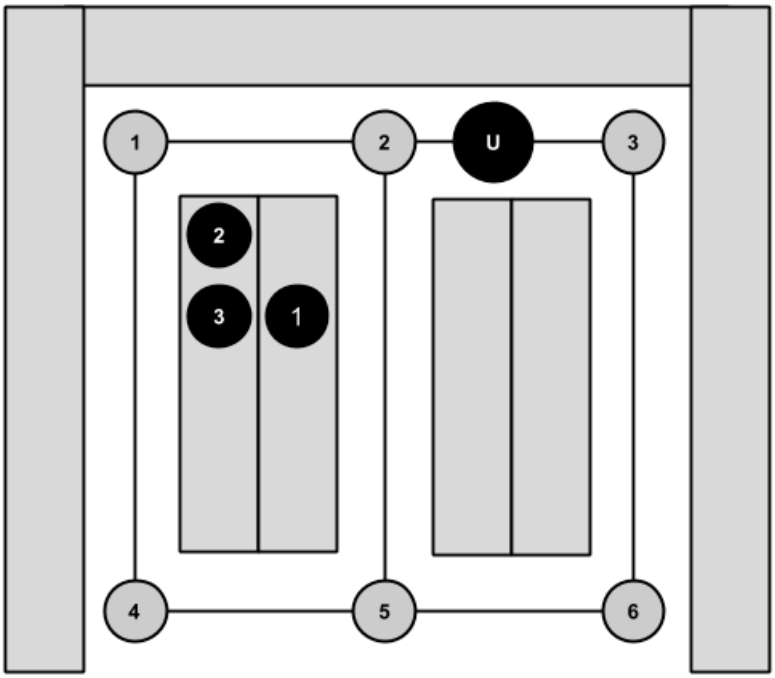


Figura 6.11: Cenário 2

Apesar desta solução, continuar a não garantir uma solução ótima, tem um comportamento mais perto do esperado e realizado habitualmente pelos clientes das superfícies comercias.

Testes

Este teste foi realizado com o intuito de saber quão eficiente é o algoritmo usado relativamente à solução ótima, a qual consiste na minimização do espaço percorrido, e se a otimização realizada melhora a solução.

Para o cálculo da solução ótima foi programado um algoritmo *bruteforce* no qual foi utilizada a distância real entre os produtos e devolvido o melhor percurso possível. Para os casos nos quais se utiliza o *nearest neighbor algorithm* e a otimização efetuada é primeiro calculada a ordem dos produtos (usando estes algoritmos) e, posteriormente, usado o A* de forma a gerar o percurso completo.

Cada teste consistiu na geração do percurso para 10 produtos colocados de forma aleatória, executado 50 vezes. Comparando a solução ótima com o *nearest neighbor algorithm* este gerou, em média, um percurso 11,49% maior que o percurso ótimo, contudo um desvio padrão de 8.19%. Para o caso da otimização realizada o teste gerou um percurso 9.01% maior que a solução ótima com um desvio padrão de 7.25%.

Uma vez que não é possível obter, em tempo real, o percurso ótimo considera-se o valor obtido satisfatório.

6.5.5 Guia

De forma a permitir indicar as ações que o utilizador deve tomar são necessárias duas informações, tais como, a posição do utilizador, que é providenciada pelo módulo de localização, e a orientação do utilizador relativamente ao espaço. Para possibilitar o cálculo desta orientação cada mapa da superfície comercial

possui informação relativa à sua orientação cardeal. Para saber a orientação do utilizador é acedida à informação do magnetómetro do *smartphone*. A diferença entre a orientação cardeal do mapa e do *smartphone* indica-nos a orientação relativa do utilizador em relação à superfície comercial onde se encontra.

A figura 6.12 mostra a árvore de decisão usada para indicar as ações que o utilizador deve respeitar de forma a recolher todos os produtos no menor espaço possível.

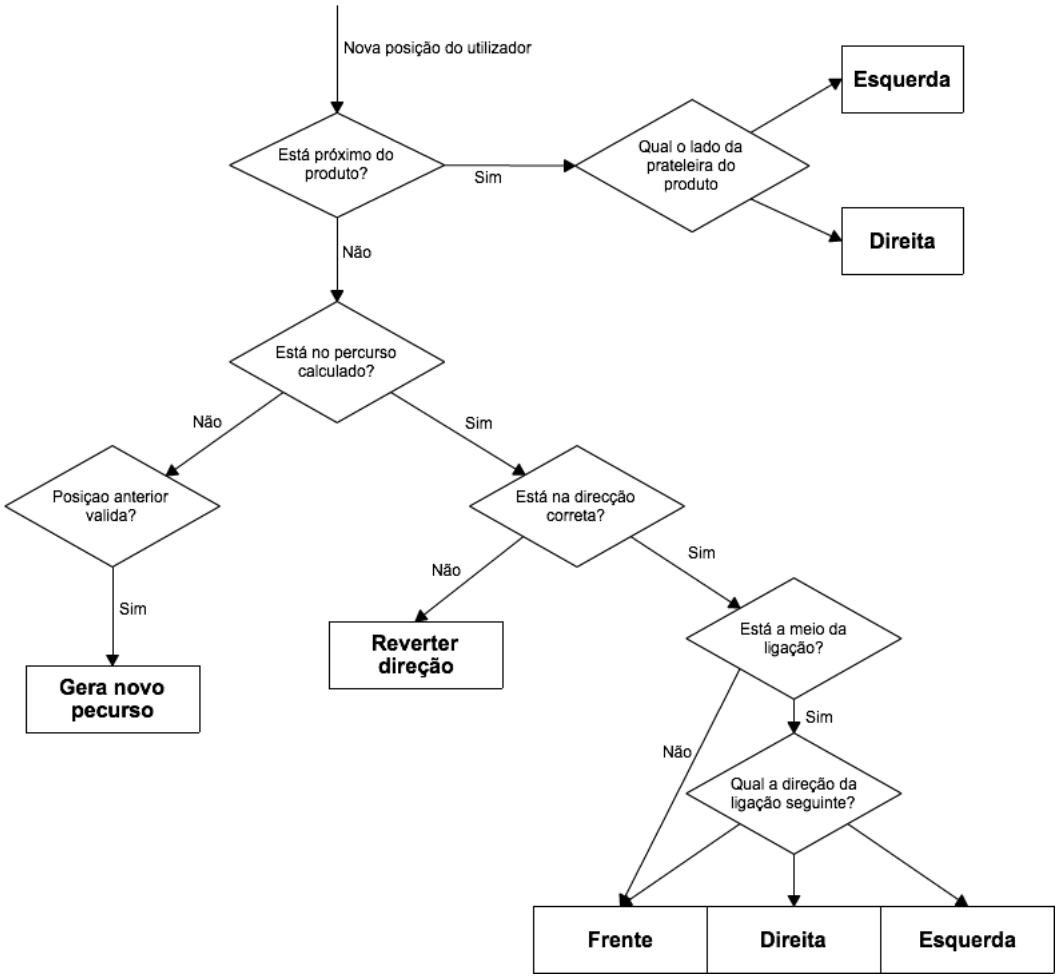


Figura 6.12: Árvore de decisão do guia de navegação

Sempre que uma nova posição do utilizador é calculada, é iniciada esta árvore

de decisão.

Começa-se por analisar se o utilizador está próximo de um produto. É considerado que um produto está próximo caso esteja até dois metros do utilizador ou no mesmo corredor onde o utilizador se encontra. Quando isto acontece é indicado ao utilizador que o produto está à esquerda ou à direita dele.

Caso o utilizador não esteja perto de um produto, é analisado se este está a seguir o percurso previamente calculado. Deste modo, se se verificar que o utilizador se encontra fora da percurso não é logo gerado um novo percurso, pois pode ter ocorrido algum fator externo que provoque um cálculo inválido da posição. Apenas quando duas leituras consecutivas forem consideradas inválidas será gerado um novo percurso.

Caso o utilizador esteja a seguir o percurso previamente calculado é analisado se este está na direção correta, isto é, se ele estiver entre o nó A e B e tiver que ir em direção a B, analisamos se está ou não virado para B. Caso não esteja, é indicado ao utilizador para reverter a sua direção.

Se o utilizador estiver na direção correta é analisado se este está no meio da ligação, que corresponde na realidade a estar no meio de uma secção (entre duas prateleiras), isto para ser possível indicar ao utilizador que deve virar à direita ou esquerda com alguma antecedência de forma inequívoca.
Se não estiver no meio da secção, indicamos que deve continuar em frente, uma vez que está orientado para a direção correta.
Se estiver no meio da secção, é analisada a ligação seguinte no caminho mais curto. É calculado o ângulo que estas ligações formam e, consoante este ângulo, é indicado se deve virar à esquerda, direita ou continuar em frente.

6.5.6 Funcionamento

As várias secções anteriores detalham o funcionamento das várias partes constituintes deste módulo. Nesta secção é explicado como estas se interligam.

Quando o utilizador pretende que a aplicação o auxilie a recolher todos os produtos, é iniciado este modulo através método init(). Esta ação começa por mapear os produtos no grafo. Após isto, é chamado o método orderWithShortestPath() que gera a ordem com que os produtos devem ser recolhidos no menor espaço possível, como descrito na secção 6.5.2. Em seguida é calculado o caminho da posição onde o utilizador se encontra até ao primeiro produto existente na lista de compras (generatePath()). Após isto é chamado o metodo showPath(), para mostrar o percurso ao utilizador e o navigationToProduct(), para iniciar o modo de navegação para este novo caminho gerado. Optou-se por não se gerar o caminho de forma a buscar todos os produtos, devido à elevada probabilidade de o utilizador alterar o seu trajeto, e por sua vez invalidar estes cálculos.

Se tudo correr como esperado, e o utilizador seguir o percurso previamente calculado, este vai recolhendo os produtos pela ordem calculada. Neste caso, sempre que o utilizador indicar que recolheu um produto, é chamado o método gotProduct(). Este método, por sua vez chama generatePath(), caso o produto seguinte não estiver no mesmo nó que o produto atual. Isto para proteger o caso onde o existe mais do que um produto no mesmo nó.

Mas existem duas situações que necessitam de fazer re-calcular a ordem com que os produtos são recolhidos. Isto acontece quando o utilizador elimina um produto (deleteProduct()) ou utilizador está fora da rota (reroute()). Se o utilizador eliminar um produto, existe a possibilidade de a ordem com que os

produtos devem ser recolhidos seja alterada. Para o caso do reroute() acontece o mesmo, devido ao utilizador não estar a cumprir o percurso indicado, e por sua vez poder invalidar a ordem previamente gerada.

Quando isto acontece, uma nova ordem para recolher os produtos é calculada (orderWithShortestPath()) Caso a nova ordem altere o produto atual a ir buscar, o método generatePath() é chamado para calcular o caminho da posição atual até ao novo produto. No caso particular do método rerout(), este irá sempre chamar o generatePath(), de forma a gerar um novo caminho, uma vez que o utilizador está fora do percurso previamente gerado.

A figura 6.13 demonstra o fluxo entre estes métodos.

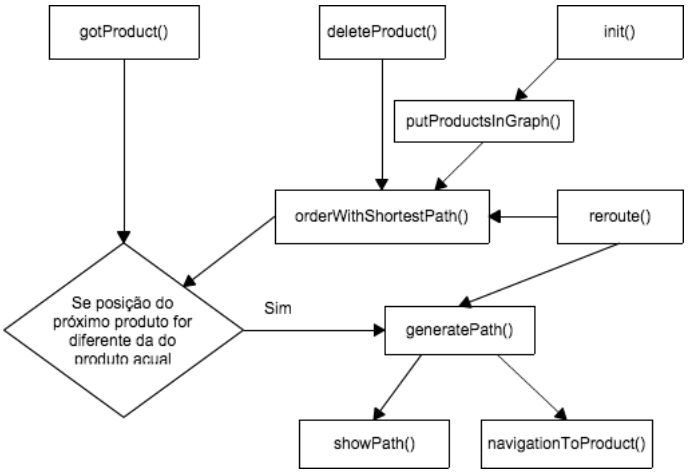


Figura 6.13: Fluxo dos métodos do guia de navegação

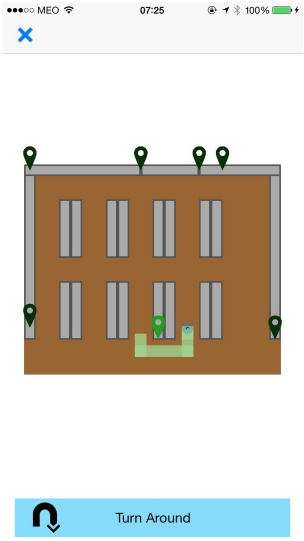


Figura 6.14: Aplicação no guia de navegação

6.6 Módulo de comunicação

Este módulo é o ponto de acesso da aplicação ao mundo exterior, sendo usado para comunicar com o servidor ou receber informação com a lista de compras do utilizador.

Nesta secção é detalhada a implementação efetuada.

6.6.1 Comunicação com outra aplicação

Para possibilitar outra aplicação comunicar com a aplicação fora do seu próprio ecossistema de aplicações, é usado *Uniform resource locator* (URL) *Schemes*

No caso específico de iOS, o seu funcionamento é semelhante ao de um GET *request* a um servidor, sendo esta informação enviada no *header*.

Sendo a informação necessária passar entre as aplicações o ID de cada produto e a quantidade que o utilizador pretende comprar, foi definida a seguinte

estrutura:

```
indoor://co.whitesmith.indoorApp?
    product=102:2&product=(ID):(Quantidade)
```

Olhando para o *request* podemos identificar 3 partes diferentes:

indoor É o valor do CFBundleURLSchemes, que serve para identificar qual o esquema que o *request* apresenta.

co.whitesmith.indoorApp É o CFBundleURLName, que serve para identificar a aplicação para onde queremos enviar o pedido. De modo a ser único, é usualmente usado o *Domain Name System* (DNS) invertido.

product=.... São os produtos que queremos passar para a outra aplicação. Neste caso, são enviados os IDs dos produtos que estão na lista de compras seguido de ":"e a quantidade que o utilizador pretende comprar separados por &.

6.6.2 Comunicação com o servidor

Nesta secção está definido qual o modelo de dados utilizados para requer cada tipo de informação ao servidor.

A aplicação comunica com o servidor em duas situações:

1. Requerer mapa da superfície onde o utilizador se encontra naquele momento
2. Requerer informação relativa a um produto

Dados da superfície

Para requerer os dados da superfície é realizado um *http request* da seguinte forma:

```
DOMAIN/stores/(store id)
```

A "store id" é valor do *major value* que cada pacote do protocolo iBeacon envia.

A aplicação espera uma resposta json com os seguintes parâmetros:

Parâmetro	Tipo
id	Int
name	String
longitude	Float
latitude	Float
updateAt	Date
maps	[Int] *

*Sendo cada valor o id do mapa

Tabela 6.2: Resposta json para cada superfície comercial

Após a receção da resposta anterior, é necessário pedir informação de cada mapa. Assim, para cada valor na lista "maps" é efetuado o seguinte pedido:

```
DOMAIN/maps/(map id).json
```

Este pedido deve devolver uma resposta json com os parâmetros:

Parâmetro	Tipo
id	Int
sizeX	Float
sizeY	Float
magneticOrientation	Float
storeID	Int
updateAt	Date

Tabela 6.3: Resposta json para cada mapa

Podemos considerar que cada mapa contém 3 partes diferentes:

- Mapa da superfície, constituído por prateleiras
- Grafo responsável por indicar quais os caminhos válidos (sendo o seu uso explicado nas secções anteriores)
- Posições dos beacons

Para requerer o mapa da superfície, é efetuado o seguinte pedido:

`DOMAIN/maps/(map id)/shelves.json`

É esperada uma reposta composta no formato:

Parâmetro	Tipo
mapID	Int
shelves	[Shelf]

Shelf	
Parâmetro	Tipo
id	Int
posX	Float
posY	Float
height	Float
width	Int
category	Int

Tabela 6.4: Resposta json para prateleiras

Uma vez que os pedidos ao servidor são efetuados de forma assíncrona, é necessário que o ID do mapa (mapID) seja enviado na resposta.

Para receber informação das posições dos beacons é realizado um pedido ao servidor no seguinte formato:

`DOMAIN/maps/(map id)/beacons`

Sendo a formato da resposta:

Parâmetro	Tipo
beacons	[Beacon]

Beacon	
Parâmetro	Tipo
majorValue	Int
minorValue	Int
posX	Float
posY	Float

Tabela 6.5: Resposta json para beacons

Por fim, é necessária informação relativa ao grafo. Sendo o seu pedido:

`DOMAIN/maps/(map id)/graph`

A reposta deve vir sobre no formato:

Parâmetro	Tipo
mapID	Int
nodes	[Node]
edges	[Edge]

Node	
Parâmetro	Tipo
id	Int
posX	Float
posY	Float

Edge	
Parâmetro	Tipo
node1	Int
node2	Int
shelves	[Int]*

*Sendo cada valor o id da prateleira

Tabela 6.6: Resposta json para o grafo

Dados dos produtos

Sempre que a informação relativa à lista de compras é recebida, é necessário saber a posição de cada produto na superfície comercial.

Para isso é efetuado um pedido na forma:

```
DOMAIN/stores/(store id)?product_id=(product1 ID)&product_id=(product1
```

Parâmetro	Tipo
products	[Product]

Product	
Parâmetro	Tipo
id	Int
relativePosition	Float
shelfId	Int
mapId	Int
name	Int

Tabela 6.7: Resposta json para produtos

6.6.3 Gestão de dados

Uma vez que as superfícies comerciais mantêm o seu espaço durante elevados períodos de tempo, foi implementado um sistema para guardar de forma persistente estes dados, de forma a minimizar a transferência de dados entre o servidor e a aplicação. Para isso, foi implementada uma base de dados relacional, mais especificamente SQLite.

Os dados guardados e as suas ligações podem ser observadas na figura 6.15.

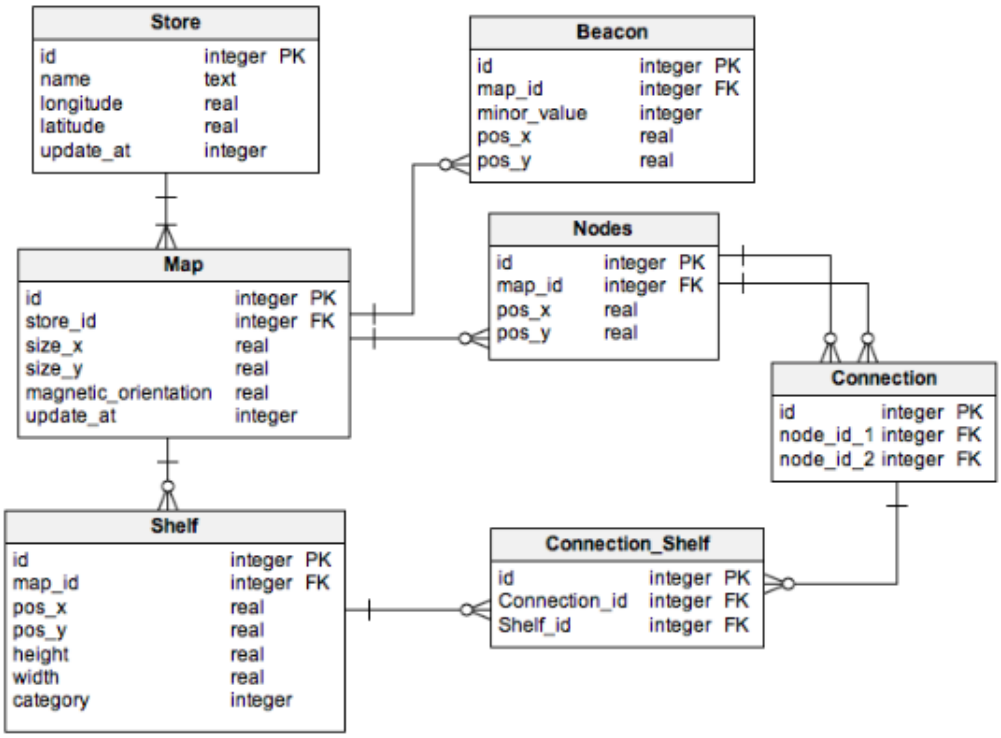


Figura 6.15: Diagrama entidade-relação da base de dados da aplicação do guia dos clientes

O campo "update_at" presente na tabela "Store" e "Map", é usado para garantir que o utilizador tem a versão mais atual dos mapas da superfície. Para isso, é comparado este parâmetro com o campo "updateAt" proveniente da resposta ao servidor. Caso o resultado desta comparação indique que a base de dados tem a versão atualizada, são usados os dados presentes na base de dados. Caso contrário, é realizada a atualização destes dados.

6.7 Testes ao sistema

Para testar o sistema foi criado um servidor em Ruby, apenas com o propósito de responder aos pedidos feitos pela aplicação no formato definido na secção 6.6.2. Foi também criada uma aplicação simples de forma a testar a comunicação com a aplicação de lista de compras, tal como definido na secção 6.6.1. Neste secção estão documentados os testes efetuados.

6.7.1 Inicialização

Teste 1: A aplicação identifica corretamente a superfície comercial onde o utilizador se encontra.

Resultado: ✓

Teste 2: A aplicação é capaz de fazer o pedido ao servidor e interpretar a resposta que contém a informação dessa superfície comercial

Resultado: ✓

Teste 3: A aplicação guarda informação da loja de forma persistente na base de dados

Resultado: ✓

Teste 4: Caso exista informação relativa á superfície comercial onde o utilizador se encontra na base de dados, usa essa informação, caso esteja atualizada.

Resultado: ✓

Teste 5: Atualiza informação relativa á superfície comercial, caso esteja desatualizada (grafo, prateleiras, posição dos beacons e informação relativa à loja)

Resultado: ✓

Teste 6: A aplicação gera um aviso caso o utilizador tenha *bluetooth* desligado

Resultado: ✓

Teste 7: A aplicação gera um aviso caso o utilizador não tenha acesso à Internet

Resultado: ✓

Teste 8: Se o telemóvel reconhecer um beacon, é possível iniciar a aplicação a partir *lock screen*.

Resultado: ✓

6.7.2 Lista de compras

Teste 1: Aplicação de guia dos clientes é capaz de requerer informação relativa á lista de compras a outra aplicação

Resultado: ✓

Teste 2: Uma aplicação é capaz de enviar a lista de compras para aplicação de guia dos cliente

Resultado: ✓

Teste 3: Caso a aplicação de guia não esteja a ser executada, outra aplicação é capaz de enviar uma lista de compras e iniciar a aplicação.

Resultado: ✓

Teste 4: A aplicação de guia dos clientes, é capaz de interpretar informação relativa á lista de compras do utilizador.

Resultado: ✓

Teste 5: A aplicação de guia dos clientes, é capaz de requerer as informações

necessárias dos produtos ao servidor.

Resultado: ✓

Teste 6: A aplicação de guia dos clientes é capaz de mostrar os produtos da lista de compras no mapa.

Resultado: ✓

6.7.3 Mapeamento

Teste 1: A aplicação é capaz de gerar e mostrar um mapa da superfície.

Resultado: ✓

Teste 2: A aplicação é capaz de mostrar a posição do utilizador no mapa.

Resultado: ✓

Teste 3: A aplicação é capaz de mostrar a direção do utilizador.

Resultado: ✓

Teste 4: O utilizador pode clicar numa prateleira e ver a sua categoria de produtos.

Resultado: ✓

Teste 5: O utilizador pode clicar num ícone do produto para ver alguns detalhes.

Resultado: ✓

Teste 6: O utilizador é capaz de eliminar um produto da sua lista de compras

Resultado: ✓

6.7.4 Navegação

Teste 1: Apenas se uma lista de compras tiver sido importada, é dada a possibilidade iniciar um modo de navegação.

Resultado: ✓

Teste 2: O utilizador consegue ver o percurso que vai te que percorrer até ao próximo produto.

Resultado: ✓

Teste 3: A aplicação dá indicações de como chegar até a um produto.

Resultado: ✓

Teste 4: Quando um produto está próximo do utilizador é mostrado informação do lado em que o produto se encontra

Resultado: ✓

Teste 5: Quando o utilizador indica que recolheu um produto, é gerado e mostrado o percurso até ao próximo produto.

Resultado: ✓

Teste 6: Quando um produto está próximo do utilizador é mostrado informação do lado em que o produto se encontra.

Resultado: ✓

Teste 7: Se o utilizador sair do percurso indicado, é gerado um novo percurso e atualizado o produto a ir buscar, caso se altere.

Resultado: ✓

Teste 8: Quando todos os produtos são recolhidos, é automaticamente fechado o modo de navegação.

Resultado: ✓

Teste 9: Se um produto for eliminado, o percurso é atualizado.

Resultado: ✓

Capítulo 7

Conclusão e Trabalho Futuro

Durante a realização deste estágio foi tirado proveito do BLE com o objetivo de auxiliar clientes das superfícies comerciais de tal forma que lhes seja permitido recolherem os produtos presentes na sua lista de compras no menor espaço possível. Deste modo, e para esse efeito, foi programada uma aplicação móvel capaz de realizar esta tarefa na qual foi utilizado um algoritmo de lateração que permite efetuar uma estimativa da posição do utilizador. Sendo que o A* e o *nearest neighbor algorithm* foram utilizados para a criação do percurso que o utilizador deve percorrer.

7.1 Conclusão

Este estágio demonstrou que é possível tirar proveito do BLE para estimar a localização do utilizador dentro de edifícios tirando partido da lateração. Com este conjunto foi possível criar um sistema de fácil instalação na superfície e com uma exatidão média de, aproximadamente, um metro na estimativa

do posicionamento dos utilizadores. Esta margem de erro mostrou ser suficientemente baixa de forma a permitir guiar os utilizadores ao longo de uma superfície.

Apesar de ser uma tecnologia recente, considera-se que o seu custo é reduzido, dado ser possível adquirir *beacons* por menos de 5 euros, sendo expectável que este preço desça ao longo dos próximos anos. Aliando o baixo custo deste método de localização no interior de edifícios com a necessidade das superfícies comercias em prestarem auxílio aos utilizadores permitindo que os mesmos se desloquem ao encontro do que pretendem, é expectável que ao longo dos anos existam cada vez mais superfícies que demonstrem interesse e implementem sistemas semelhantes ao deste estágio.

7.2 Trabalho Futuro

Para o trabalho futuro têm de ser programadas as aplicações descritas, mas não implementadas, neste estágio, bem como criar a versão para o Android deste sistema.

No caso específico da aplicação implementada, é necessário encontrar um parceiro comercial que permita a validação do sistema numa superfície comercial. Para melhorar a exatidão do algoritmo de localização é interessante juntar outras tecnologias, como o uso de sensores como, por exemplo, o magnetómetro. Outra consideração interessante, é validar as posições estimadas com a posição anterior e com o mapa da superfície comercial. Deste modo seria possível invalidar posições, isto é, tendo conhecimento da posição anterior do utilizador seria possível prever que a distância á qual o utilizador se encontrará, sendo que esta não poderá ultrapassar determinado limite, tendo em conta a barreira

do tempo e do espaço percorrido.

Por outro lado, aquando da geração do percurso mais curto seria interessante estudar uma forma de ter uma heurística mais perto do realidade. Caso tal situação verifique, tanto o algoritmo A* como *nearest neighbor algorithm* iriam beneficiar na sua performance e no seu desempenho, respetivamente.

Referências

[1] Nor Aida Mahiddin, Noaizan Safie, Elissa Nadia, Suhailan Safei, and Engku Fadzli. Indoor Position Detection Using WiFi and Trilateration Technique. *International Conference on Informatics and Applications*, pages 362–366, 2012.

[2] Sudarshan S Chawathe. Beacon Placement for Indoor Localization using Bluetooth. *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems*, pages 980–985, 2008.

[3] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 37(6):1067–1080, 2007.

[4] G Trein. Simple approach for indoor mapping using low-cost accelerometer and gyroscope sensors.

[5] How does an estimote beacon work? `https://community.estimote.com/hc/en-us/articles/204086423-How-does-an-Estimote-Beacon-work-`. Acedido em: 2014-11-15.

[6] a Boukerche, Habf a B Oliveira, E F Nakamura, and a a F Loureiro. Localization systems for wireless sensor networks. *IEEE wireless ...*, 14(December):6–12, 2007.

[7] ITU. Overview of the Internet of things, 2012.

[8] What makes a smartphone smart? `http://cellphones.about.com/od/smartphonebasics/a/what_is_smart.htm`. Acedido em: 2014-12-08.

[9] Oxford dictionaries - smartphone. `http://www.oxforddictionaries.com/definition/english/smartphone`. Acedido em: 2014-12-08.

[10] Mobile now exceeds pc: The biggest shift since the internet began. `http://searchenginewatch.com/sew/opinion/2353616/mobile-now-exceeds-pc-the-biggest-shift-since-the-internet-began`. Acedido em: 2015-08-28.

[11] Device share of page views across countries in europe. `http://www.smartinsights.com/wp-content/uploads/2013/02/European-Digital-Media-Use-Europe.png`. Acedido em: 2015-08-28.

[12] Aplicações na *Apple Store* mensalmente. `http://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store`. Acedido em: 2015-01-06.

[13] Mobile technology fact sheet; highlights of the pew internet project's research related to mobile technology. `http://www.pewinternet.org/fact-sheets/mobile-technology-fact-sheet/`. Acedido em: 2014-12-08.

[14] Eu5 smartphone penetration reaches 55 percent in october 2012. https://www.comscore.com/Insights/Press-Releases/2012/12/EU5-Smartphone-Penetration-Reaches-55-Percent-in-October-2012. Acedido em: 2014-12-08.

[15] Mobile in-store research, how in-store shoppers are using mobile devices. https://ssl.gstatic.com/think/docs/mobile-in-store_research-studies.pdf. Acedido em: 2015-01-09.

[16] Krista Garcia. Supermarkets and mobile, satisfying grocery shoppers' app-etites. http://www.sap.com/bin/sapcom/en_us/downloadasset.2014-04-apr-24-16.supermarkets-and-mobile--satisfying-grocery-shoppers-app-etites-pdf.bypassReg.html. Acedido em: 2014-12-23.

[17] Latitude. Next-gen retail, mobile and beyondº. http://files.latd.com/Latitude-Next-Gen-Retail-Study.pdf. Acedido em: 2015-01-03.

[18] Cartão de cliente usado em 90% das compras no continente. http://www.jornaldenegocios.pt/empresas/detalhe/cartatildeo_de_cliente_usado_em_90_das_compras_no_continente.html. Acedido em: 2014-11-01.

[19] Tesco: Key facts. http://www.tescoplc.com/index.asp?pageid=71. Acedido em: 2015-08-23.

[20] Hank mullany entrevista para a fortune. http://fortune.com/2014/09/10/toys-r-us-holiday/. Acedido em: 2014-12-12.

[21] Yapeng Wang, Xu Yang, Yutian Zhao, Yue Liu, and Laurie Cuthbert. Bluetooth positioning using RSSI and triangulation methods. *2013 IEEE 10th*

Consumer Communications and Networking Conference, CCNC 2013, pages 837–842, 2013.

[22] O S Oguejiofor, a N Aniedu, H C Ejiofor, and a U Okolibe. Trilateration Based localization Algorithm for Wireless Sensor Network. (10):21–27, 2013.

[23] Sudarshan S. Chawathe. Low-latency indoor localization using bluetooth beacons. *2009 12th International IEEE Conference on Intelligent Transportation Systems*, pages 1–7, 2009.

[24] K. Kaemarungsi. Efficient design of indoor positioning systems based on location fingerprinting. *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, 1:181–186, 2005.

[25] Tareq Alhmiedat and Ghassan Samara. An Indoor Fingerprinting Localization Approach for ZigBee Wireless Sensor Networks. 105(2):190–202, 2013.

[26] Subrata Goswami. *Indoor Location Technologies*. 2012.

[27] Jaewoo Chung, Matt Donahoe, Chris Schmandt, Ig-Jae Kim, Pedram Razavai, and Micaela Wiseman. Indoor location sensing using geomagnetism. *Proceedings of the 9th international conference on Mobile systems, applications, and services - MobiSys '11*, pages 141–154, 2011.

[28] Danilo Navarro and Gines Benet. Magnetic map building for mobile robot localization purpose. *ETFA 2009 - 2009 IEEE Conference on Emerging Technologies and Factory Automation*, pages 4–7, 2009.

[29] Near field communication. `https://developer.android.com/guide/topics/connectivity/nfc/index.html`. Acedido em: 2015-03-20.

[30] Busra Ozdenizci, Kerem Ok, Vedat Coskun, and Mehmet N. Aydin. Development of an indoor navigation system using NFC technology. *Proceedings - 4th International Conference on Information and Computing, ICIC 2011*, pages 11–14, 2011.

[31] Annika Paus. Near Field Communication in Cell Phones. *Security*, 2007.

[32] Veljo Otsason, Alex Varshavsky, Anthony Lamarca, and Eyal De Lara. Accurate GSM Indoor Localization. *Pervasive and Mobile Computing*, 3(6):698–720, 2007.

[33] Quentin Ladetto and Bertrand Merminod. Digital magnetic compass and gyroscope integration for pedestrian navigation. *9th Saint Petersburg International Conference on Integrated Navigation Systems*, pages 111–120, 2002.

[34] Masaki Yoshino, Shinichiro Haruyama, and Masao Nakagawa. High-accuracy positioning system using visible LED lights and image sensor. *2008 IEEE Radio and Wireless Symposium, RWS*, pages 439–442, 2008.

[35] Frédéric Evennou and François Marx. Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning. *Eurasip Journal on Applied Signal Processing*, 2006:1–11, 2006.

[36] Chin Heng Lim, Yahong Wan, Boon Poh Ng, and Chong Meng Samson See. A real-time indoor WiFi localization system utilizing smart antennas. *IEEE Transactions on Consumer Electronics*, 53(2):618–622, 2007.

[37] F. Lassabe, P. Canalda, P. Chatonnay, F. Spies, and D. Charlet. Refining WiFi indoor positioning renders pertinent deploying location-based multimedia guide. *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, 2:126–130, 2006.

[38] Karen Duarte, S Jorge, and Pedro Furtado. Information and Assisted Navigation System for Blind People. pages 2–4, 2014.

[39] Mario Muñoz Organero, Pedro J. Muñoz Merino, and Carlos Delgado Kloos. Using bluetooth to implement a pervasive indoor positioning system with minimal requirements at the application level. *Mobile Information Systems*, 8(1):73–82, 2012.

[40] Silke Feldmann, Kyandoghere Kyamakya, Ana Zapater, and Zighuo Lue. An indoor Bluetooth-based positioning system : concept , Implementation and experimental evaluation. *International Conference on Wireless Networks*, pages 109–113, 2003.

[41] A look at the basics of bluetooth technology. `http://www.bluetooth.com/Pages/Basics.aspx`. Acedido em: 2015-03-21.

[42] The low energy technology behind bluetooth smart. `http://www.bluetooth.com/Pages/low-energy-tech-info.aspx`. Acedido em: 2015-03-21.

[43] F Agostaro, F Collura, a Genco, S Sorce, and Dinfo Dipartimento. Problems and solutions in setting up a low-cost Bluetooth positioning system. *Informatica*, 3(4):1102–1106, 2004.

[44] Getting started with ibeacon. `https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf`. Acedido em: 2014-11-20.

[45] Reverse engineering the estimote. http://makezine.com/2014/01/03/reverse-engineering-the-estimote/. Acedido em: 2015-03-21.

[46] How do ibeacons work? http://www.warski.org/blog/2014/01/how-ibeacons-work/. Acedido em: 2015-03-21.

[47] Aisle411 presentation for location intelligence conference 2013 v4. http://pt.slideshare.net/KrisKolodziej/aisle411-presentation-for-location-intelligence-conference-2013-v4-22434225. Acedido em: 2015-03-21.

[48] Native, html5, or hybrid: Understanding your mobile application development options. https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options. Acedido em: 2015-03-23.

[49] The state of native vs. web vs. hybrid. https://dzone.com/articles/state-native-vs-web-vs-hybrid. Acedido em: 2015-03-23.

[50] Number of apps available in leading app stores as of july 2015. http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/. Acedido em: 2014-12-08.

[51] Active stakeholder participation: An agile best practice. http://www.agilemodeling.com/essays/activeStakeholderParticipation.htm. Acedido em: 2015-04-06.

[52] About swift. `https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/`. Acedido em: 2015-04-08.

[53] Xiaojie Zhao, Zhuoling Xiao, and Andrew Markham. Does BTLE measure up against WiFi? A comparison of indoor location performance. *Proceedings of European Wireless 2014; 20th European Wireless Conference*, pages 263–268, 2014.

[54] Jie Yang and Yingying Chen. Indoor localization using improved rss-based lateration methods. *GLOBECOM - IEEE Global Telecommunications Conference*, 4, 2009.

[55] B. Moses Sathyaraj, L. C. Jain, a. Finn, and S. Drake. Multiple UAVs path planning algorithms: A comparative study. *Fuzzy Optimization and Decision Making*, 7(3):257–267, 2008.

Anexo A

Estado da Arte

Neste documento pretende-se fazer uma análise do estado atual das aplicações móveis de grandes superfícies.

Os campos analisados são:

- Lista de compras, que verifica se a aplicação tem capacidade de guardar uma lista de compras;
- Cartão de fidelização, verifica se é capaz de substituir e complementar os cartões de fidelização;
- Compras online, verifica se é possível a realização de compras online;
- Promoções, analisa a facilidade de acesso a produtos em promoção;
- Compras habituais, analisa se a aplicação é capaz de guardar as compras habituais;

- Detalhes de produtos, analisa a capacidade de pesquisa de produtos e seus detalhes (como o preço e índices nutricionais);
- Loja mais próxima, analisa a capacidade da aplicação indicar ao utilizador qual a loja mais próxima;
- Localizador de produtos, analisa a capacidade da aplicação em indicar ao utilizador onde se encontra determinado produto.

Ao todo foram analisadas 13 aplicações disponíveis no mercado. Para saber valor estimado do número de downloads foi usado o serviço xyo.net. Para o nível de satisfação da aplicação, foram usados dados de satifação acessíveis na Apple Sotre e Google Play. [1]

Continente

O Continente é um dos maiores hipermercados portuguêses, que pertence ao grupo Sonae. O principal foco desta aplicação é o auxílio dos seus mais de três milhões de utilizadores que têm um cartão da marca.[18]

Número de *downloads*:

- 192 mil no sistema operativo Android;
- 55 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5 (versão 2.1.11)

Satisfação na Google Play: 3.9 em 5

[1] Dados de 2014-11-04

Na tabela A.1 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	✓	✗	Folhetos	✗	Preço	✓	✗

Tabela A.1: Análise da aplicação Continente

Observações:

- A aplicação apenas consegue mostrar os folhetos das promoções em vigor, não listando os produtos de forma simples;
- Não é capaz de mostrar mais detalhes do que o preço e quantidades.

Auchan

O grupo fracês Auchan, é uma das maiores empresas na área a nível mundial. A sua aplicação não está disponível para o público português, apesar de ter presença em Portugal com o nome de Jumbo. A versão testada foi a francesa.

Número de *downloads*:

- 509 mil no sistema operativo Android;
- 201 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5 (versão 5.0.5)

Satisfação na Google Play: 3.3 em 5

Na tabela A.2 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	✗	✓*	✓	✗	✓	✓	✗

Tabela A.2: Análise da aplicação Auchan

Observações:

- É apenas possível fazer compras online através de outra aplicação do grupo.

Intermarché

O Intermarché é uma cadeia de supermercados francesa que opera também no mercado português.

Número de *downloads*: Não disponível

Satisfação na Apple Store: Não disponível

Satisfação na Google Play: 3.5 em 5

Na tabela A.3 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	✗	✗	✓	✗	✗	✓	✗

Tabela A.3: Análise da aplicação Intermarché

Observações:

- A lista de compras não é capaz de reconhecer produtos. Limita-se a adicionar o que o utilizador escreve.

Tesco Groceries

Tesco Groceries é uma aplicação mobile que pertence à Tesco Stores Ltd. sediada no Reino Unido. Foi desenvolvida para ser usada nas mais de 3300 lojas presentes no Reino Unido. Esta aplicação está mais otimizada para compras online.

Número de *downloads*:

- 2 milhões no sistema operativo Android;
- 658 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5 (versão 7.5)

Satisfação na Google Play: 3.5 em 5

Na tabela A.4 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	✓	✓	✓	✓	✓	Código Postal	✗

Tabela A.4: Análise da aplicação Tesco Groceries

Observações:

- Pesquisa de lojas apenas por código postal.

Walmart

A Wal-Mart Stores, Inc opera nos Estados Unidos da América como Walmart.
A aplicação dá um foco especial para as promoções existentes nos seus estabelecimentos.

Número de *downloads*:

- 14 milhões no sistema operativo Android;
- 5,2 milhões no sistema operativo iOS.

Satisfação na Apple Store: 4,5 em 5 (versão 5.1)
Satisfação na Google Play: 4,3 em 5

Na tabela A.5 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	N.A.	✓	✓	✗	✓	✓	Prateleira

Tabela A.5: Análise da aplicação Walmart

Observações:

- Esta organização não possui cartões de fidelização.

ASDA

A ASDA é um dos maiores supermercados no Reino Unido. Atualmente é detida pela companhia americana Wal-Mart Stores, Inc. A aplicação ASDA está orientada para compras online.

Número de *downloads*:

- 2 milhões no sistema operativo Android;
- 964 mil no sistema operativo iOS.

Satisfação na Apple Store: 4,5 em 5 (versão 3.1)

Satisfação na Google Play: 4,6 em 5

Na tabela A.6 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	N.A.	✓	✓	✗	✓	✓	✗

Tabela A.6: Análise da aplicação ASDA

Observações:

- Esta organização não possui cartões de fidelização.

Sainsbury's

A Sainsbury's é outro grande supermercado no Reino Unido.

Esta aplicação está nitidamente virada para compras online. Uma vez que esta é a principal função e reencaminha o utilizador para um website, isto originou algumas críticas por parte dos utilizadores.

Número de *downloads*:

- 283 mil no sistema operativo Android;
- 224 mil no sistema operativo iOS.

Satisfação na Apple Store: 1,5 em 5 (versão 3.1.2)

Satisfação na Google Play: 3,1 em 5

Na tabela A.7 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	✓	✓	✓	✗	✓	✓	✗

Tabela A.7: Análise da aplicação Sainsbury's

Lidl

A Lidl é das maiores empresas do seu sector na distribuição alimentar, com sede na Alemanha e presente em mais de 20 países europeus.
Esta aplicação está orientada para a informação de preços e promoções aos seus clientes.

Número de *downloads*:

- 2,7 milhões no sistema operativo Android;
- 989 mil no sistema operativo iOS.

Satisfação na Apple Store: 3,5 em 5 (versão 4.6)
Satisfação na Google Play: 4,4 em 5

Na tabela A.8 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	N.A.	✗	✓	✗	Preço	Código Postal	✗

Tabela A.8: Análise da aplicação Lidl

Observações:

- Esta organização não possui cartões de fidelização.
- Apenas permite encontrar as lojas mais próximas a partir do código postal.

Kroger Co.

A Kroger está presente nos Estados Unidos da América.

A aplicação está orientada para os cartões de fidelização da marca, permitindo criar um a partir da aplicação. Interessante analisar que apesar de a aplicação estar completamente funcional e com um aspeto atraente, os utilizadores queixam-se de falta de funcionalidades.

Número de *downloads*:

- 2,1 milhões no sistema operativo Android;
- 193 mil no sistema operativo iOS.

Satisfação na Apple Store: 2 em 5 (versão 4.6)

Satisfação na Google Play: 4,2 em 5

Na tabela A.9 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	✓	✓	✓	✗	✓	✓	✗

Tabela A.9: Análise da aplicação Kroger

Costco

A Costco é das maiores empresas de revenda a nível mundial, vendendo um vasto tipo de produtos.
O grupo tem uma aplicação com o mesmo nome, fortemente virada para promoções e venda online.

Número de *downloads*:

- 2,1 milhões no sistema operativo Android;
- 325 mil no sistema operativo iOS.

Satisfação na Apple Store: 2 em 5 (versão 4.6)
Satisfação na Google Play: 3,6 em 5

Na tabela A.10 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	✓	✓	✓	✗	✗	✓	✗

Tabela A.10: Análise da aplicação Costco

Observações:

- A aplicação reencaminha-nos constantemente para sites externos, como por exemplo, para realizar compras online.
- Lista de compras apenas guarda o que escrevemos não identificando produtos.

fnac.pt

A Fnac é uma empresa dedicada a vender equipamentos eletrónicos e produtos culturais, presente em vários países a nível mundial. A aplicação está fortemente virada para o comércio online.

Número de *downloads*:

- menos de mil no sistema operativo Android;
- 12 mil no sistema operativo iOS.

Satisfação na Apple Store: 4 em 5

Satisfação na Google Play: 4,2 em 5

Na tabela A.11 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✗	✗	✓	✓	✗	✓	✓	✗

Tabela A.11: Análise da aplicação fnac.pt

Observações:

- Alguns utilizadores queixam-se da impossibilidade de adicionar o cartão de fidelização.

Ikea

A Ikea é uma empresa de origem sueca que se dedica à venda de móveis de baixo custo. Está atualmente presente em mais de 40 países mundialmente.
A aplicação dedica-se ao auxílio dos clientes nos seus estabelecimentos.

Número de *downloads*:

- 4,8 milhões no sistema operativo Android;
- 338 mil no sistema operativo iOS.

Satisfação na Apple Store: 3 em 5
Satisfação na Google Play: 3,9 em 5

Na tabela A.12 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
✓	✗	✗	Folhetos	✗	✓	✓	Prateleira

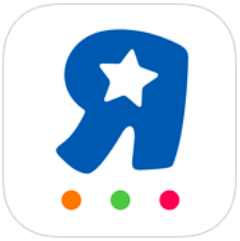
Tabela A.12: Análise da aplicação Ikea

Observações:

- A aplicação é capaz de nos indicar qual o corredor e secção onde o produto se encontra e se tem ou não stock na loja.

Toys ”R” Us Shoping

A Toys ”R” Us é uma multinacional americana com mais de mil e quinhentas lojas espalhadas pelo mundo, que se dedica a vender brinquedos.
A aplicação tem como foco principal a venda online de produtos.



Número de *downloads*:

- 503 mil no sistema operativo Android;
- 62 mil no sistema operativo iOS.

Satisfação na Apple Store: 3,7 em 5
Satisfação na Google Play: 3,5 em 5

Na tabela A.13 é feita uma análise a esta aplicação.

Lista de compras	Cartão de fidelização	Compras online	Promoções	Compras habituais	Detalhes de produtos	Loja mais próxima	Localizador de produtos
Wish List	✗	✓	✓	✗	Preço	✓	✗

Tabela A.13: Análise da aplicação Toys ”R” Us

Anexo B

Diagramas de Gant Detalhados

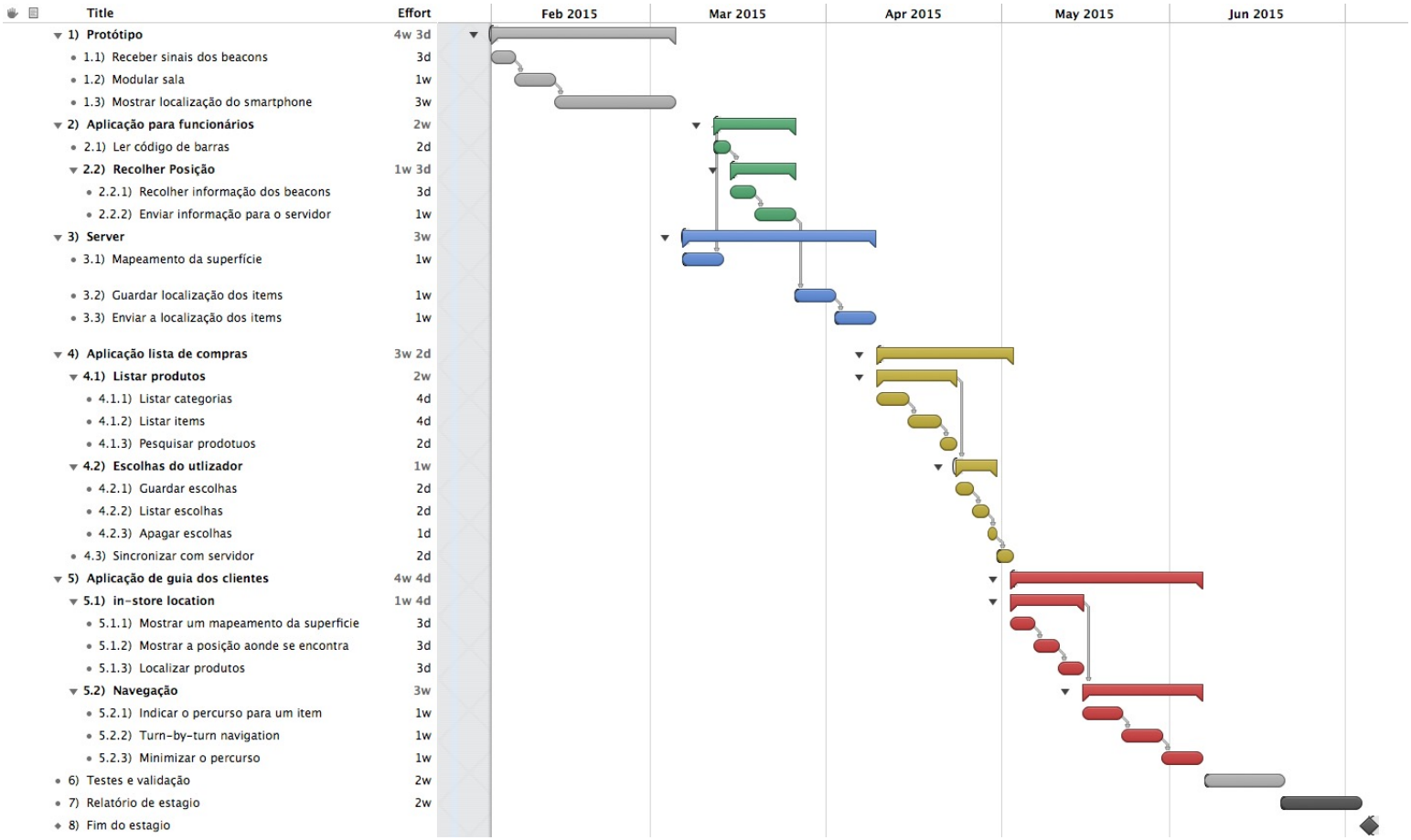


Figura B.1: Diagrama de Gant planeado para o segundo semestre (detalhado)

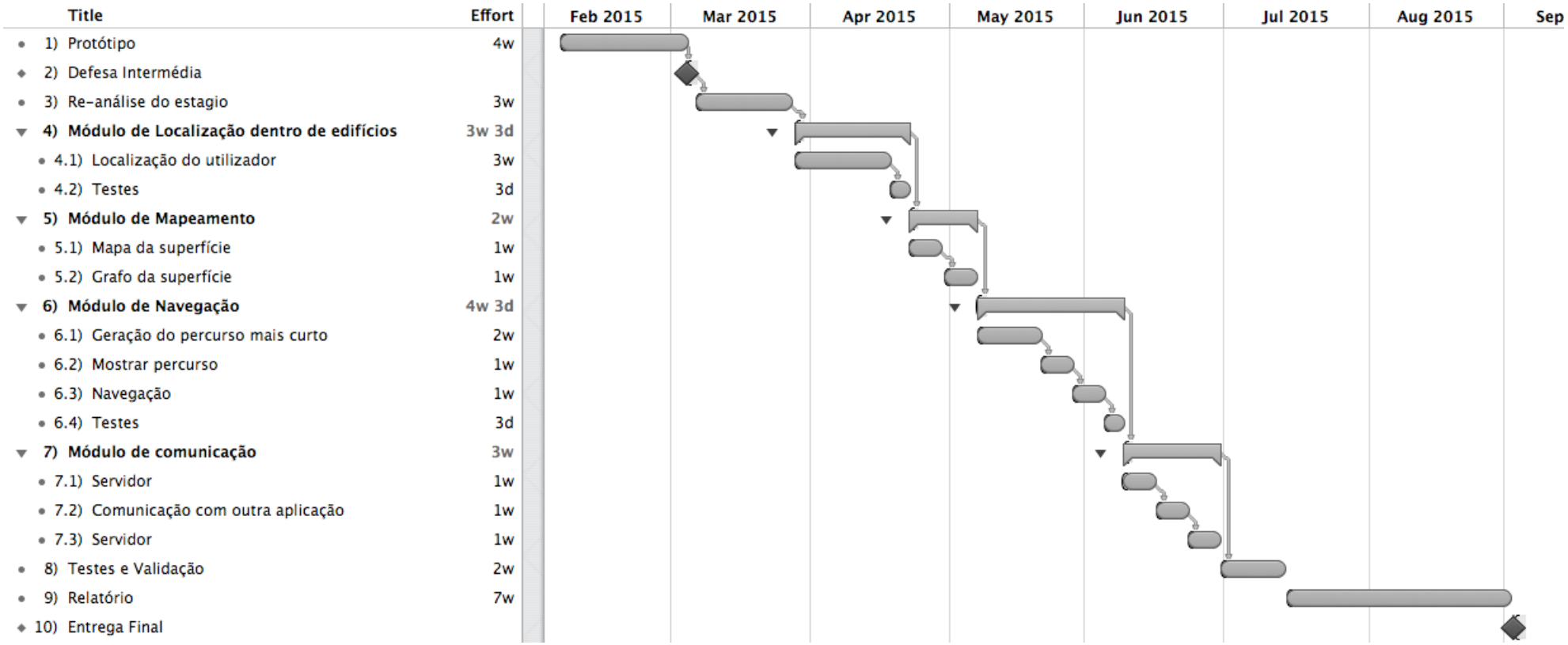


Figura B.2: Diagrama de Gant real do segundo semestre (detalhado)