UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

Departamento de Engenharia Geográfica, Geofísica e Energia

Desenvolvimento de uma aplicação móvel para traçado de percurso em ambiente *indoor*

André Filipe Fernandes do Adro

Mestrado em Sistemas de Informação Geográfica – Tecnologias e Aplicações

Dissertação orientada por:

Prof. Doutor João Catalão Fernandes

Prof. Doutora Ana Paula Pereira Afonso

2019

Agradecimentos

Acabou…

Com esta entrega encerro um dos capítulos mais importantes da minha vida. Contudo, existiram diversos subcapítulos, uns bons, outros maus, mas todos eles fazem parte de mim, por isso estarão comigo para o resto da minha vida.

Correndo o risco de parecer ingrato e poder deixar de lado alguma pessoa que fez parte desta fase deixo um sincero obrigado a todas as pessoas que de uma maneira ou de outra estiveram comigo ao longo destes longos anos.

Contudo, existem um conjunto de pessoas dignas de serem mencionadas que estiveram mais próximas de mim, tendo algumas até percorrido um caminho semelhante ao meu.

Em primeiro lugar gostaria de agradecer à minha família por me terem apoiado e me terem "deixado" em paz de modo a eu poder conseguir concluir e encerrar este capítulo. Ao meu Pai por me ter mostrado como se lida com situações difíceis na vida e como enfrentá-las de cabeça erguida. À minha Mãe, por me ter dado um verdadeiro susto, tendo sobrevivendo ao mesmo, mas que me fez ganhar alguma consciência de que em primeiro lugar está a saúde, só depois veem o resto. Ao meu irmão que me fez ver que a vida continua especialmente pelo facto de me ter tornado tio do miúdo mais reguila que eu conheço. À minha cunhada por ter paciência para aturar o meu irmão e me ter dado o sobrinho mais fofo que existe.

Em segundo lugar, aos meus amigos que mais consigo identificar neste capítulo, Ferreira, Gradiz e Ion. Obrigado por me terem feito sentir tão limitado em comparação em vocês e por me terem trazido para o vosso nível académico tendo-me ajudado a crescer e a evoluir de uma maneira que eu nunca pensei ser possível. Ainda nesta nota, um obrigado a todos os amigos que ganhei, ou que já faziam parte da minha vida e por terem tido paciência de todas as negas que dei. Ao Serrenho, Gonçalves, Rodolfo, Algarvio, Montez, Nádia, Durão, Janeco, Cíntia, Miguel, Inês, Duarte, Catarina, Carol, Joana.

Em terceiro, um especial obrigado aos Professores João Catalão e Ana Paula Afonso, por apesar do meu desaparecimento, ainda terem tido a disponibilidade e entrega de me ajudar a acabar este capítulo garantido que era entregue dentro de prazos praticamente impossíveis de atingir.

Por último, um especial obrigado a todos os que já não fazem parte da minha vida, obrigado por me terem permitido crescer e evoluir de uma maneira totalmente desconhecida para mim. Onde quer que estejam, obrigado, e desculpem-me, especialmente a ti L…

Resumo

Com as inovações tecnológicas da última metade de século, a tecnologia tem possibilitado ao ser humano alcançar sítios que antes eram inimagináveis. A área de posicionamento não é exceção à regra. Com a constante navegação no *outdoor* através de tecnologias como o GPS, apenas o *indoor* está inexplorado. Devido a limitações da tecnologia GPS tem-se observado um crescente interesse nesta temática com o uso de tecnologias alternativas para o posicionamento e navegação *indoor*.

Neste documento será apresentado uma solução de posicionamento *indoor* somente com a utilização do Wi-Fi, sendo possível solicitar percursos que permitam a um utilizador chegar ao seu ponto de destino. Para tal será utilizado uma técnica de votação através de *fingerprinting* de uma rede previamente calibrada para encontrar a posição do utilizador.

A área de interesse é o piso 1, do edifício C8 da Faculdade de Ciências da Universidade de Lisboa, e será realizado um trabalho que permita a um utilizador solicitar caminhos para pontos de interesse.

Para tal, este trabalho tem como objetivo criar um *Location Based Service*, através da disponibilização de interfaces REST, que permita a sua reutilização noutros espaços de interesse sendo para tal apenas necessário o aprovisionamento correto dos dados.

Os resultados consistiram no desenvolvimento de uma aplicação em *Android* que comunica com uma API de serviços disponibilizados num servidor aplicacional, permitindo assim que o utilizador desta aplicação possa solicitar caminhos para chegar a um ponto de interesse, sendo que, à medida que se desloca neste caminho a sua posição é atualizada com base no Wi-Fi.

Palavras-chave: LBS, Wi-Fi, posicionamento *indoor, Fingerpriting*, REST, A*ndroid*

Abstract

With the recent technological advances of the last half of a century, technology has been the enabler for the human being to reach unthinkable territories. The field of positioning is no exception. With the constant outdoor navigation through the use of technologies such as GPS, only the indoor is uncharted.

Due to limitation of the GPS there has been an increase in the interest of these particular field with the use of complementary technologies for indoor positioning and navigation.

This document will present a solution for indoor positioning strictly using Wi-Fi, allowing for an user to reach its destination point. For that it will be used a voting technique using a fingerprinting pre-calibrated network to find the position of the user.

The area of interest will be the floor 1 on building C8 on the premises of faculty of sciences of the University of Lisbon that allows for a user to inquiry paths for a corresponding point of interest.

For that, this project has the purposes of developing an agnostic LBS, through the exposure of REST interfaces that allow its reusage in another spaces of interest in the premises, as long as the data is provided in a compliant way.

The results were achieved through the development on an *Android* application that communicates with a pre-established API installed in an applicational server, allowing that an user of the application might query paths to reach its point of interest, being that, while he transverses this path its position is updated only with the resources of the Wi-Fi.

Keywords: LBS, Wi-Fi, *indoor* positioning, *Fingerpriting*, REST, A*ndroid*

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos

AES – Advanced Encryption Standard

AoA – Angle of Arrival

AP – Access Point

API – Application programming interface

BLE – Bluetooth Low Energy

BSSID – Basic Service Set Identifier

CRUD – Create, Read, Update, Delete

GPS - Global Positioning System

HTTP – Hyper Text Transfer Protocol

IMU – Inertial measurement units

IOC – Inversion of Control

ISP – Internet Server Provider

JSR – Java Specification Request

JPA – Java Persistence API

JTA – Java Transaction API

LBS – Location based services

LED – Light emitting diode

Li-Fi – Light Fidelity

LSE – Least Squares Estimation

MAC – Media Access Control

PDR – Pedestrian dead reckoning

REST – Representation State Transfer

RF – Radio Frequency

RFC – Request for Comments

RFM – Reference Fingerprinting map

RSSI – Received signal strength

RTLS – Real Time Location Systems

SDK – Standard Development Kit

SO – Sistema Operativo

TDoA – Time difference of arrival

TOA – Time of arrival

UI – User Interface

UMTS – Universal Mobile Telecommunications Service

URL -Universal Resource Locator

VLC – Visible Light Communication

VOR – VHF Omnidirectional Range

WAP – Wireless Application Protocol

WINS – Wi-Fi Indoor Navigation System

WLAN – Wireless Local Area Network

WPS – Wi-Fi Position System

Capítulo 1 - Introdução

Com a evolução da nossa sociedade para uma sociedade digital tem existido um acentuado crescimento do uso de novas tecnologias. Se classificarmos a informação geográfica como o posicionamento das "coisas" (Continentes, Países, instituições, edifícios, gabinetes, pessoas) podemos dizer que o posicionamento é critico na sociedade atual.

No contexto urbano em que hoje nos inserimos, facilmente nos encontramos conectados, quer seja através de redes sociais, quer através da simples compra do passe numa caixa automática da rede Multibanco. Podemos até dizer que foi criada uma elevada dependência entre o nosso estilo de vida com a tecnologia sendo refletida no contínuo aumento de empregos nas áreas das tecnológicas (Premack, 2018). Os exemplos são diversos de como empresas ligadas à tecnologia tem reescrito o modo como encaramos o nosso dia-a-dia. Empresas que assentam num modo colaborativo de consumo, como o *Facebook* – a empresa mais popular de conteúdos de media que não produz qualquer conteúdo, mas que refez o modo como divulgamos e acedemos a informação; *Uber* – empresa de "táxi" que não possui qualquer veículo, mas que eventualmente acabará com o conceito de propriedade automóvel; *AirBnB* – plataforma mais utilizada para aluguer de imóveis que não ocupa qualquer espaço físico de aluguer, que alterou o modo como um turista encara uma cidade antes de a visitar potenciado até o rejuvenescimento destas; *Amazon* – que mudou o modo como a distribuição é realizada e como o consumidor pode aceder a estes bens; são apenas alguns destes exemplos (Grozdanic, 2015).

Saber onde eu posso "fazer algo" (comer, beber, passear o animal de estimação, o que está no cinema, etc.) é atribuído à facilidade de acesso a informação. Especialmente através do uso de *apps* que não só permitem aceder à informação, mas também interagir com esta. Só no ano de 2015 foram registados 10.38 mil milhões de USD$ de receitas em aplicações de dispositivos móveis (Dogtiev, 2018).

Nos últimos anos o desenvolvimento de aplicações que se encontram na palma da nossa mão, através da utilização *smartphones*, tem revolucionado o modo como os negócios são realizados atualmente. A proximidade e facilidade de interação com estas aplicações dita a "direção" das massas. Estas *apps* tem ditado como as massas podem interagir com o que as rodeia. E é nesta proximidade que alguns problemas podem surgir.

Tendo em conta o problema do posicionamento podemos dizer que o posicionamento em espaços *outdoor* está resolvido. É um facto que empresas como a *Google* dominam a informação Geográfica e a moldam conforme a querem, mas quando se trata de posicionamento *indoor* tal não acontece, tanto por limitações legais (legislação) como tecnológicas (sinais utilizados por GPS não atravessam barreiras físicas como edifícios).

Olhando para este contexto, um dos aspetos que deverá ser explorado reside na falta de disponibilização de serviços baseados na localização em ambientes *indoor*. Estes serviços devem permitir às pessoas explorar e utilizar espaços *indoor* da mesma maneira que já utilizam o *outdoor*. Ou seja, serviços em que os seus principais *inputs* sejam dados geográficos, denominados serviços baseados na localização (*Location Based Services* - LBS) (Schiller & Voisard, 2004).

Debruçando-nos sobre este assunto, multinacionais dificilmente conseguirão "mapear" e providenciar serviços para posicionamento *indoor* da mesma maneira que o fizeram para o outdoor.

Desse modo tem existindo tentativas de adaptar serviços de posicionamento através de tecnologias paralelas onde o *Global Positioning System* (GPS) é removido da equação, passa apenas a "conduzir" as pessoas até a um ponto genérico de informação no mapa, como uma porta de um edifício, e aqui outra tecnologia, assume a responsabilidade de levar um utilizador ao sítio pretendido.

Um exemplo de um tipo de tecnologia deste tipo é o *Tango*. Este sistema utiliza a realidade aumentada para triangular a posição através de *distinct visual features in the environment* ao qual a *Google* já a denominou de *visual positioning service* (Lomas, Tech Crunch, 2017).

Outros exemplos prendem-se com a utilização de outros tipos de sensores que possibilitam o posicionamento indoor. Principalmente com tecnologias wireless, como o Wi-Fi, *Bluetooth* e até mesmo Luz (Li-Fi).

> "*The moment we get cut off from GPS the mapping experience becomes rubbish. We need something to fill the gap for mapping indoors, where accurate positioning is difficult. We're not clear on what that will be yet – whether Bluetooth beacons or another suite of sensors*". (Gibbs, 2015)

De facto, espera-se que estes serviços continuem a crescer, tendo em 2017 sido valorizado em 5.681 mil milhões de USD e as previsões apontam que cheguem aos 30.575 mil milhões de USD em 2023 (Markets, 2018).

Este fator é algo benéfico já que o posicionamento *indoor* traz benefícios que não se limitam apenas a questões económicas, mas também sociais. Por exemplo, situações em que um serviço de urgência precise de chegar a um cidadão, estes serviços irão agilizar e baixar o risco de vida das pessoas já que a possibilidade de socorristas não saberem exatamente onde a pessoa em risco se encontra num edifício é real, e mesmo que por alguma razão esta informação tenha sido providenciada, não tem modo de como chegar até esta da maneira mais eficiente e rápida.

1.1. Motivação

Conforme mencionado, existe um crescente interesse na área dos serviços de localização *indoor*, tanto do ponto de vista económico como social. Assim sendo, existe um potencial de crescimento que permite

a esta área se desenvolver e expandir tendo um palco muito interessante e que se irá certamente desenvolver nos próximos anos, e potenciar também o advento de tecnologias relacionáveis com esta temática. Podendo dar aos utilizadores destes serviços novas interações com os espaços que as rodeiam. Principalmente em espaços tipo "labirinto" como é por vezes o caso de campus universitários devido ao seu tamanho e complexidade. Tal é o caso da Faculdade de Ciências da Universidade de Lisboa.

Com cerca de 75 000 m^2 e 8 edifícios, uma estrutura de acessos complexa, em que por vezes o acesso entre edifícios é feito somente de uma forma quase desenhada para uma pessoa se perder, um sistema baseado na localização ajudaria pessoas novas a deslocarem-se no campus. Acrescentaria valor à maneira como as pessoas se deslocam, não só a pessoas novas, mas como a utilizadores assíduos do espaço. Exemplo práticos, como por exemplo saber que um acesso se encontra interdito ou que um elevador se encontra avariado tornaria o desafio de se deslocar neste espaço bastante menos desafiante.

Desse modo a criação de um LBS mitigaria o desafio de deslocação neste espaço. E é sobre esta temática que este projeto se foca.

A aspiração de ter um produto que englobasse esta temática e pudesse ter os diversos utilizadores a utilizar e interagirem com a plataforma de modo a serem mais 'ágeis' a deslocarem-se no campus de forma eficiente, tanto *indoor*, como *outdoor* seria algo como um *coup de gras* ao labirinto que é a área da Faculdade de Ciências. Mas tal sai do âmbito de uma dissertação pelo que este projeto apenas se irá debruçar sobre a vertente *indoor*, deixando a porta aberta para ser expandida e incrementada conforme as consecutivas interações sobre esta.

1.2. Objetivo

O objetivo deste projeto é desenvolver um protótipo de uma aplicação de navegação *indoor* baseado numa rede Wi-Fi.

Como qualquer sistema aplicacional em que exista uma interação entre utilizador – máquina, um LBS tem um conjunto de dependências informáticas que permitem esta iteração. O objetivo é criar estas dependências para poderem alimentar um LBS, sendo que o foco seja na componente *indoor*. Mais precisamente no algoritmo de posicionamento que permita acumular funcionalidades conforme o seu crescimento. O resultado será uma aplicação que permita a um utilizador navegar através de uma área delimitada do campus, e que permita fazer consultas de qual o caminho para chegar a determinados pontos de interesse.

De modo a ser possível criar um sistema que tenha espaço para crescimento e não fique "acoplado" a uma aplicação é necessária segmentar este LBS em duas componentes principais. A primeira prende-se com o servidor. Ou seja, uma estrutura que disponibilize um conjunto de interfaces que possa ser consultado por um "cliente" e que disponibilize a informação necessária do modo mais agnóstico

possível, não ficando enviesado ao cliente em questão, mas tendo um foco no "negócio" de posição geográfica. Não só deverá providenciar dados geográficos referentes ao campus como deverá permitir a criação de dados geográficos. A segunda está ligada ao cliente. Ou seja, à componente que consome as interfaces disponibilizadas pelo servidor e mostra os dados de uma forma legível para o utilizador. Portanto, a aplicação móvel que recolhe dados e envia ao servidor de modo a que este possa inferir a localização do mesmo e assim, permita ao utilizador que este interaja com o espaço em redor e assim se desloque ao longo deste.

Deste modo, de modo a ser possível a segmentação mencionada em cima os passos para a realização do projeto foram subdivididos e ordenados por sequência nos seguintes aspetos:

- Levantamento científico e tecnológico de possíveis requisitos. Sendo que é uma área com algum investimento é certo que já existem aplicações que tenham criado o seu LBS. Desse modo é realizado o levantamento do estado de arte sobre tecnologias existentes, de que modo são implementadas, vantagens e desvantagens e que sistemas operativos existem.
- Levantamento de qual o tipo de arquitetura aplicacional adequado à comunicação entre servidor – cliente. De que modo a componente servidor pode ser disponibilizada de modo a ser escalável, tanto funcionalmente como aplicacional e que conjunto de boas práticas devem ser levadas em conta neste desenho e implementação.
- Levantamento de algoritmos de posicionamento indoor tendo em conta os dados disponíveis e construir em cima destes.
- Desenho da arquitetura a implementar no servidor. De que modo deve ser implementado o servidor, de modo a que o seu crescimento não cause qualquer impacto para as interfaces disponibilizadas a serem consumidas pelo cliente, de maneira a que o algoritmo de posicionamento não limite estas.
- Desenho de arquitetura e implementação do cliente. Levantamento de boas práticas a serem utilizadas pela aplicação cliente, tanto do ponto de vista aplicacional como a nível de experiência de utilizador permitindo o consumo e envio de dados para o servidor através das interfaces disponibilizadas.

1.3. Contribuição Tecnológica

Toda esta dissertação assenta no pressuposto de utilizar ferramentas open-*source*. Aquando da sua publicação irá ser disponibilizado todo o seu código fonte num servidor aberto, onde será disponibilizado e encorajado o contributo de outros indivíduos que queiram contribuir e fazer crescer o LBS da Faculdade de Ciências como um produto, já que o objetivo deste projeto não é terminar, permitir a criação de uma aplicação móvel que permita a utilizadores interagirem com a sua Faculdade e eventualmente integrarem com serviços de localização externos, como por exemplo o *Open Street Map*.

1.4. Estrutura da Dissertação

Esta dissertação encontra-se dividida da seguinte forma:

Capítulo 2 – Corresponde ao levantamento do estado de arte. De que modo um sistema de posicionamento *indoor* funciona e que possíveis limitações possam surgir ao longo da sua implementação. Será identificado as várias abordagens existentes e que vantagens e desvantagens existem entre elas.

Capítulo 3 – Identifica que abordagem será utilizada, assim como providencia algumas noções do que é pretendido alcançar. É detalhada o objetivo da arquitetura escolhida de um ponto de vista abstrata.

Capítulo 4 – Este, corresponde à consolidação dos vários temas identificados nos capítulos anteriores. Detalha de que modo cada componente do sistema se encontra implementado e que tecnologias foram utilizadas. Identifica também de que maneira o sistema se interliga e comunica entre si, sendo apresentado no final um conjunto de ensaios que demostram o funcionamento e precisão do sistema.

Capítulo 5 – Por último este capítulo realiza um ponto de situação do sistema e se este cumpre os requisitos inicias descritos. É também mencionado as limitações do sistema assim como algumas considerações para futuros trabalhos.

Capítulo 2 - Estado da Arte

Este capítulo tem como objetivo realizar o levantamento de informação com uma componente técnica, que possibilite a criação de um *Location Based Service* (LBS) descrito no capítulo anterior.

Encontra-se dividido em três secções:

- Arquitetura do Sistema – de que modo pode ser construído um LBS e que arquitetura aplicacional pode ser utilizada. Aplicações somente cliente, ou seja, não necessitam necessariamente de um servidor para efetuar posicionamento e/ou navegação. Aplicações onde o posicionamento é efetuado por parte da estrutura.
- Técnicas de posicionamento – transversal a qualquer tecnologia utilizada. Aqui é realizado o registo matemático e científico de como é possível efetuar posicionamento de um modo agnóstico à tecnologia utilizada.
- Tecnologias – modo de implementação de um LBS *indoor* mediante a tecnologia escolhida. São escolhidas as tecnologias Wi-Fi, *Bluetooth* e *Visible Light Communication* (VLC), já que estas são as tecnologias mais utilizadas e mais recentes.

2.1. Arquitetura do Sistema

O posicionamento e a navegação *indoor* estão cada vez mais a tornar-se relevantes para diversas indústrias. Do ponto de vista de arquitetura aplicacional existem diversas maneiras de o realizar, quer seja através de soluções baseadas em cliente, quer seja através de soluções baseadas em servidor, sendo que existam abordagens híbridas que utilizam ambos os tipos de arquitetura.

O posicionamento baseado no cliente significa que a posição é detetada num dispositivo móvel (por exemplo um *smartphone*). Este é realizado através de uma aplicação, que calcula a posição baseada em pontos de acesso, como por exemplo um router ou um AP de Wi-Fi, *Beacons Bluetooth* ou *Light emitting diode* (LED).

Quando o posicionamento se baseia num servidor, existe a identificação de um dispositivo. Este dispositivo envia um identificador único para pequenos recetores que se encontram distribuídos pela área de interesse. Estes dispositivos podem por sua vez estar ligados a um servidor, obtendo deste modo a posição do dispositivo, ou podem devolver dados para o dispositivo, que este então realize a comunicação com um servidor providenciando os dados necessários para saber o posicionamento do dispositivo.

2.1.1. Baseado em Cliente

O posicionamento é adquirido diretamente no dispositivo. Tem de existir uma aplicação instalada que analise os sinais dos pontos de acesso de dados como um router Wi-Fi, um LED e/ou *beacons*. A aplicação terá de conter uma base de dados onde a fonte e a força do sinal sejam identificadas. Deste modo, o equipamento pode detetar a sua posição sem ter de estar necessariamente ligado a um ponto de acesso. O proprietário da aplicação pode enviar mensagens aos visitantes (mensagens denominadas *push*) – por exemplo publicidade baseada em localização ou informação útil.

O posicionamento baseado em cliente não quer dizer que os dados do utilizador nunca saem do dispositivo. Podem existir soluções que enviem os dados do ambiente em redor e enviem estes para um servidor. Este por sua vez responde para o dispositivo com a sua localização, regra geral este tipo de arquitetura utiliza *fingerprinting* para obter uma precisão da posição do dispositivo elevada.

2.1.2. Baseado em Servidor

Este tipo de posicionamento deteta todos os dispositivos e não existe a necessidade de uma aplicação, embora perca valor, já que apenas consegue identificar que existe "algo" no espaço, não consegue saber "quem", já que não existe uma relação do dispositivo com outra informação, somente "existe".

Quando combinado com equipamentos e/ou pessoas, devidamente identificadas, é possível detetar o "onde" e o "quem", como por exemplo o posicionamento de medicamentos médicos numa clínica, veículos em fábricas, ou bens em retalho. Deste modo podem existir mecanismos de monitorização que permitem saber quando um objeto deixa uma área especificamente definida (proteção contra roubo).

2.2. Técnicas de Posicionamento

Antes de ser possível identificar como e quais as tecnologias que podem permitir criar um LBS *indoor* é necessário perceber e entender de que modo um sistema de posicionamento é criado. Desse modo olhando para o que já existe criado no *outdoor* (excluindo sistemas que utilizem dispositivos satélites em órbita, como o GPS) podemos facilmente identificar os sistemas de telecomunicações como sistemas que utilizam técnicas para identificar a posição de um indivíduo, e quais as vantagens e desvantagens das várias técnicas (Cisco, 2008).

Os sistemas de posicionamento podem ser classificados pelas técnicas de medição que utilizam para determinar a localização de um cliente (através de um dispositivo, como por exemplo o telemóvel). As abordagens podem variar em termos das técnicas utilizadas para detetar e medir a posição do dispositivo móvel no ambiente envolvente. Tipicamente, os *Real Time Location Systems* (RTLS) podem ser agrupados em quatro categorias para encontrar a posição:

- *Cell of origin* - Célula de origem (célula mais próxima)

- *Distance* - Distância (*lateration*)
- *Angle* - Ângulo (*angulation*)
- *Location patterning* - Padrão de localização (*pattern recognition*)

É necessário levar em conta que por ser utilizada uma técnica, não quer dizer necessariamente que não se possa utilizar outra, ou seja, existe a possibilidade de implementar uma ou mais destas técnicas simultaneamente. Pode ser observado situações em que as abordagens utilizadas tentam otimizar a precisão em dois ou mais ambientes com diferentes características de propagação de sinal. A popularidade de RTLS é tanto que geralmente surge uma quinta categoria que envolve um RTLS que tenta medir a posição utilizando pelo menos dois destes métodos.

É importante reter que, independentemente da tecnologia utilizada, a natureza de "tempo real" de um RTLS é apenas tempo real relativo a um carimbo temporal (*timestamp)* da leitura da força de sinal ou a medição do ângulo de incidência. O tempo de resposta entre o servidor – cliente podem introduzir discrepâncias entre o valor reportado pelo dispositivo e o servidor.

2.2.1. Técnicas Baseadas em "Células" - *Cell of Origin*

Uma das formas mais simples de estimar uma localização aproximada em qualquer sistema baseado em células é o conceito de *cell-of-origin*. (Figura 2.1)

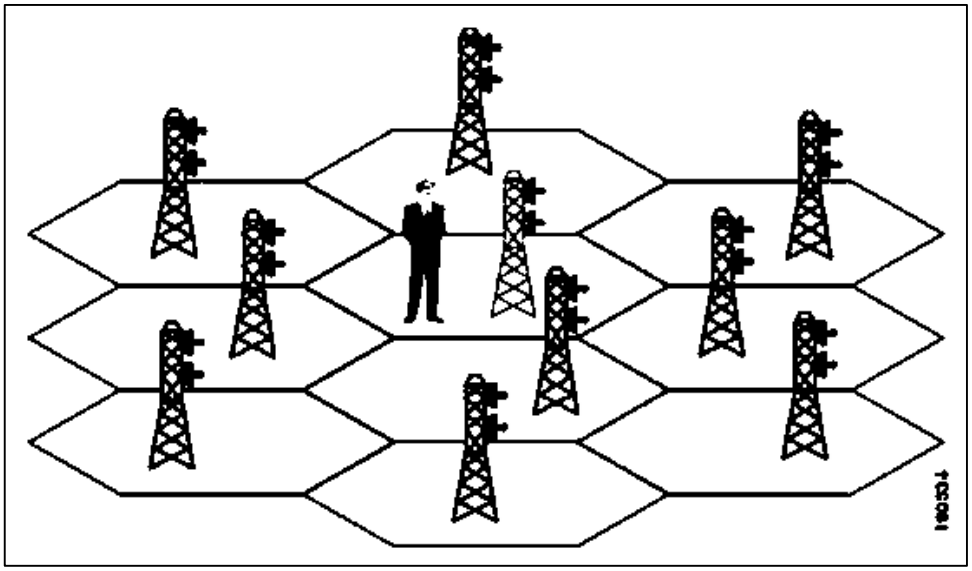


Figura 2.1: Técnica de Posicionamento utilizando *Cell-of-Origin*. (Cisco, 2008)

Na sua forma mais simples é uma técnica que não faz nenhuma tentativa explícita para resolver a posição do dipositivo móvel para além de indicar a célula em que este dispositivo está ou foi registado. A principal vantagem desta técnica é a sua simplicidade de implementação. Esta não necessita da implementação de algoritmos complicados e complexos, por isso o desempenho computacional para obter o posicionamento é bastante rápida. Quase todas as redes WLAN (*Wireless Local Area Network*)

baseadas em células ou outros sistemas *Radio Frequency* (RF), baseados em células, podem ser facilmente adaptados para providenciar um posicionamento. Contudo, a maior desvantagem desta abordagem está relacionada com a granularidade do posicionamento, onde em áreas urbanas podem ter uma precisão entre 100 e 1000 metros (Rouse, Search Mobile Computing, 2010). Por diversas razões, os clientes podem ser associados a células que não estão nas proximidades, apesar do facto de existirem outras células candidatas. Esta granularidade pode ser especialmente frustrante quando se tenta estimar a posição atual de um cliente numa estrutura de vários andares onde existe uma sobreposição considerável de células.

De modo a determinar que áreas da célula possuem a maior probabilidade de conter o dispositivo móvel na sua área, é necessário um método adicional para resolver a localização dentro da área da célula. Isto pode passar por um método manual (como uma pessoa procurar na área toda da célula pelo dispositivo) ou um método assistido computacionalmente. Quando se recebe um sinal de uma célula estas providenciam uma força do sinal – RSSI (*received signal strength indication*) – que o dipositivo móvel capta. O uso de técnicas em que mediante a força do sinal é estimado qual a célula mais provável pode aumentar a granularidade sobre a célula de origem. Neste tipo de abordagens, a localização do dispositivo móvel é alcançada com base na célula que consegue detetar o dispositivo móvel em que o sinal seja mais forte.

Deste modo se for utilizada esta técnica, a probabilidade de selecionar a verdadeira célula de origem é significativamente melhor. Dependendo dos requisitos e necessidades da precisão, o desempenho deverá ser mais do que suficiente para localizar dispositivos moveis.

Assim esta técnica pode ser utilizada em situações em que a precisão do posicionamento não seja critica já que por norma tem uma relação esforço-desempenho aceitável. Contudo, casos em que seja necessária uma maior precisão no sistema de localização esta técnica poderá ficar aquém das necessidades. Desse modo existem outras técnicas que conseguem obter melhores precisões de posicionamento, técnicas essas a serem descritas nas próximas secções.

2.2.2. Técnicas Baseadas na distância (*Lateration*)

2.2.2.1. Tempo de chegada - Time of Arrival (ToA)

As técnicas ToA são baseados na medição precisa do tempo de chegada do sinal transmitido de um dispositivo móvel a diversas estações. Como os sinais deslocam-se a uma velocidade conhecida (cerca da velocidade da luz +- 300 metros por microssegundo, Moshe (2019), a distância entre um dispositivo móvel e cada estação pode ser determinado pelo tempo de propagação entre estes. A técnica requer um conhecimento muito preciso do tempo de início da transmissão e tem de ser garantido que todas estações, assim como os dispositivos móveis, se encontram sincronizados exatamente a partir da mesma fonte.

Conhecendo tanto a velocidade como a velocidade de propagação, é possível calcular a distância entre o dispositivo móvel e a estação recetora.

A representação circular da área à volta das estações pode ser construída para onde a localização do dispositivo móvel tem uma alta probabilidade de estar presente. Informação do ToA de duas estações resolve a posição do dispositivo móvel para dois pontos igualmente prováveis. ToA *tri-lateration* utiliza três estações e permite ao dispositivo móvel ser encontrado com uma precisão mais elevada (Figura 2.2).

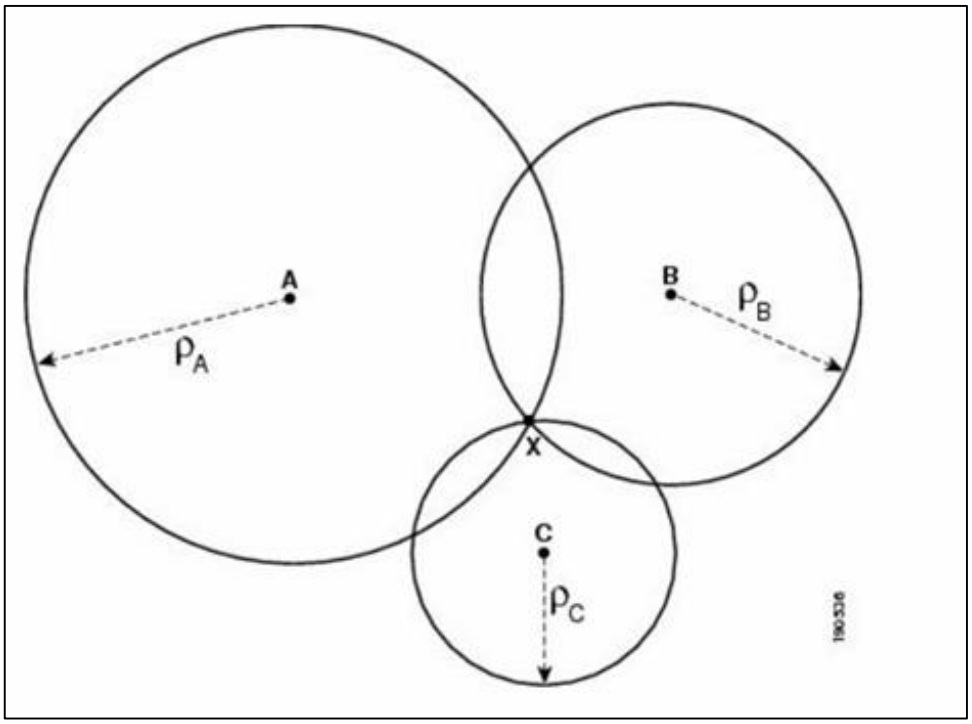


Figura 2.2: Técnica de Posicionamento utilizando *Time of Arrival*. (Cisco, 2008)

O tempo necessário para a mensagem transmitida da estação A, B e C é medida como p_A, p_B e p_C. Sabendo a velocidade de propagação, a distância do dispositivo móvel e as três antenas pode ser calculada como D_A, D_B e D_C. Cada distância calculada é utilizada para construir um gráfico circular à volta das respetivas antenas. Do ponto de vista individual de cada estação, o ponto *X* é estimada com alguma precisão que existe algures no gráfico circular correspondente. A interseção destes gráficos circulares acabar por resolver a posição do dispositivo móvel X como ilustrado na Figura 2.2. Em alguns casos, pode existir mais do que uma possível solução para a localização do dispositivo móvel X, mesmo quando utilizando três estações. Nestes casos, quatro ou mais estações são necessárias para desempenhar ToA *multi-lateration*.

Técnicas de ToA são capazes de encontrar a localização num espaço de duas dimensões assim como três dimensões, já que a resolução 3D pode ser alcançado através da construção de um modelo esférico em vez de circular.

Uma limitação desta técnica é a necessidade de sincronização ao nível do nanosegundo de todas as estações, especialmente o dispositivo móvel. Dado que a velocidade de propagação é elevada as pequenas discrepâncias na sincronização do tempo podem resultar em grandes erros no cálculo da localização. Por exemplo, um erro de tempo tão pequeno como 100 nanossegundos pode resultar numa discrepância de 30 metros. Soluções baseadas em ToA são tipicamente desafiantes em ambientes onde existem várias quantidades, interferências ou "ruídos" físicos.

Numa aproximação o GPS é um exemplo de um sistema ToA onde o timing de precisão é alcançado através de relógios atómicos.

2.2.2.2. Diferença de Tempo de Chegada – Time Diference of Arrival (TDoA)

Esta técnica utiliza medidas de tempo relativas em cada estação em vez de medidas de tempo absoluto. Graças a isto não necessita de utilizar fontes de tempo sincronizado entre as diversas estações e dispositivos móveis o que diminui significativamente a dificuldade do posicionamento. Com TDoA, uma transmissão com uma fonte desconhecida de tempo é rececionada em várias antenas, onde apenas as antenas recetoras necessitam de se encontrar sincronizadas.

Implementações deste género tem por base um conceito matemático conhecido como *hiperbolic lateration*. Nesta abordagem, pelo menos três estações recetoras necessitam de estar sincronizadas.

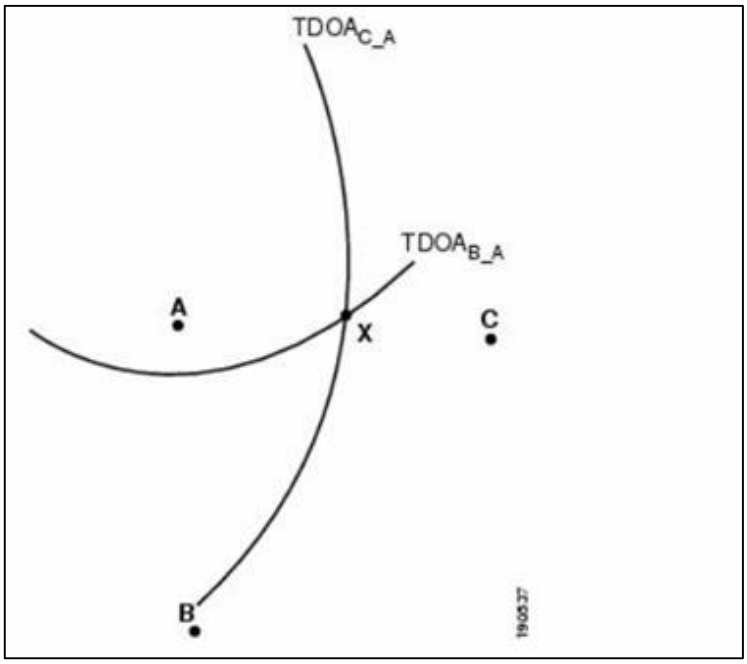


Figura 2.3: Técnica de Posicionamento utilizado Time Diference of Arrival. (Cisco, 2008)

Na Figura 2.3, é assumido que quando a estação A transmite uma mensagem, esta mensagem chega ao dispositivo móvel X com o tempo T_A e que à estação *B* com tempo T_B. Esta diferença de tempo de chegada da mensagem é calculada entre as estações *B* e *A* como a constante *k*.

O valor de $TDoA_{B-A}$ pode ser utilizado para construir a hipérbole com focis nas localizações de ambas as estações A e B. Esta hipérbole representa o locus de todos os pontos no plano x-y, a diferença destas distâncias de os dois foci é igual a k(c) metros. Matematicamente, isto representa todas as localizações possíveis do dispositivo móvel X.

A localização provável do dispositivo móvel X pode ser representada por um ponto ao longo desta hipérbole. De modo a aumentar a precisão da localização do dispositivo móvel X, uma terceira estação na localização C é utilizada para calcular o tempo de diferença da mensagem entre as estações C e A.

O conhecimento da constante permite a construção de uma segunda hipérbole representado o locus de todos os pontos no placo x-y, a diferença é que as distâncias dos dois foci (isto é, as das estações A e C) é igual a k1(c) metros.

Uma quarta estação e uma terceira hipérbole podem ser adicionadas para fazer TDoA *hiperbollic multi-lateration*. Isto pode ser necessário para resolver casos onde TDoA *hyperbolic tri-lateration* providencia mais do que uma solução de posição.

Os sistemas modernos de TDoA tem métodos derivativos de lidar com osciladores locais de relógios que são utilizados para evitar restrições estritas de precisão de sincronização com recetores. Por exemplo, ajustes de tempo podem ser calculados periodicamente em relação a uma fonte de tempo única. Estes ajustes de relógio podem depois ser utilizados para corrigir o relógio de referência algures no sistema. No caso de recetores TDoA que são capazes de transmitir dados, outra abordagem pode envolver a troca periódica de pacotes de *timing* entre os vários recetores. Assim, diferenças de tempo entre cada recetor e um "recetor de referência podem ser contabilizados, com o ajuste aplicado dentro do sistema.

Os sistemas de alcance de aeroportos são um exemplo bem conhecido de sistemas. No mundo de telecomunicações móveis, TDoA é também mencionado como *Enhanced Observed Time Difference* (E-OTD), onde é possível ter um alcance, num ambiente *outdoor*, de cerca de 60 metros em áreas urbanas e de 200 metros em áreas rurais embora estes valores não possam ser tomados como absolutos.

TOA e TDoA tem diversas similaridades. Ambos provam que são bastante adequadas para sistemas de posicionamento em ambientes *outdoor*. Para além disso, tem obtido bons resultados em ambiente *semi-outdoor* como anfiteatros e estádios, assim como ambientes *indoor* com muito espaço livre e poucas barreiras físicas, como stands automóveis e armazéns abertos. Estes sistemas apresentam melhor performance em edifícios que são grandes e relativamente abertos, com baixos nível de obstrução e tetos elevados que permitem grandes áreas de aberturas entre o conteúdo do edifico e o seu teto. É precisamente nestes cenários que sistemas TDoA e ToA operam no seu pico de precisão e *performance*.

2.2.2.3. Força de Sinal Rececionada – Received Signal Strenght Indicator (RSSI)

Até ao momento já foram descritas duas técnicas de *lateration* (ToA e TDoA) que usam o tempo para medir a distância. *Lateration* também pode ser obtida se for utilizado a força do sinal recebida (RSSI) em vez do tempo. Com esta abordagem, o RSSI é medido pelo dispositivo móvel ou o sensor recetor. De forma a ser possível identificar o dispositivo no espaço é necessário obter um conjunto de dados, tais como a localização dos emissores recetores, força de *output* do transmissor e quais as perdas de propagação do sinal entre o emissor e o recetor que é obtido através de um modelo matemático.

Um modelo matemático que represente perdas de sinal poder ser resumindo entre a diferença entre o nível de sinal transmitido, medido da antena emissora, e o nível de sinal recebido, medido na antena recetora. Não leva em conta perdas de sinal e representa o nível de atenuação do sinal presente no ambiente devido a efeitos de propagação de espaço live, reflexão, difração e dispersão (Cisco, 2008).

Conforme a distância entre emissores e recetores poderá existir um rácio proporcional á distância destes. Fatores físicos como a frequência, o ambiente e o grau de obstruções (ou "desordem") presentes no ambiente (tais como paredes, portas e até mesmo outros dispositivos).

Resolvendo a distância entre o recetor e o dispositivo móvel é possível gerar um gráfico da área envolvente do recetor, sendo que a localização do dispositivo móvel é estimada que esteja algures contido nesta área. Como noutras técnicas, o *input* de outros recetores em outras células (neste caso, informação da força do sinal) pode ser utilizado para realizar RSSI *tri-lateration* ou RSSI *multi-lateration* aumentando a precisão da localização.

A informação da força do sinal utilizada para determinar a posição pode ser obtida de duas fontes:

- o dispositivo móvel que reporta a intensidade do sinal recebido proveniente da infraestrutura (*client-side*)
- a infraestrutura que reporta a forço do sinal recebido do dipositivo móvel (*network-side*)

Em redes Wi-Fi a granularidade em que RSSI é reportada pelo dispositivo (ou cliente) tipicamente varia de implementador para implementador. Por isso, conforme a implementação, muitos dispositivos acabam por reportar forças de sinal utilizando métricas inconsistentes e dispares entre si, tornando a sua integração algo propenso a erros que por sua vez leva a posicionamentos inconsistentes.

Um dos modos de evitar este problema, passa por determinar a localização que utiliza medidas de RSSI em *network-side* sendo que se tornam agnósticas às implementações dos fabricantes, já que todas as medidas de RSSI são realizadas na infraestrutura da rede e não no dipositivo móvel. Esta é solução que geralmente é abordada e implementada por fabricantes de soluções de *lateration* com RSSI, já que o nível de controlo é mais elevado e tipicamente garante mais consistência.

Implementações que recolham o RSSI através do dispositivo devem utilizar um conjunto de passos para evitar baixas precisões de posicionamento resultante de inconsistências de *hardware* entre diversos

dispositivos moveis. Sendo que não é realístico assumir que todos os dispositivos moveis irão providenciar o mesmo *hardware*, é necessário um método que "equilibre" qualquer variação em relação a algum modelo de referência.

Um modo de resolver este problema passa pela normalização dos valores. Por exemplo, assumir que uma posição de um sistema que espera que o sinal RSSI varia entre os -127dB e +127dB em 254 incrementos de 1dBm cada torna-se invalido quando confrontado com dispositivos móveis que reportem o RSSI entre -111dB a +111dB em 74 incrementos de 3dBm cada. Desse modo é necessário normalizar estes valores de modo a que as discrepâncias entre os dispositivos sejam atenuadas.

Tipicamente, a responsabilidade de fornecer tal normalização recai no fornecedor da solução de localização.

Uma das grandes vantagens pela implementação que utilizem esta técnica passa pelo custo, já que não é necessário *hardware* específico do lado do dispositivo móvel ou na infraestrutura. Isto torna estas técnicas bastante atrativas de um ponto de vista custo/performance. Contudo, uma desvantagem é que anomalias de propagação com origem em condições anisotrópicas no ambiente podem degradar a precisão de posicionamento significativamente. Isto é devido em parte porque a propagação em qualquer estação está longe de ser um padrão puramente circular.

Certos modelos matemáticos não levam em conta a da medição de variações observadas em sítios específicos, tipicamente assumem valores conhecidos de perca de sinal. Algumas não levam em conta passos para levar em conta atenuantes de *multipath*, o que dificilmente conseguem produzir resultados aceitáveis, exceto em situações muito especificas, tais como situações controladas onde existe sempre uma linha de visão "limpa" e estabelecida entre o dispositivo móvel e os sensores.

2.2.3. Técnicas Baseadas no Ângulo (*Angulation*)

2.2.3.1. Ângulo de Chegada – Angle of Arrival (AoA)

Por vezes mencionado como direção de chegada (*Direction of Arrival* - DoA), localiza o dispositivo móvel ao determinar o ângulo de incidência do sinal que chega à estação. Pode depois utilizar relações geométricas para estimar a localização da interseção de duas linhas formadas por uma linha radial em cada estação conforme na Figura 2.4.

Num espaço de duas dimensões, são necessárias pelo menos duas estações para estimar a localização com alguma precisão.

Na sua forma mais "pura" – que consiste numa linha de visão limpa entre o dispositivo móvel X e as estações A e B – antenas mecanicamente instaladas nas estações são ajustadas em função do sinal mais

forte. O posicionamento das antenas direcionais pode ser diretamente utilizado para determinar e medir os ângulos de incidência de cada estação.

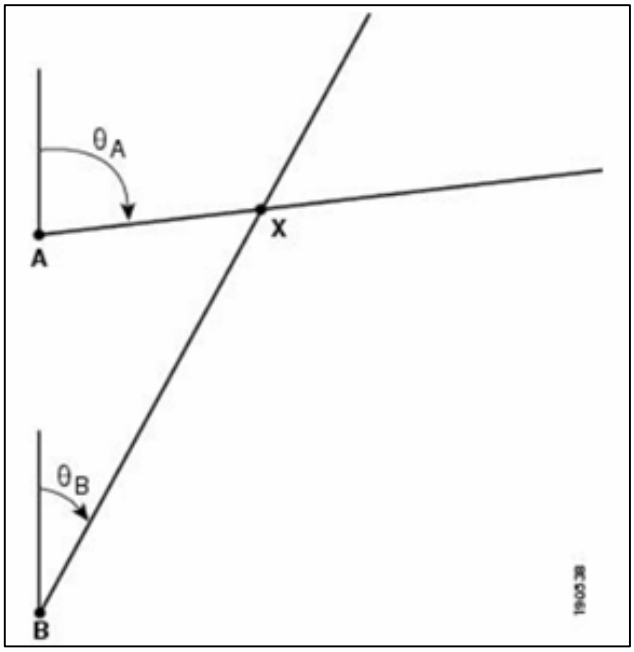


Figura 2.4: Técnica de Posicionamento utilizando Angulo de chegada. (Cisco, 2008)

Em implementações comerciais e militares, são utilizados múltiplos conjuntos de estações para recolher a amostra dos sinais, eliminando assim a necessidade de antenas mecânicas, que geralmente são mais complexas e tem um nível de manutenção elevado. Esta técnica atualmente envolve calcular o TDoA entre elementos de cada conjunto de antenas através da medição das diferenças recebidas em fases de cada elemento. Num conjunto de antenas propriamente construído, existe uma pequena, mas percetível perceção de tempo e de fase por elemento. Por vez referenciada por *reverse beam-foarming*, esta técnica envolve diretamente medidas de tempo de chegada de cada sinal a cada elemento, computado o TDoA entre os elementos do conjunto de antenas e convertendo esta informação para uma medida AoA. Isto é possível devido ao facto de que dentro do *beam-forming*, o sinal de cada elemento é atrasado pelo tempo (alteração entre fase) para "dirigir" o ganho do conjunto de antenas.

Uma implementação conhecida de AoA é o sistema VOR (*VHF Omnidirection Range*) utilizado para a navegação de aviões. Antenas VOR transmitem múltiplos "radiais" VHF em que cada um emite em diferentes ângulos de incidência. O recetor VOR num avião pode determinar em qual o radial em que este se encontra situado, à medida que se aproxima da estação VOR, e assim o angulo de incidência correspondente. Usando pelo menos duas estações VOR, o navegador do avião pode utilizar equipamento AoA que se encontram a bordo para realizar uma *angulation* (ou *tri-angulation* se utilizar três ou mais estações VOR) e determinar precisamente a localização do avião.

Estas técnicas podem também ser aplicadas em telecomunicações de modo a tentar fornecer a localização de utilizadores de telemóveis. Foi inicialmente utilizada para cumprir com os regulamentos que obrigam sistemas celulares a identificar a localização de um utilizador a reportar uma emergência. Múltiplos sinais de células de torres calculam o sinal AoA de um utilizador, e usam esta informação para realizar *tri-angulation*. Esta informação é passada para processadores que calculam a localização do utilizador e convertem os dados AoA para coordenadas de latitude e longitude, que depois é emitida para sistemas de emergências.

Uma desvantagem comum é que AoA, assim como algumas das outras técnicas mencionadas, é a suscetibilidade de interferências de *multipath*. Como mencionado previamente, AoA funciona bem em cenários em que a linha de visão é uma reta, mas a precisão sofre quando confrontado com a reflecção de sinais de objetos circundantes. Infelizmente em áreas urbanizadas, AoA torna-se praticamente inútil devido ao facto de raramente existirem duas estações (*base station)* com uma linha de visão desobstruída.

2.2.4. Técnicas Baseadas em Padrões de Localização (*Pattern Recognition*)

Corresponde a técnicas que se baseiam na amostragem de sinais com padrões de comportamentos em ambientes específicos. Uma solução que implemente *location patterning* não necessita de hardware especializado tanto nos dispositivos móveis como nas estações base. Pode ser implementado totalmente em *software*, o que pode reduzir a complexidade e o custo significativamente quando comparado com *angulation* ou sistemas baseados em *lateration*.

Fundamentalmente assumem o seguinte:

- Cada localização potencial de um dispositivo possui uma assinatura distinta. Quanto mais próximo da realidade, melhor é a performance do sistema.
- Cada piso ou subsecção possui caraterísticas de propagações únicas. Apesar de todos os esforços no posicionamento de equipamentos idênticos, nenhum piso, edifício, ou espaço são realmente idênticos do ponto de vista da perspetiva de *pattern recognition* RTLS.

Apesar de a maioria das soluções existentes se basear em assinaturas com base na força do sinal (RSSI), pode ser estendido para incluir ToA, AoA ou TDoA-based RF. As implementações podem ser tipicamente divididas em duas fases:

- Calibração
- Operação

Durante a fase de operação, o sistema leva em conta a possibilidade de igualar a assinatura RF de um dispositivo em relação a uma base de dados de assinaturas RF obtida durante a fase de calibração. Como a base de dados de assinaturas RF gravadas é suposta ser compilada durante o período representativo na

operação do site, variações como atenuações das paredes e outros objetos podem ser diretamente contabilizados na fase de calibração.

2.2.4.1. Fase de Calibração

Durante a fase de calibração, são recolhidos dados através da exploração do ambiente pretendido com um dispositivo móvel onde é permitido que múltiplas estações (pontos de acesso no caso de redes Wi-Fi) recolham uma amostra da força do sinal do dispositivo móvel (isto é referenciado como *network-side local patterning*).

Uma representação gráfica da área que vai ser calibrada é tipicamente "coberta" por um conjunto de pontos numa grelha ou notações para guiar o operador em determinar precisamente onde os dados de amostra devem ser adquiridos. Em cada localização de amostra, o conjunto (vetor de localização) dos valores RSS associados com a calibração do dispositivo são armazenados numa base de dados reconhecida como *training set*. O tamanho do vetor desta amostra é determinado pelo número de estações que podem detetar o dispositivo móvel.

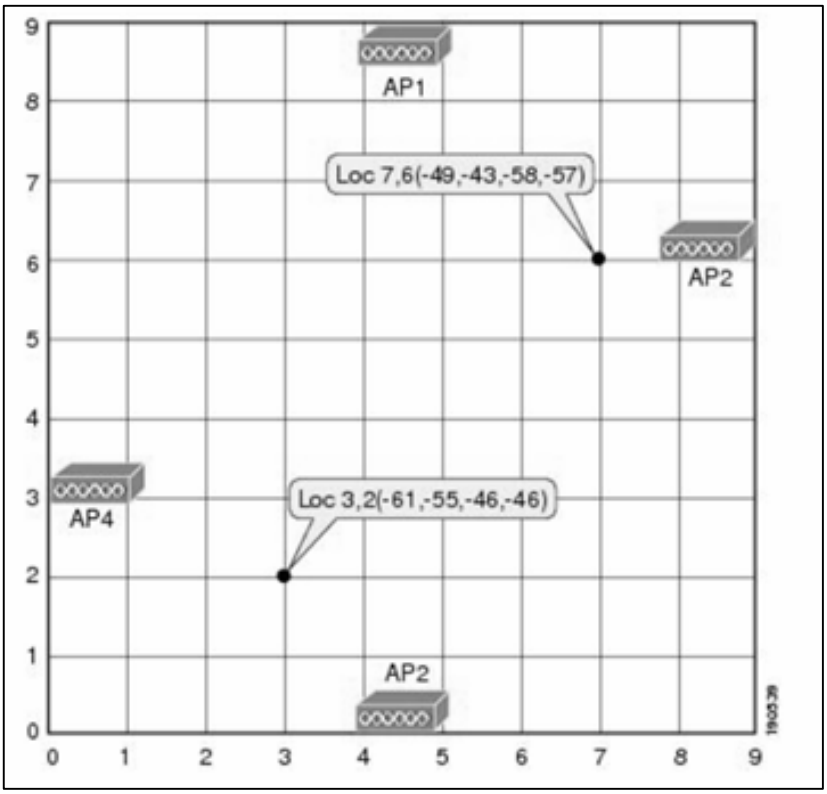


Figura 2.5: Representação 2D da fase de calibração. (Cisco, 2008)

A Figura 2.5 é uma ilustração simplificada desta abordagem, mostrando duas amostras e como os seus respetivos vetores de localização podem ser formados da deteção do cliente RSSI.

Devido a *fadding* e outros fenómenos, o sinal observado de um dispositivo móvel numa localização particular não é estático e parece variar ao longo do tempo. Como resultado, *software* de calibração

tipicamente armazena várias amostras da força do sinal para um dispositivo móvel durante o processo de amostragem. Dependendo da técnica, o atual vetor de elementos armazenados pode levar em conta esta variação através de uma ou mais abordagens "imaginativas". O vetor de localização assim torna-se um vetor de elementos da média de sinal.

2.2.4.2. Fase Operacional

Na fase operacional um grupo de estações providencia medidas da força do sinal relativamente ao dispositivo móvel rastreado e encaminha essa informação para um servidor de identificação de localização. O servidor de localização usa um complexo algoritmo de posicionamento e a base de dados do *training set* para estimar a localização do dispositivo móvel. O servidor responde com a estimativa da localização para a aplicação cliente que solicitou a informação do posicionamento.

Podem ser classificados em três grupos básicos:

- **Determinísticos**. Tentam encontrar a distância mínima estatística entre o vetor RSSI detetado e os vetores de localização das várias amostras de calibração. Isto pode, ou não, ser igual à mínima distância física entre a localização do atual dispositivo e a localização da fase da calibração. O ponto de amostra com a menor distância estatística e o vetor detetado de localização é geralmente tratado como a melhor localização estimada na base de dados de calibragem. Exemplos de algoritmos determinísticos são os baseados na computação de distâncias *Euclidiana*, *Manhattan* ou *Mahalanobis*.

  Pode também ser utilizado *dead reckoning* (DR) que funciona tendo em base a última localização estimada do dispositivo ao utilizar a velocidade ou aceleração. É um sistema de navegação altamente preciso, embora, tendo em conta que todas as localizações são determinadas pelo último passo, em caso de erro este aumenta em conformidade.
- **Probabilísticos**. Usam inferência probabilística para determinar a probabilidade de uma localização particular dada por um vetor de localizações que já foi detetado previamente. A própria base de dados de calibração é considerada como uma condição probabilística a *priori* pelo algoritmo para determinar a probabilidade de uma ocorrência de uma localização particular. Exemplos de tal abordagem incluem inferências de probabilidade *Bayesiana*.
- Outras técnicas vão além de abordagem probabilistas e determinísticas. Um exemplo de tal abordagem envolve assumir que o modelo é demasiado complexo para ser analisado matematicamente e necessita de funções não lineares discriminantes para classificação como por exemplo redes neuronais (Turgut, Aydin, & Sertbas, 2016).

De uma maneira similar a RSSI *lateration*, sistemas de localização em tempo real que usem esta técnica permite utilizar a infraestrutura existente. Isto pode geralmente tornar-se uma vantagem sobre abordagens que usem AoA, ToA e TDoA, dependendo na particularidade de cada implementação. Estas

são capazes de fornecer uma boa *performance* em ambiente *indoor*, onde é necessário um mínimo de três sensores estarem disponíveis no raio do dispositivo. Sendo que a precisão terá tendência para aumentar conforme o número de estações presentes.

Regra geral obtêm-se bom desempenho quando existem suficientes conjuntos de estações para permitir que localizações individuais sejam distinguidas pela aplicação de posicionamento. Contudo em grandes áreas, onde é possível os clientes deslocarem-se livremente sem barreiras físicas, os tempos de calibragem naturalmente aumentam. Deste modo, por vezes é segmentado o ambiente em áreas onde o cliente tem uma probabilidade de deslocação superior a áreas inferiores. A quantidade de calibração assim como recursos computacionais alocados a estas duas segmentações é ajustado pela aplicação de posicionamento em relação à probabilidade de o cliente estar ali.

Os mapas de radio, ou bases de dados de calibração utilizados por este tipo de posicionamento tendem a ser muito ajustados às áreas utilizadas na sua criação, com pouca oportunidade para reutilização já que a probabilidade é praticamente inexistente que duas áreas tenham conjuntos de calibrações semelhantes. Devido a isto, não é possível utilizar as mesmas calibrações para múltiplos pisos.

A precisão tipicamente está no seu auge imediatamente a seguir à sua calibração. Nesse momento, a informação encontra-se atualizada e reflete condições do ambiente. Conforme o tempo progride e alterações ocorrem que afetam a propagação, a precisão é esperada que degrade. Como as calibrações de *training* sets degradam-se ao longo do tempo, reverificações periódicas e possíveis recalibrações tornam-se o dia-dia se for necessário um elevado nível de precisão.

2.3. Tecnologias de Redes Sem Fios para Posicionamento *indoor*

2.3.1. Wi-Fi

2.3.1.1. Funcionamento

O *Wireless* começou como um modo de enviar programas de áudio através do ar. Rapidamente começou a ser apelidado de rádio e quando foi adicionado a imagem, surgiu a televisão. A palavra *wireless* deixou de estar na moda a partir de meados do século XX, mas nos últimos anos tem tido o seu renascimento. Por volta de 2007, aproximadamente metade dos utilizadores de internet expectavam utilizar algum tipo de acesso *wireless* – muitos deles em países em desenvolvimento onde as ligações tradicionais, baseadas em redes de telefones, não se encontravam disponíveis. (Woodford, Explain that Stuff, 2018)

A internet *wireless* é apenas uma maneira de utilizar ondas de rádio para enviar e receber dados de internet em vez de sons de rádio ou imagens de televisão. Mas ao contrário de radio e de TV, é tipicamente utilizada para enviar sinais em distâncias mais curtas utilizando transmissores mais fracos.

Para existir internet *wireless* é necessário a presença de um router ou um ponto de acesso (*Access Point* - AP) que se encontra ligada à infraestrutura do ISP (*internet server provider*). É possível utilizar um router para ligar vários computadores à internet. Noutras palavras, o router desempenha duas funções: cria uma rede *wireless* local de computadores (*Local Area Network* – LAN), conectado todos os equipamentos juntos e também providencia a estes equipamentos uma porta para a internet.

Uma rede de computadores é algo muito formal, quase como uma reunião de empresa, com pré acordados comportamentos. As máquinas na rede têm de se encontrar ligadas de uma maneira formal e comunicarem de maneira muito organizada. As regras que ditam estas "regras" de configuração e comunicação são baseadas no standard internacional denominado *Ethernet* (também conhecido como IEEE 802.3) (Rouse, Search Networking, 2018).

Um AP sem fios liga um computador (ou computadores) utilizando ondas de rádio em vez de cabos. Contem um transmissor e recetor de radio de baixo consumo com alcance a rondar os 90 metros (mediante o tipo de AP encontrado e dependendo de limitações físicas que possam existir). O AP pode enviar e receber dados de internet para cada computador na rede que também esteja equipado com acesso à internet. Em temos práticos o AP torna-se um ponto de acesso informal para a internet criando uma "nuvem" invisível de conectividade à sua volta, conhecido como *Hotspot*. Qualquer computador dentro desta "nuvem" pode-se ligar à rede, formando assim uma *wireless* LAN.

A Internet wireless está a melhorar a todo o momento, por isso novas formas de Wi-Fi continuam a evoluir. É possível ver equipamentos sem fios marcados como 802.11a, 802.11b, 802.11n: estes são todos compatíveis com variações da 802.11, embora sejam relativamente mais rápidas.

Um Wi-Fi *hotspot* é um sítio público onde se pode ligar um equipamento à internet através de uma rede *wireless*. Os *hotspots* que são encontrados em aeroportos, cafés, bibliotecas, universidades usam um ou mais AP *wireless* para criar uma rede *wireless* ao longo de uma grande área.

Embora tanto o Wi-Fi como o *Bluetooth* sejam tecnologias de redes sem fios as diferenças encontram-se no comprimento de onda que ocupam no espetro eletromagnético, e na sua forma de conectividade, onde o *Bluetooth* é conecta dois equipamentos relativamente próximos, já o Wi-Fi é um método de ligar dispositivos *wireless* à internet através de um ponto de acesso partilhado – o AP – que tipicamente realiza a ligação com cabo a uma linha de telefone.

Wi-Fi não é a única maneira de aceder internet sem cabos. Nos finais dos anos 90 e inícios 2000, alguns telemóveis tinham browsers muito rudimentares que podiam mostrar versões simplificadas das páginas web chamada WAP (tecnicamente conhecido como *Wireles Application Protocol*). WAP era muito lento a arrancar e tornou-se obsoleto por redes de telemóveis mais rápidas e *smartphones* (Brislen, 2000).

Mais importante é o facto do Wi-Fi ser responsável por mais de 60% do tráfego de internet a nível mundial. O que levanta questões de segurança e como seria de esperar este tem um conjunto de atributos de segurança. Para aceder à rede um utilizador tem de ter uma password para o WPA2, também

conhecido como Wi-Fi *protected access*. Isto é onde é inserido a password para aceder a rede. Existe outra caraterística de segurança chamado de *Advanced Ecncryption Standard* (conhecido como AES) que foi desenvolvido pelo governo dos Estados Unidos da América para manter dados seguros enquanto transmite de um dispositivo para outro. "*Cada instância de informação de cada comunicação que navega pelo* Wi-Fi *é exclusiva e encriptada e apenas as duas partes envolvidas a conseguem entender*"

Outra caraterística importante do Wi-Fi é a sua retrocompatibilidade. Isto é como todos os computadores antigos se conseguem ligar a novos routers, que por consequência são mais velozes e eficiente. (Pullen, 2015)

2.3.1.2. Posicionamento & Casos de uso

O maior benefício de Wi-Fi é que a tecnologia já se encontra disponível e facilmente aplicável na maior parte do mundo. Pode ser utilizada com hardware presente e é uma solução relativamente barata.

Posicionamento *indoor* com Wi-Fi faz ainda mais sentido quando o sinal de GPS é inexistente. Por exemplo, em infraestruturas complexas como centros comerciais, museus, estações de comboios, aeroportos, hospitais, escritórios e edifícios industriais. Uma aplicação de posicionamento aumenta o serviço ao cliente e permite a análise de padrões de visitantes. Permite também que consumidores interajam com informação relevante no seu redor – como por exemplo obter cupões de desconto ou guiá-los num mapa.

As aplicabilidades de posicionamento com uso de Wi-Fi são vastas, mas é principalmente indicado para retalho, feiras, aeroportos, estações de comboio, museus e hospitais. Tem como grande vantagem a reutilização de infraestrutura existentes com pontos de acesso de Wi-Fi existentes. A grande desvantagem é que aplicação de posicionamento exclusivos de Wi-Fi automaticamente excluem equipamentos *Apple*, já que estes não suportam posicionamento via Wi-Fi principalmente por restrições de fabricante (Pichler, 2019).

Para marcas e representantes, podem ver (em muitos casos pela primeira vez) uma imagem mais clara de como as pessoas se deslocam num espaço físico, até mesmo criar um mapa de pontos. Existem áreas congestionadas, ou áreas onde ninguém vai? Quanto tempo demoram as filas de caixa? De que modo o espaço pode ser otimizado? (Verma, 2017)

Por vezes é possível ver uma aplicação de GPS que solicita o acesso à rede Wi-Fi. Possivelmente é estranho assumir que Wi-Fi tem algo relacionado com "posicionamento de GPS", mas de facto as duas podem-se interligar para uma maior precisão de localização.

Este mecanismo híbrido é particularmente útil em áreas urbanas onde existe redes de Wi-Fi a transmitir. Contudo, os benefícios são ainda maiores quando se equaciona cenários onde o GPS não funciona, como debaixo do solo, em edifícios ou centros comercias, onde o sinal deste é demasiado fraco, intermitente ou simplesmente inexistente.

Algo para se reter é que *Wi-Fi Position System* (WPS), como seria de esperar, não funciona quando fora do alcance de sinal Wi-Fi.

Para solucionar este problema os dispositivos que tem tanto GPS e Wi-Fi podem ser utilizados para enviar informação de uma rede Wi-Fi para um servidor ou sistema. Este sistema passa assim a ter conhecimento de que redes Wi-Fi se encontram naquela localização. Em termos práticos o dispositivo apenas envia o ponto de acesso *Basic Service Set Identifier* (BSSID ou *Mac address*) com a localização determinada pelo GPS.

Quando um dispositivo móvel, como um smartphone, utiliza o GPS é para determinar a localização, este também verifica redes próximas que possam ajudar a identificar a rede. Quando a localização e redes próximas são encontradas, a informação é gravada em sistemas de posicionamento, como *Google Maps*. A próxima vez que alguém estiver perto de uma destas redes, mas que não tenham um bom sinal de GPS, podem ser utilizados estes serviços para determinar uma localização aproximada já que a localização da rede é conhecida no contexto GPS.

Um exemplo prático é o de um dispositivo móvel com acesso de GPS em pleno, e tendo o Wi-Fi ligado a uma loja de conveniências facilmente é encontrada a localização da loja já que o GPS está a funcionar, por isso a localização e alguma informação de redes Wi-Fi são enviadas a empresas especificas como por ex. *Microsoft*, *Apple* ou *Google*. Mais tarde, um dipositivo móvel entra nessa mesma loja com o Wi-Fi ligado, mas sem o sinal de GPS ligado. Contudo, como a localização da rede é conhecida (já que um dipositivo enviou esta informação previamente), a localização pode facilmente ser identificada.

Esta informação está constantemente a ser recolhida por empresas como *Microsoft*, *Apple* e *Google*, o que permite providenciar serviços de localização mais precisos. Algo a assinalar é que esta informação recolhida é de domínio publico; não são necessárias as *passwords* do Wi-Fi para funcionar.

Obter a localização de cliente de dispositivos móveis através deste modo faz parte de qualquer contrato de fornecedores de serviços de comunicação ou dispositivos móveis, embora a maioria dos dispositivos permitam que o utilizador do dispositivo desligue os serviços de posicionamento. Similarmente, se não se pretender que uma rede *wireless* não seja utilizada deste modo é possível excluí-la. Por exemplo a *Google* deixa de fazer a recolha destes tipos de dados se na rede de Wi-Fi a mesma estiver identificado com "_nomap" no final do seu nome – "aMinhaRede_nomap" (Zahradnik, 2018).

A precisão depende de um conjunto de fatores, como por exemplo o número de redes disponíveis (num cenário urbano atual é fácil estar-se a um alcance de diversas redes), reflexos, e o mais importante, barreiras físicas, tal como paredes, tetos e até mesmo pessoas. A precisão de Wi-Fi utilizado para posicionamento *indoor* pode variar até 15 metros (Srivastava, 2018).

Uma das vantagens é a possibilidade de utilizar *sensor fusion* – utilização de outros sensores do *smartphone* – de modo a aumentar a precisão, mas que implica a instalação de uma aplicação no

dispositivo. Nesta situação a calibração do posicionamento funciona com uma medida de referência na aplicação, onde a força do sinal de Wi-Fi é critica para determinar a posição do dispositivo.

Caso se prefira utilizar uma arquitetura sem o uso de aplicação, ou seja, seria a infraestrutura a reconhecer a posição dos dispositivos que tenham Wi-Fi a precisão será inferior, já que não é permitido a utilização de *sensor fusion*.

Os casos de uso, do ponto de vista técnico, são diversos, já que cada fornecedor de serviços de localização tenta uma abordagem diferente, mas de uma maneira geral a implementação de um sistema de posicionamento baseado em Wi-Fi é relativamente simples, já que os pontos de acesso já se encontram presentes na infraestrutura do edifício. Mediante a abordagem e arquitetura definidas o utilizador não tem necessariamente de se conectar a estas redes, basta ter o Wi-Fi ligado. A força do sinal de Wi-Fi (indicação da força de sinal recebido – RSSI) e a MAC *address* (*media access control*) são suficientes. Conforme o tipo de arquitetura escolhida poderá existir a necessidade de ter uma aplicação instalada no equipamento que calcula a posição baseado nos RSSI captados no alcance do dispositivo.

Contudo, a abordagem WPS normalmente precisa de levantamento de dados, como a localização dos pontos de acesso, parâmetros de propagação, ou mapas de RSSI. Estes acabam por ser altamente repetitivos e consumidores de tempo já que são dados que variam conforme o tempo passa. Remoção ou adição de AP, percas de sinal de um ou mais APs, alterações físicas que tornam os dados recolhidos na fase de calibração irrelevantes, entre outros.

Deste modo, Y.Zhuang *et al*. (2015) propõem uma solução que através de *crowd sourcing* este processo de calibração, tanto o inicial, como o realizado à posteriori, é inexistente. O objetivo é que os dados obtidos através de *crowd sourcing* atualizem os dados dos pontos de acesso presentes na base de dados mantendo-os atualizados. Este processo é automática e ocorre no *background* e sem quaisquer restrições por parte do utilizador, tornando-o totalmente autónomo. A nível de posicionamento a abordagem também é simples, primeiro os valores RSSI são convertidos em distâncias usando o modelo de propagação através dos parâmetros de propagação obtidos da base de dados, com isto a posição do utilizador é estimada através da raiz quadrada não linear destes dados. Os resultados obtidos foram que a média de erros da posição era inferior a 6 metros e que à medida que os trajetos eram mais percorridos esta precisão ia aumentado. Também identificaram que mediante o dispositivo utilizado o sistema poderia ficar com um desempenho de posicionamento degradada e que à medida que eram utilizados mais pontos de acesso a precisão de posicionamento do dispositivo aumentava.

Outras soluções, como a proposta por Carrera *et al.* (2018) utiliza *sensor fusion* para obter a localização do dispositivo. Criam um sistema, que corre no dispositivo, que explora *inertial measuremente units* (IMU) obtidos dos vários sensores embutidos em *smartphones* tais como o acelerómetro giroscópio, sensores de magnetismo, etc. já que estes registam constantemente a atividade do utilizador do

dispositivo e podem ser utilizados para descobrir os movimentos relativos deste. Acabam por fazer *pedestrian dead reckoning* (PDR) com os valores obtidos do IMU. Sendo que PDR é suscetível a erros, combinam os valores rádios obtidos do RSSI e com a informação do piso obtido num *training set* criando um filtro que suaviza os erros produzidos por interferências magnéticas e ruídos dos sensores utilizados em *smartphones*. De modo a reduzir a complexidade computacional representam o ambiente físico num modelo baseado em grafos.

Apresentam um método de reamostragem dupla que é capaz de mitigar os erros causados pela frequência de amostragem baixa de Wi-Fi dos dipositivos. Sobre isto desenham um mecanismo de recuperação de falha de localização, com uma abordagem de *machine learning*, tornando o sistema mais resiliente.

Acabam por obter resultados em que a média de erro da precisão é de 1.15 metros, sendo que 90% dos resultados tem valores de 1.8 metros com um desvio padrão de 0.8 metros. Sobre estes valores é também importante notar que em caso de perca de posicionamento o sistema demora cerca 3.4 segundos em média a recuperar a sua posição.

2.3.2. *Bluetooth* & *Beacons*

2.3.2.1. Funcionamento

Bluetooth, similarmente a Wi-Fi, é uma tecnologia rádio, embora seja maioritariamente desenhada para comunicação de curto alcance (~10 m). Tipicamente pode ser utilizada para efetuar a transferência de dados entre dispositivos (como fotografias de um telemóvel para outro), ligar periféricos, como o rato *wireless* a um portátil, ligar uns auscultadores como um dispositivo de mãos livres a um veículo, entre outros. Os dispositivos eletrónicos que funcionam deste modo tem antenas embutidas (emissores e recetores) que permitem a comunicação com outros dispositivos com *Bluetooth*.

O *Bluetooth* pode ter as mesmas aplicações de LBS que o Wi-Fi, embora a sua forma de aplicabilidade seja diferente devido ao seu alcance, ou seja é impossível ter uma rede que consiga cobrir uma área considerável e ter alguma precisão de posicionamento. Assim sendo é necessário ter *hardware* que ajude o posicionamento com recurso a *Bluetooth*, é neste contexto que surgem os *beacons*.

Um *Beacon* é um pequeno transmissor rádio que utiliza *Bluetooth* alimentado por pequenas baterias. Estes pequenos dispositivos transmitem constantemente sinais *Bluetooth Low Energy* (BLE). Isto permite que dispositivos, como *smartphones*, captem estes sinais. Funciona quase como um farol onde um dispositivo com *Bluetooth*, "vê" um *Beacon* quando este se encontra no seu alcance (Kontakt, 2019).

2.3.2.2. Posicionamento & Casos de uso

No caso de *Bluetooth*, no contexto de posicionamento, é necessário ter em conta que, embora estes possuam uma antena no seu interior, esta não é direcional. É virtualmente impossível saber a localização

de um dispositivo utilizando apenas um *Beacon* já que é apenas possível estimar a distância em relação ao *Beacon*, não de onde este sinal origina. Assim é necessário ter pelo menos 3 *Beacons* em alcance para comparar os valores de RSSI de cada um e poder ser realizada *trilateration*. Tendo em conta Wang *et al.* (2013) pode ser obtido de três maneiras diferentes.

- Método de estimativa de quadrado mínimo (*Least squares estimation* – LSE). Ao minimizar os desvios padrões das somas dos quadrados mínimos pode ser encontrada a posição do utilizador.
- Método de três fronteiras. Ao definir a equação que representa a distância entre o utilizador e os *Beacons*, pode ser encontrada a posição do utilizador.
- Método centroide. Um polígono é primeiro definido de acordo com os vértices, sendo que estes são definidos pela interseção de pontos entre os arcos de distância. O centro deste polígono é assumido como a posição do utilizador.

O outro método onde os cálculos de certa forma são mais "rudimentares" passa pela utilização de *fingerprinting*. Consiste em duas fases: calibração e operacional. Na primeira é feito uma recolha de RSSIs em pontos de controlo de modo a criar um mapa de referência. Na segunda fase é recolhido os RSSIs numa posição desconhecida sendo que estes são comparados com o mapa de referência de modo a resolver a posição do dispositivo. Contudo sendo que os dados tem de ser recolhidos com posições diferentes da fase operacional o processamento é elevado.

Conforme demonstrado por *Zeng, et al.* (2018) pode ser criado um modelo para estimar as posições dos *Beacons* e gerar um mapa de referência de dados (*reference fingerprinting* map – RFM) através de grafos sem necessidade de qualquer tipo de hardware específico para a fase de calibragem e com desconhecimento prévio da localização dos Beacons ou de pontos de controlo. Consiste num utilizador equipado com um dispositivo (*smartphone*) com o *Bluetooth* ativo deslocar-se numa região recolhendo dados de inércia e dados da força do sinal BLE (calculado através de RSSI). Os dados da inércia são processados através de um método de PDR para gerar limitações em várias localizações. Adicionalmente as leituras BLE recolhidas para o *fingerprinting* são utilizadas para gerar limitações entre localizações (com *fingerprinting* semelhantes) e os RSSIs são utilizados para gerar limitações entre as localizações e a localizações dos *Beacons*. Estas limitações são depois utilizadas para calcular uma função de custo com uma estrutura ao quadrado. Em adição o mapa de referência de dados pode ser estimado através das estimativas das localizações.

Estes acabam por ter resultados aceitáveis para muitas aplicações indoor, 1.27 metros em cenários com vários Beacons e 2.26 metros em cenários com poucos Beacons.

Tendo em conta a questão da precisão, esta pode ser afetada mediante a forma como a solução é implementada (Infsoft, 2019).

2.3.3. Luz – *Visible light communication* (VLC)

2.3.3.1. Funcionamento

Conforme o nome indica é uma tecnologia que permite comunicar (transferir informação) através de Luz. Este utiliza a luz visível, que ocupa o espetro entre dos 380 nm aos 750 nm com uma frequência entre 430 THz a 790 THz. O recetor de VLC apenas recebe sinais se estiver presente no mesmo "quarto" da fonte do emissor, assim recetores fora deste espaço não serão capazes de receber sinal, tornando assim este sistema mais seguro que sistemas baseados em sistemas de radio (*radio frequency* – RF), já que os sinais não podem ser intercetados por dispositivos que não tenham acesso ao mesmo espaço.

Uma das vantagens é a reutilização da fonte de luz, onde pode ser utilizada tanto como iluminação como comunicação, poupando assim a energia gasta em sistemas RF.

As potenciais aplicações de VLC incluem Li-Fi, comunicação veículo-veículo, *robots* em hospitais, comunicação debaixo de água, informação de outdoors, ou seja, basicamente tudo o que emite luz. O Li-Fi utiliza luz visível para comunicar podendo atingir 10 Gb/s. No caso de comunicação veículo-veículo pode ser utilizado para informar alterações de faixa de rodagem, travagens, violação de regras de trânsito evitando acidentes. Permite uma comunicação sem latência devido à sua elevada largura de banda. Também é aplicável em cenários onde seja sensível a ondas eletromagnéticas, como aviões e hospitais, onde as ondas rádio interferem com as ondas de outros dispositivos (Khan, 2017).

Sistemas VLC aproveitam as vantagens dos LEDs, que podem ser emitidos a grandes velocidades sem afetar a luminosidade, pelo menos de uma forma percetível pelo olho humano. Embora o facto de ser necessário uma linha de visão constante para funcionar possa ser encarado como uma vantagem de segurança, esta também é uma limitação da tecnologia conjuntamente com interferências causadas por outras luzes.

2.3.3.2. Posicionamento & Casos de uso

Pode ser utilizado como uma tecnologia de posicionamento, principalmente para ambientes *indoor*. LEDs especiais em lâmpadas florescentes cintilam indiscriminadamente e que pode ser detetado pela câmara de um dispositivo – *smartphone* – ou um sensor de fotografia separados, como por exemplo acoplado a um cesto de compras. Isto permite por exemplo navegação *indoor* (via app).

Funciona da seguinte maneira: cada lâmpada tem o seu próprio ID emitido através de um pulso de luz, este é "capturado" por um dispositivo que esteja no seu alcance. A app depois acede um mapa, em que cada lâmpada, com o seu id, está identificada e georreferenciada, conseguido inferir a posição do utilizador. Este tipo de sistemas funcionam ao ter controlo dos pulsos emitidos pelos LEDs de modo a que tenham um certo padrão. Este padrão não é detetado pelo olho humano (funciona na casa das centenas de *hertz*), mas pode ser detetado por uma câmara de um *smartphone* ou um *tablet*. Usando estes

dados o dispositivo comunica com uma *app* e calcula um conjunto de cálculos de modo a entender a sua localização dentro da estrutura. Como estes sistemas não dependem de outras tecnologias, como Wi-Fi ou *Bluetooth*, não existem problemas de comunicação e tendo em conta que todos estes cálculos ocorrem no dipositivo a velocidade é rápida. Um caso prático de uma implementação com esta tecnologia é o da *ByteLight's* © (Cangeloso, 2013).

Outro modo de realizar o posicionamento, que não necessita de cálculos complexos como o mencionado em cima, é proposto por Kim, *et al.* (2017). O processo efetua apenas cálculos simples e processamento de imagem e similarmente os cálculos são processados do lado do dispositivo. Consiste na contagem de *stripes* captadas pela camara em cada pulso o que permite a inferência da posição geográfica das LEDs.

Capítulo 3 - Sistema de Navegação Orientado a Wi-Fi

Este capítulo tem como objetivo descrever a arquitetura do Sistema de Navegação Orientado a Wi-Fi – WINS (Wi-Fi *Indoor Navigation System*). Irá focar-se na arquitetura geral do projeto e nas diversas componentes de modo a ser possível criar uma aplicação que permita a um utilizador saber a localização de uma sala e a partir da sua posição atual obter um caminho até à sala escolhida.

Neste capítulo será também abordado o Sistema Operativo (SO) *Android*, na medida que o desenvolvimento da aplicação foi realizada para este SO.

3.1. Arquitetura WINS

A Figura 3.1 representa a arquitetura geral do sistema desenvolvido. Este é composto por 2 componentes principais: componente Servidor (WINS) e a componente Cliente (Client).

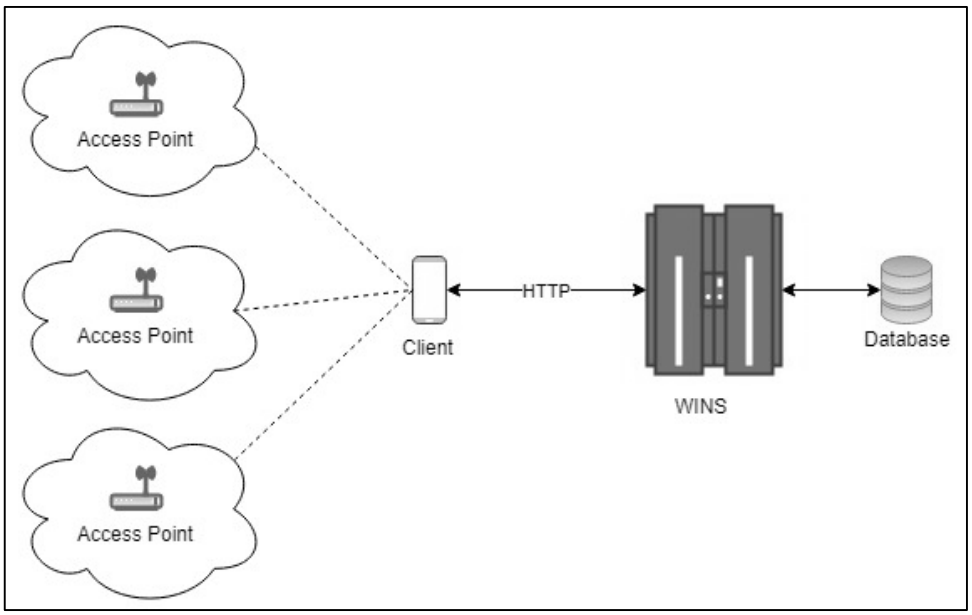


Figura 3.1: Arquitetura aplicacional do sistema WINS

Embora seja possível desenvolver um LBS apenas recorrendo a uma componente Cliente tal não foi realizado já que um dos objetivos do WINS é deixar uma estrutura que se possa acrescentar funcionalidades. Deste modo foi escolhido uma arquitetura onde um Cliente comunica com um Servidor e esta componente terá de devolver a posição e o caminho ao utilizador da aplicação Cliente.

Deste modo o foco principal da navegação e posicionamento está no componente Servidor. A única responsabilidade da componente cliente é disponibilizar uma app, de modo a que o utilizador possa solicitar direções para os pontos de interesse disponibilizados, obtendo rotas o que permitirá a

navegação. Em *background* a aplicação deverá obter os dados do seu redor de modo a poder comunicar com servidor através de uma API.

O Servidor terá 3 responsabilidades principais, sendo que todas são disponibilizadas via a sua API. Providenciar dados de interesse ao utilizador, calcular a posição do utilizador e calcular rotas. O Servidor comunica por *Hyper Text Transfer Protocol* (HTTP) com o cliente, tendo uma base de dados que utilizará para armazenar dados.

Um dos problemas desta arquitetura é que não existirá um sistema de *fallback*. Ou seja, se o Servidor se encontrar desligado, automaticamente o Cliente deixa de funcionar. Embora seja um cenário pouco desejado, é deixada esta implementação para trabalhos futuros.

3.1.1. Servidor – *Back-end*

Esta componente acaba por ser a mais importante do WINS, uma vez quer é no servidor que se realizam todas os cálculos para estimar a posição do utilizador através do seu dispositivo móvel.

O Servidor terá duas principais responsabilidades. Em primeiro, obter a posição do utilizador com o dispositivo, em segundo providenciar um caminho a ser percorrido pelo utilizador do dispositivo

Será auxiliado por uma base de dados sobre um SGBD que permitirá armazenar dados, dando oportunidade de otimizar os algoritmos de posicionamento recorrendo a processos de *machine learning*, embora sejam excluídos deste projeto, já que se encontra fora de âmbito.

A explicação detalhada desta componente será efetuada na secção 4.1. O que é importante reter é que será um servidor que poderá ser alcançado via o protocolo HTTP e disponibilizará uma API REST. (UCI, 2019)

3.1.2. Cliente - Aplicação

Identificar e escolher qual a tecnologia mais indicada para tecnologia de posicionamento é um decisor critico para o fator de sucesso de projetos de navegação *indoor*. Sendo que o objetivo é implementar um sistema de posicionamento que recorra a Wi-Fi, automaticamente o SO IOS é excluído, conforme explicado no capítulo 2.3.1.1.

Desse modo é escolhido o SO Android para a implementação desta aplicação. Esta escolha não limita futuros trabalhos, que pretendam adicionar componentes dependentes de outros sensores, tal como *Beacons*, a esta solução, já que este SO também permite a interação com *Beacons*.

A responsabilidade da componente cliente é de recolha e apresentação dos dados num formato legível para o utilizador num mapa.

Em primeiro lugar irá recolher dados do ambiente em que o dispositivo se encontra, comunicando-os à componente Servidor. O Servidor por sua vez irá devolver a posição do utilizador. Esta posição, para

todos os efeitos não será legível para o utilizador. Desse modo o cliente terá de transformar estes dados, para algo percetível pelo utilizador. Sendo que os dados rececionados do servidor são coordenadas, irá mostrar estas coordenadas no modo mais fácil para o utilizador os interpretar. A posição do utilizador será apresentada num mapa onde em conjugação com uma rota, o utilizador pode chegar ao destino solicitado.

3.2. Sistema Operativo *Android*

Esta secção abordará o *lifecycle* de uma *Activity* do SO *Android* e não será exaustiva, de modo a fundamentar algumas decisões tomadas na implementação da componente cliente.

As *Activities* são uma componente de o SO *Android* que permite que o utilizador esteja a interagir, despoletando uma ação, como tirar uma fotografia, enviar um email ou visualizar um mapa. Enquanto *Activities* podem ser mostradas ao utilizador como um menu que ocupa o ecrã inteiro, podem também ser utilizadas de outras maneiras. Como por exemplo, componentes flutuantes no ecrã, como um botão de ações, ou componentes compostas por outras *Activities*, como um painel lateral de ações que por sua vez podem chamar outras *Activities*.

Deste modo podemos dizer que um utilizador interage com uma aplicação através de *Activities* que por sua vez comunicam com outras componentes permitindo transformar dados para uma tradução visual, interpretável pelo utilizador, como uma rota, que apenas é um conjunto de pontos distribuídos no espaço físico sobre um mapa em que o utilizador navega sobre ela.

As *Activities* no sistema são tratadas como camadas. Quando uma nova atividade é iniciada, é geralmente colocada no topo e torna-se a *Activity* em execução – as *Activities* anteriores continuam sempre "debaixo" esta e não serão acedíveis enquanto a *Activity* corrente não para a sua execução. Pode existir uma ou várias camadas de *Activities* visíveis no ecrã.

Estas tem essencialmente quatro estados (Google, 2019):

- *Active* ou *Running* – trata-se de uma *Activity* que se encontra acedível e manuseável pelo utilizador.
- *Visible* – quando perdeu foco do utilizador. Esta está completamente "viva" (no sentido que a aplicação não a irá encerrar caso precise de recursos).
- *Stopped* ou *Hidden* – encontra-se totalmente submersa por outra *Activity*, mas mantém o estado e dados, contudo não é visível pelo utilizador, pelo que em caso de necessidade de memória poderá ser encerrada pelo sistema.
- *Destroyed* – simplesmente não existe para o sistema. Pode ter sido encerrada ou simplesmente morta pelo sistema. Quando apresentada novamente ao utilizador deve ser completamente iniciada e restaurada ao seu estado prévio.

A Figura 3.2 descreve o ciclo de vida de uma *Activity*.

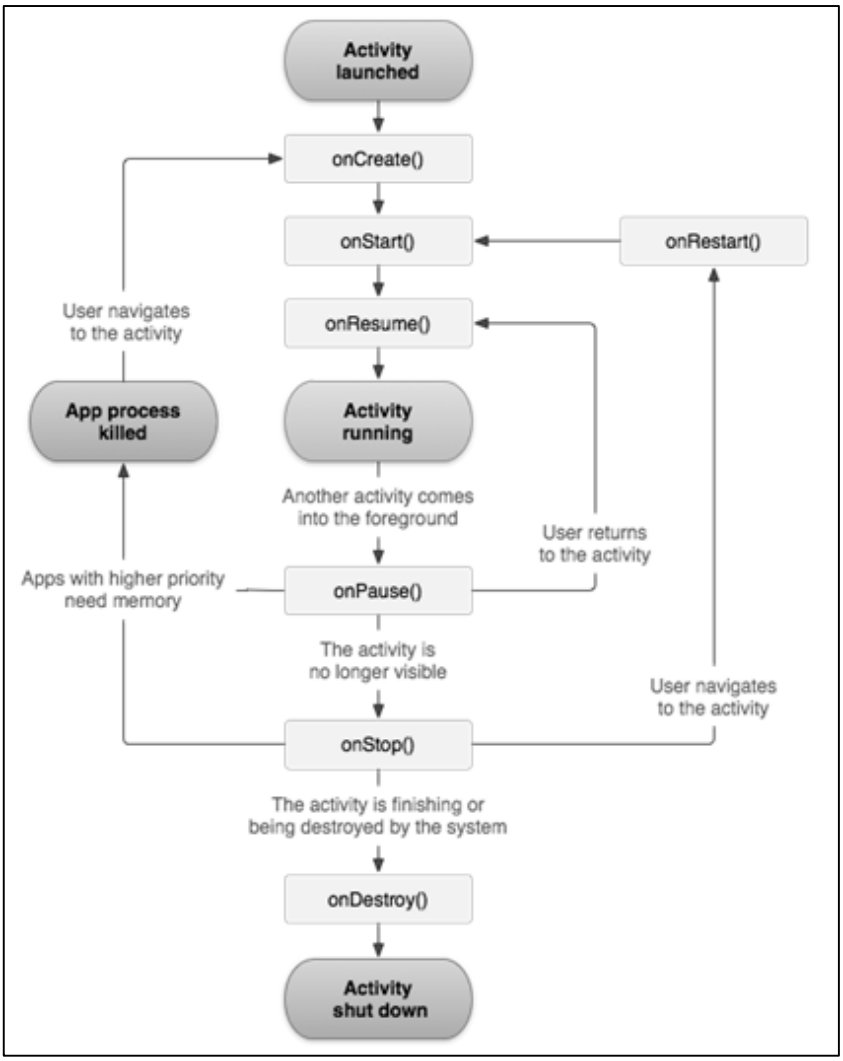


Figura 3.2: Ciclo de vida de uma Activity. (Google, 2019)

O ciclo de vida inteiro de uma *Activity* ocorre entre a primeira chamada *onCreate()* até a *onDestroy()*. Uma Activity tem que definir o seu estado no *onCreate()* e libertar todos os seus recursos em *onDestroy()*. Por exemplo se tiver uma *thread* para realizar o *download* de dados de um sistema externo pode criar esta *thread* no *onCreate()* e pará-la no *onDestroy()*.

O ciclo de vida "visível" de uma *Activity* ocorre entre o *onStart()* até ao *onStop()*. Durante este tempo um utilizador pode ver a *Activity* no ecrã, embora não possa interagir com ela. Pode ser utilizado para alterar componentes de *User Interface* (UI) ao qual o utilizador terá acesso depois. Estes métodos podem ser chamados múltiplas vezes à medida que a *Activity* fica visível e escondida do utilizador.

A mais importante ocorre quando uma Activity se encontra ativa entra as chamadas *onResume*() até um correspondente *onPause()*. Durante este período a Activity encontra-se visível, ativa e passiva ao nível de interação com o utilizador.

É sobre estes métodos que é possível desenvolver a lógica necessária de modo a que a aplicação funcione, interligando as diversas camadas de *Activities*, permitindo assim ao utilizador uma boa experiência e do lado aplicacional uma boa gestão de memória não causando impactos ao utilizador enquanto este utiliza a aplicação.

Capítulo 4 - Implementação WINS

Este capítulo tem como objetivo descrever a implementação do WINS, assim como demostrar os resultados obtidos. Irá focar-se nas diversas tecnologias utilizadas e nas decisões tomadas aquando a sua implementação.

O WINS está dividido em duas componentes: componente Servidor e componente Cliente. Para cada componente existirá uma secção correspondente onde é detalhada a implementação. Adicionalmente, noutra secção para este efeito, são também demostrados os resultados obtidos. Sendo que no contexto deste projeto existe uma componente de desenvolvimento *Android*, a linguagem de programação escolhida para o seu desenvolvimento foi o *Java* (Oracle, 2019).

4.1. Servidor – *Back-end*

O Servidor é a peça mais importante do WINS. Aqui irá ser desenvolvido toda a lógica de negócio inerente de um sistema de posicionamento com uso de Wi-Fi. Na Figura 3.1, apresentada previamente, é possível observar o modo como a componente do servidor foi conceptualizada e implementada.

É importante referir que não foi considerada nenhuma componente de segurança. Sendo que o objetivo deste projeto tem como foco disponibilizar um conjunto de interfaces que depois possam ser utilizadas por aplicações clientes, esta é uma componente que intrinsecamente é necessário contemplar. Tal decisão foi tomada devido ao facto de ser uma componente que só por si daria um outro projeto, o que acabaria por tomar espaço desta, retirando o contexto geográfico e posicional.

4.1.1. Pressupostos

Existem um conjunto de pressupostos, ou requisitos, que tem de ser endereçados. Podem ser repartidos em dois principais: interoperabilidade com outros sistemas e posicionamento com uso de Wi-Fi.

- No âmbito da interoperabilidade com outros sistemas pode ser mencionado sucintamente que se encontra no contexto de normas. Ou seja, de que modo pode ser esperado que este sistema seja acedido e utilizado, em particular os protocolos de comunicação disponibilizados e qual o formato dos dados transmitidos sobre estes protocolos.
- No âmbito do Wi-Fi é necessário identificar as necessidades que um sistema como este necessita para funcionar com o objetivo de ter uma precisão aceitável tendo em conta o objetivo do sistema.

4.1.1.1. Interoperabilidade

De modo a que as interfaces disponibilizadas pelo WINS possam ser acedidas via *Web* foi escolhido o protocolo HTTP. Este é um protocolo aplicacional para sistemas distribuídos, colaborativos e informacionais. Tem sido utilizado pela *Web* desde 1990 e é utilizado como um protocolo genérico de troca de informação entre *user agentes*, *proxies/gateways* e outros sistemas na internet, incluído os que são suportados por *SMTP, NNTP, FTP, Gopher e WAIS*. Encontra-se assente em *Request for Comments* (RFCs) 7230 – Message Syntax and Routing, 7231 – Semantics and Content, 7232 – Conditional Requests, 7233 – Range Requests, 7234 Caching, 7235 – Authentication.

Sobre este protocolo é desenvolvido uma estrutura REST – *Representational State Transfer*. A escolha desta norma é devido ao facto de esta definir um conjunto de normas, embora concetuais, que garante a interoperabilidade com outros sistemas que também conheçam e apliquem esta norma corretamente. Deste modo, qualquer aplicação que se pretenda integrar com o sistema apenas terá de conhecer ditas convenções de antemão, tornando a sua integração muito mais flexível.

O REST é um estilo arquitetural de como um sistema de Serviços deve ser desenhado. Embora não exista um RFC que denote todo o detalhe de implementação destes tipos de sistemas estes estão intrinsecamente ligados a HTTP.

Enquanto o HTTP é puramente um protocolo de comunicação em que, de maneira muito sucinta, podemos dizer que é um envelope com um conjunto de caraterísticas e ações, o REST define de que modo este envelope deve ser entregue e interpretado.

Observando os exemplos em baixo, onde temos um exemplo sem REST e outro com REST pode ser verificado que se refletem na mesma ação:

GET	get_room/8.1.64
POST	delete_room/8.1.64

Neste pedido HTTP obteríamos um dado e no seguinte pedido, apagaríamos esse mesmo dado. Num paradigma REST teríamos os seguintes pedidos:

GET	rooms/:id
DELETE	rooms/:id

Ou seja, ambos acabariam por desencadear a mesma operação, daí o REST ser totalmente uma conceptualização de como deve ser implementado uma API. Um dos aspetos mais importantes do REST recai na sua terceira letra, *Statelessness*. Isto significa que o servidor não necessita de ter qualquer tipo de conhecimento do cliente que o acede e vice-versa. Deste modo, tanto o servidor como o cliente podem interpretar a mensagem sem terem a necessidade de algum estado prévio. Este tipo de "estado" é forçado através do uso de recursos em vez de comandos (como mostrado em cima). Estas regras ajudam aplicações REST a serem fiáveis, rápidas e escaláveis como componentes que podem ser geridas,

atualizadas e reutilizadas sem afetarem o sistema como um todo, dando uma elevada separação de responsabilidades no contexto aplicacional (Codecademy, 2019).

Sendo que o objetivo principal da componente do Servidor do WINS é ser o mais genérico possível fica apenas a faltar o detalhe do formato dos dados. Sendo que o foco são os dados geográficos foi escolhido o formato *GeoJSON*, já que suporta um conjunto de tipos geométricos como *Point, LineString, Polygon, Multipoint, MultiLineString* e *MultiPolygon* (RFC-7946, 2019).

4.1.1.2. Posicionamento Wi-Fi

Relativamente ao contexto do posicionamento é necessário identificar de que modo este será realizado. Sendo que o objetivo deste projeto não passa pela alta precisão do posicionamento, embora seja um fator importante, não será o objetivo obter o máximo de precisão possível com utilização do Wi-Fi. O que será importante fornecer é a identificação do utilizador no espaço físico e para onde este deseja navegar.

Assim, o utilizador da aplicação terá de providenciar o destino pretendido, onde a aplicação traduz este destino para coordenadas geográficas. Com base na sua posição, obtida através do Wi-Fi, o sistema calculará um caminho em qual o utilizador se desloca. À medida que o utilizador se desloca no espaço físico a aplicação comunicará com o sistema, com um conjunto de parâmetros, onde o sistema identificará a posição do utilizador.

4.1.2. Modelo de Dados

Nesta secção será detalhado o modelo de dados utilizado pelo WINS para disponibilizar um caminho ao utilizador. Na primeira seção será retratado o modelo de dados, ou seja, que tipo de entidades irão existir na base de dados. Na segunda seção será abordado o modo como é efetuado o carregamento dos dados.

De notar que foi escolhido o Sistema de Gestão de Base de Dados *MySQL* © (MySQL, 2019) por ser uma ferramenta *OpenSource*.

4.1.2.1. Entidades

Embora o modelo esteja representado no mesmo SGBD, este pode ser repartido em duas componentes de modo a que sua leitura seja menos complexa. Em primeiro, encontram-se as entidades que modelam o espaço físico, como edifícios, pisos e salas. No segundo, encontram-se os dados que representam a navegação, quase como algo abstrato que não é possível tocar no espaço, como os pontos de controlo, e RSSI de pontos de acesso.

O que é comum a estas duas famílias é que ambas têm sempre uma componente geográfica e deste modo estão sempre associadas a entidades que representam algo no espaço físico, correspondentemente *polygon, polylines, points.*

Observando a Figura 4.1 são apresentadas as entidades que representam o espaço físico. Onde neste contexto o espaço físico corresponde a figuras geométricas georreferenciadas.

Mediante o tipo de entidade varia a sua estrutura. Todas elas tem um contexto geográfico, onde partem sempre de um Ponto (*Point*) que respetivamente tem os seus dados geográficos através da latitude, longitude e altitude. Do Ponto pode depois surgir uma entidade Linha (*Polyline*) e da Linha pode surgir uma entidade Polígono (*Polygon*) composta por várias entidades do tipo Linha. Assim estas três entidades, são as principais no contexto do WINS, já que sem elas, não existem dados com uma vertente geográfica.

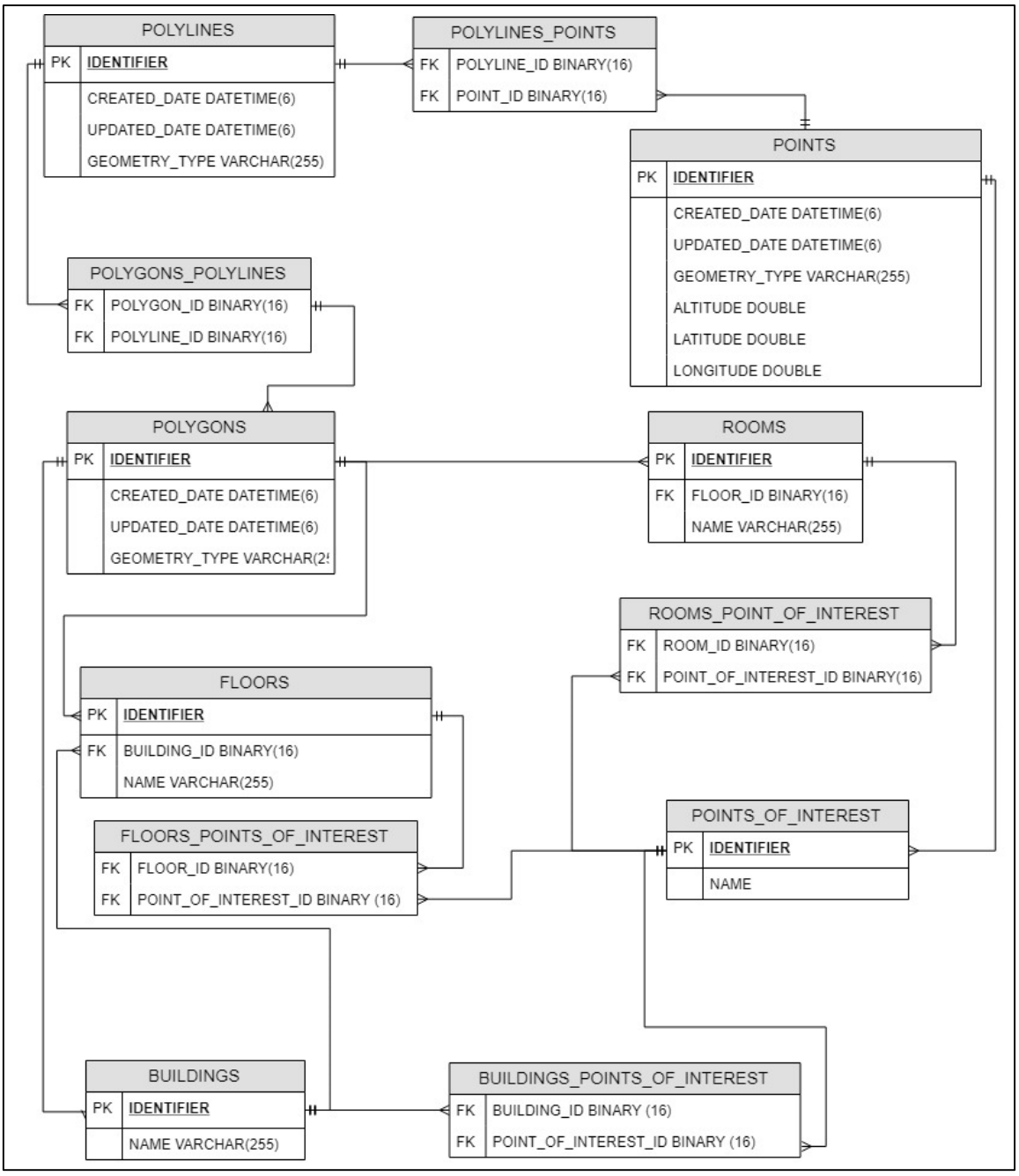


Figura 4.1: Modelo da base de dados com os dados necessários para representar entidades geográficas.

Tendo por base estas entidades geográficas é simples criar entidades, onde basta que estas referenciem qual(ais) o(s) contexto(s) e assim obter a sua representação geográfica, que é o caso das entidades Edifícios (*Buildings*), Pisos (*Floors*). Salas (*Rooms*) e Pontos de Interesse (*Point_Of_Interest*).

Na Figura 4.2 são apresentadas as entidades relacionados com o contexto de navegação, onde a diferença de estas para as entidades do espaço físico reside no facto de que estas não representarem algo físico no espaço. Funcionam como entidades que irão representar os dados necessários para o sistema poder calcular a navegação de um utilizador. Similarmente às entidades do espaço físico, estas também têm uma vertente geográfica já que têm relações com entidades do tipo Ponto e Linha, como é o caso das entidades Pontos de Controlo (*Control_Points*), Arestas (*Edges*).

A exclusão a entidades que tenham um contexto geográfico são as entidades que representam Frequências (*Frequencies*) e Pontos de Acesso (*Access_Points*). Será sobre estas que irá residir a lógica de providenciar a localização de um utilizador com base no Wi-Fi, que se encontra detalhado na secção 4.1.5.2.

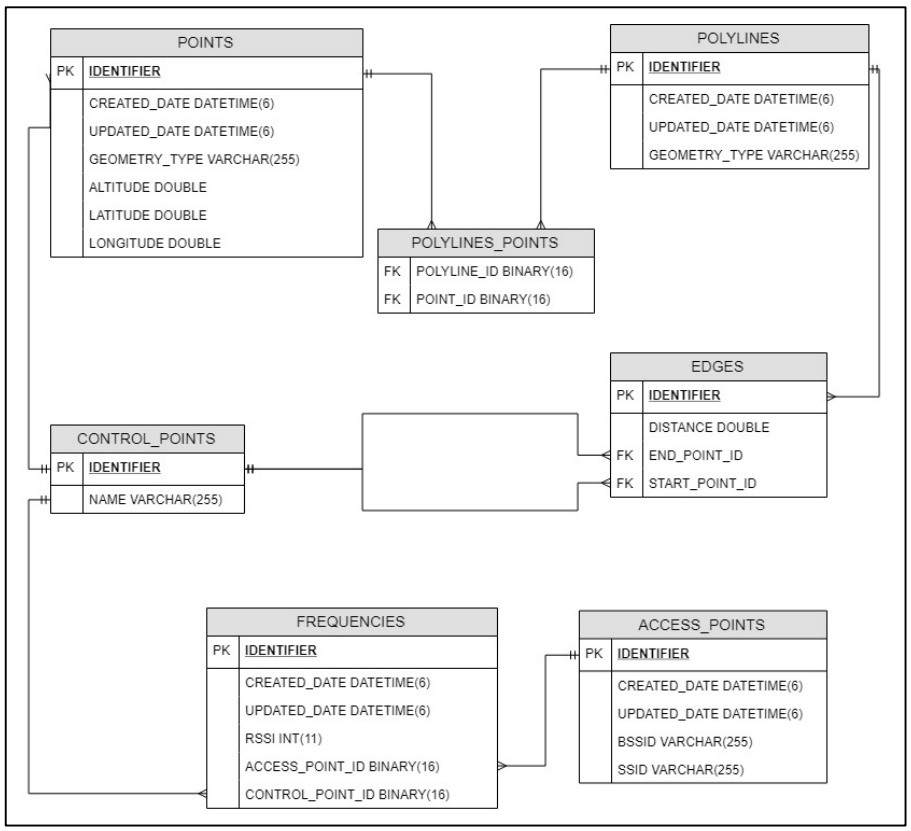


Figura 4.2: Modelo da base dados com os dados necessários para a rede de navegação.

A maioria destas entidades acabam depois por ser traduzidas para uma representação geográfica através da consulta do catálogo de serviços em que alguns cenários terão a sua representação em *GeoJSON*, detalhada nos capítulos seguintes.

4.1.2.2. Transformação e Carregamento de Dados

Relativamente aos dados que se encontram guardados no WINS estes foram extraídos de um Sistema de Informação Geográfica – SIG. Foram disponibilizados no âmbito da disciplina de Serviços de Localização e Geoinformação e correspondem ao piso 1 do edifício C8 da Faculdade de Ciências da Universidade de Lisboa. Sendo que se tratam de dados num contexto muito específico foi necessário a extração e transformação para o WINS. A sua representação espacial pode ser observada na Figura 4.3.

Estes dados representam de que modo um utilizador pode deslocar-se no espaço físico. Este terá a possibilidade de solicitar caminhos, do ponto de onde se encontra, para salas. O caminho sugerido será sempre providenciado sobre a linha "Caminho", conforme apresentado na Figura 4.3, sendo que este deslocar-se-á sobre esta. Sendo que uma sala é uma figura geométrica espacial o caminho será dado até à sua porta, aqui identificado como "Acesso".

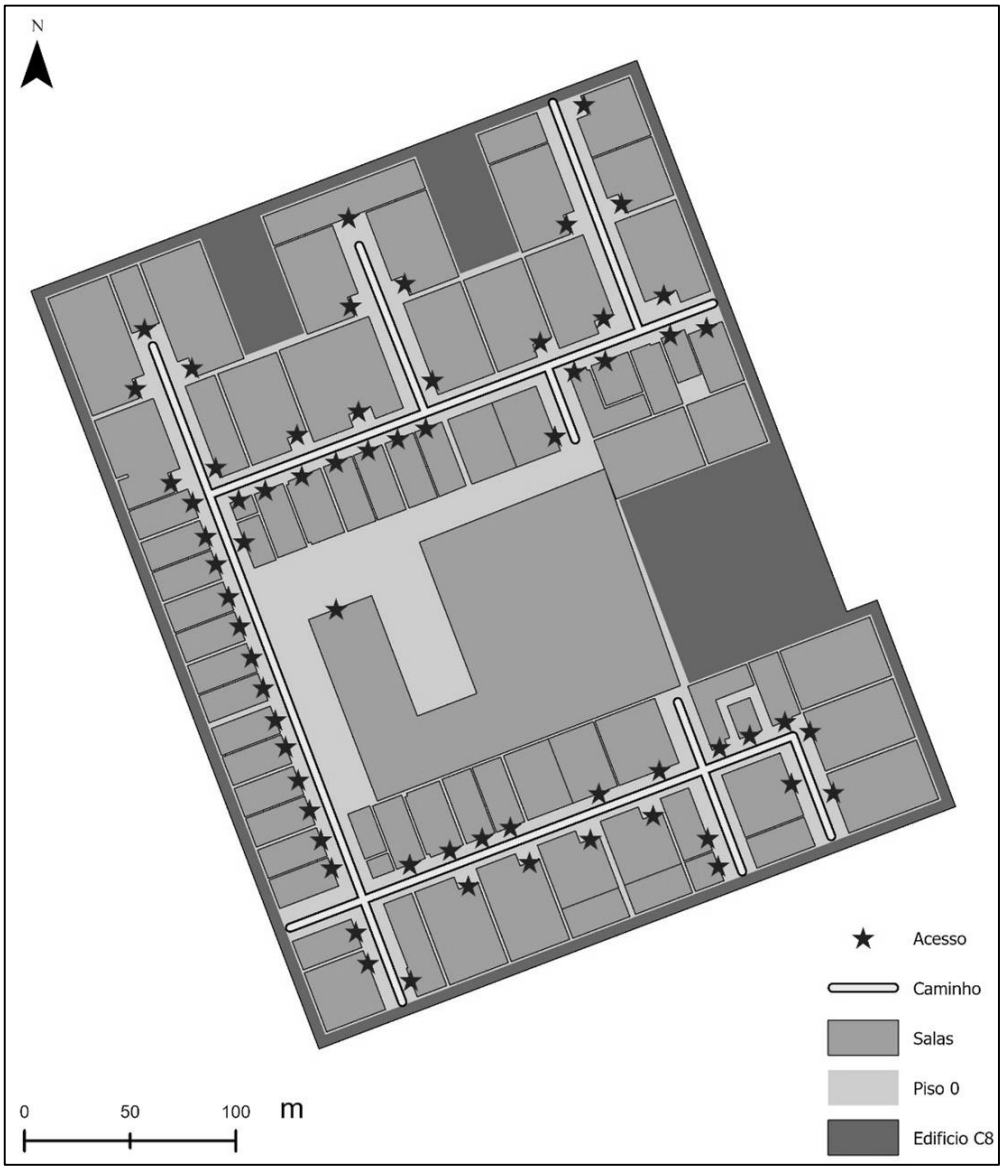


Figura 4.3: Representação espacial do piso 0 do Edifício C8 da FCUL.

Sendo que se trata de dados em formato vetorial, primeiro foi necessário converter estes dados.

Do ponto de vista de estrutura dos dados estes podem ser subdivididos em três formatos:

- Ponto – é dado com coordenadas geográficas X-Y;
- Linha – consiste num conjunto de dados representados por *n* Pontos;
- Polígono – conjunto de dados do tipo Linha que determinam uma área numa superfície. Importante de notar que as coordenadas do "primeiro" ponto de um polígono corresponde as coordenadas do "ultimo" Ponto, fazendo com que um Polígono seja a representação geométrica de uma área "fechada".

O primeiro passo consistiu na sua extração para um formato *GeoJSON*. Para tal foi utilizado o *Software* ArcGIS © ESRI através da ferramenta *Model Builder*, conforme demostrado na Figura 4.4

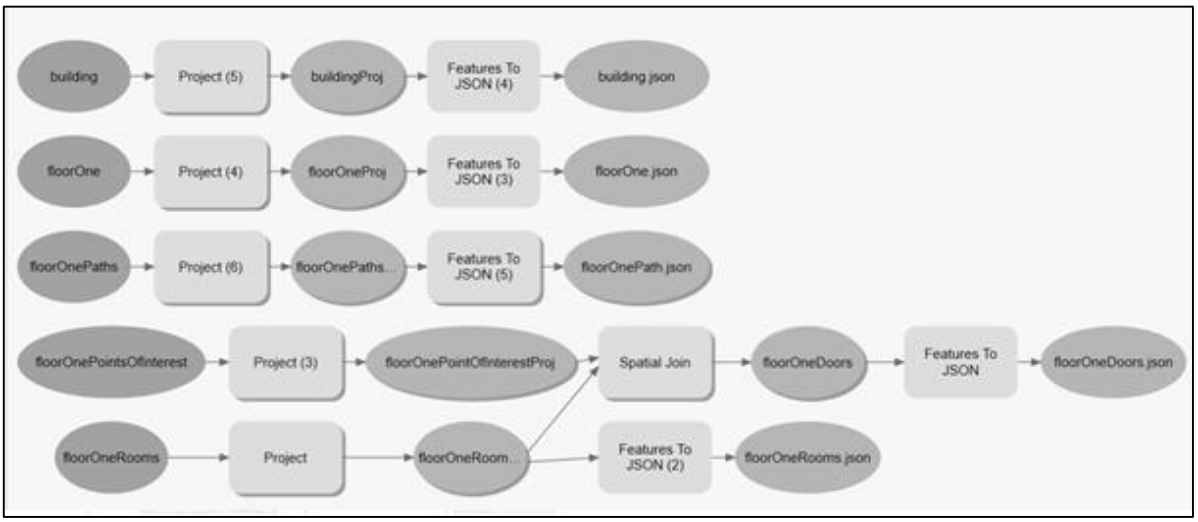


Figura 4.4: Ferramenta de extração de dados para formato GeoJSON.

Sendo que estes dados correspondem na sua origem a coordenadas geográficas não existe dificuldade na importação destes para o WINS, mas sim, algum tratamento prévio. O primeiro passo consiste na sua projeção para o sistema de coordenadas WGS84. Este é realizado de forma a que os dados possam ser mostrados na aplicação cliente. Após esta conversão é necessário apenas exportar estes ficheiros para o formato *GeoJSON* e importar para o sistema. De notar que a exceção a este processamento ocorre nos dados referentes a "Acesso" (na Figura 4.4 identificada como *floorOnePointsOfInterest*) onde estes de modo a poderem ser relacionáveis com as Salas, tem que ser realizado um *Spatial Join* que consiste na criação de uma relação entre duas, ou mais, entidades tendo por base as suas representações geográficas.

Após este processamento é realizada a sua importação para o WINS através de uma classe *Java* que efetua a leitura destes ficheiros e os guarda na base de dados.

4.1.3. Servidor Aplicacional

O servidor aplicacional é um sistema que fornece um conjunto de funcionalidades de negócio do *software* instalado e permite "desacoplar" a lógica do negócio com outras componentes, como rede, redundância de dados, alta disponibilidade, balanceamento de carga e segurança (Techopedia, 2019).

Para o contexto deste projeto, e tendo em conta que a linguagem escolhida foi o *Java*, foi decidido utilizar o servidor aplicacional *Apache Tomcat* ©. Este é um software de código aberto e implementa as tecnologias de *Java Servlet, JavaServer Pages, Java Expresion Language* e *Java WebSocket.* É desenvolvido como uma ferramenta de código aberto e lançado sobre o licenciamento da *Apache Licence version 2*. (Apache, 2019)

4.1.4. API

Nesta secção será detalhada a forma como a API disponibilizada pela WINS foi implementada.

Embora já tenha sido mencionado no capítulo anterior que o WINS irá disponibilizar uma API com um paradigma REST é importante referir que esta é apenas uma interface em que um sistema se consegue integrar com o WINS. Na sua génese o que o WINS pretende disponibilizar é um conjunto de interfaces a serem utilizadas por outros sistemas.

Similarmente quando se visita um restaurante e se solicita um menu para saber o que escolher, o WINS disponibilizará um "menu", daqui por diante mencionado como catálogo, que permitirá a um outro sistema solicitar pedidos e consequentemente obter respostas ao que solicitou.

Sendo que o objetivo deste projeto passa por disponibilizar um conjunto de serviços como uma API, foi escolhido o REST por ser uma norma ainda com um forte relevo para a comunidade de desenvolvimento (Higginbotham, 2018).

A API pode ser repartida em duas componentes, Pontos de Interesse e Navegação, desse modo, esta secção encontra-se dividida em duas famílias de serviços, sendo que todos os serviços presentes nas tabelas seguintes e identificados com "…" indica que são precedidos do URL *http://host:port/wifi-indoor-navigation-system/api/v1/.*

Importante de referir que aquando do desenvolvimento do catálogo de serviços REST do WINS foi também utilizado a especificação *OpenAPI* com o recurso da ferramenta *SWAGGER* © (Swagger, 2019). A razão de utilizar esta ferramenta e especificação recai na facilidade com que, no âmbito de desenvolvimento, é possível desenvolver e integrar automaticamente com o WINS, o que acaba por tornar o desenvolvimento mais simples e eficiente. Em termos práticos o que esta ferramenta disponibiliza é uma interface gráfica, que é acedida através de um browser em que é mostrado os vários campos para submeter um pedido HTTP. Esta permite visualizar um conjunto de informação referente à API e deste modo explorá-la. Expõe um conjunto de operações, como segurança, onde é definido se

os pedidos deverão ser solicitados através de HTTPS ou HTTP e um ícone de "*Authorize*" que se refere a vários protocolos de segurança que a API tenha decido utilizar, como o *OAuth2*.0 (RFC-6749, 2019)

Em termos práticos traduz um pedido, que consiste num pedido HTTP contido num envelope, para um formato percetível para o ser humano como demonstrado na Figura 4.5.

Tendo algum conhecimento prévio de um pedido HTTP é facilmente identificável onde se encontram as várias secções do pedido. Na Figura 4.5 é possível identificar que se trata de um pedido POST, que será realizado sobre o URL /pet. Neste pedido será enviado um objeto *JSON* com a estrutura mostrada na figura e as respostas possíveis terão um código 405.

Acaba por se tornar uma ferramenta basta útil para a integração com o WINS.

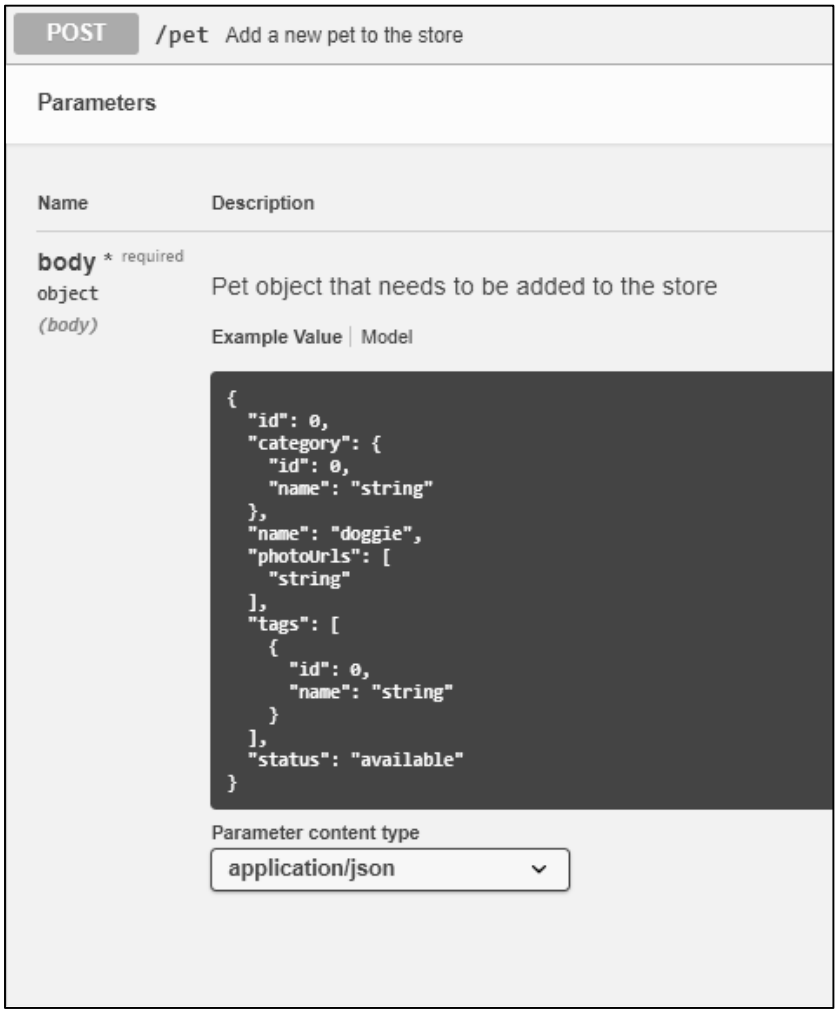


Figura 4.5: Representação de um pedido POST utilizando a ferramenta *Swagger*. (Swagger, 2019)

Serviços Pontos de Interesse

Estes serviços disponibilizam um conjunto de operações sobre algumas das entidades apresentadas na secção 4.1.2.1. São necessários de modo a que o sistema possa ser independente à sua área de estudo. Assim é possível interagir com estes serviços podendo manipular o estado dos dados que se encontram guardados no sistema.

As entidades em questão são nomeadamente as que tinham uma representação física, como Edifícios, Pisos, Salas e Pontos de Interesse. Mediante a operação solicitada o WINS terá um comportamento diferentes, mas todas as entidades dispõem de operações *Create, Read, Update, Delete* (CRUD) sendo que é possível consultar detalhadamente as operações possíveis na Tabela 4.1.

De notar que os diversos serviços estão contidos uns nos outros, ou seja, uma Sala para ser criada tem que ser criada contida num Piso, que por sua vez tem que ser contido num Edifício e todos estes serviços tem uma representação *GeoJson*.

Tabela 4.1: Catálogo de serviços correspondentes a pontos de interesse com o formato *GeoJson*.

Meta-informação	Dados	
Descrição:	Obtém todos os recursos do tipo de *building* (Anexos *Polygon*)	
Pedido:	URI:	…/geo-json/buildings
	Método:	GET
	Parâmetros:	Vazio
	Conteúdo:	Vazio
Resposta:	Conteúdo:	Lista de *buildings*
Descrição:	Cria um recurso do tipo *building* (Anexos *Polygon*)	
Pedido:	URI:	…/geo-json/buildings
	Método:	POST
	Parâmetros:	Vazio
	Conteúdo:	Objeto do tipo *building*
Resposta:	Conteúdo:	Id do recurso criado
Descrição:	Obtém um recurso do tipo *building* (Anexos *Polygon*)	
Pedido:	URI:	…/geo-json/buildings/{*buildingId*}
	Método:	GET
	Parâmetros:	*buildingId* Id do recurso a obter
	Conteúdo:	Vazio
Resposta:	Conteúdo:	Objeto do tipo *building*
Descrição:	Atualiza um recurso do tipo *building* (Anexos *Polygon*)	
Pedido:	URI:	…/geo-json/buildings/{*buildingId*}

	Método:	PUT	
	Parâmetros:	*buildingId*	Id do recurso a atualizar
	Conteúdo:	Objeto do tipo *building*	
Resposta:	Conteúdo:	Vazio	

Descrição:	Apaga um recurso do tipo *building* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId*}	
	Método:	DELETE	
	Parâmetros:	*buildingId*	Id do recurso a apagar
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Vazio	

Descrição:	Obtém todos os recursos do tipo de *floor* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors	
	Método:	GET	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *floor*	

Descrição:	Cria um recurso do tipo *floor* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors	
	Método:	POST	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
	Conteúdo:	Objeto do tipo *floor*	
Resposta:	Conteúdo:	Id do recurso criado	

Descrição:	Obtém um recurso do tipo *floor* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}	
	Método:	GET	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso a obter
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo *building*	

Descrição:	Atualiza um recurso do tipo *floor* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}	
	Método:	PUT	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso a atualizar

	Conteúdo:	Objeto do tipo *building*	
Resposta:	Conteúdo:	Vazio	
Descrição:	Apaga um recurso do tipo *floor* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}	
	Método:	DELETE	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso a apagar
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Vazio	
Descrição:	Obtém todos os recursos do tipo de *room* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}/rooms	
	Método:	GET	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso correspondente ao *floor*
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *room*	
Descrição:	Cria um recurso do tipo *room* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}/rooms/{*roomId*}	
	Método:	POST	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso correspondente ao *floor*
	Conteúdo:	Objeto do tipo *room*	
Resposta:	Conteúdo:	Id do recurso criado	
Descrição:	Obtém um recurso do tipo *room* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}/rooms/{*roomId*}	
	Método:	GET	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso correspondente ao *floor*
		roomId	Id do recurso a obter
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo *room*	
Descrição:	Atualiza um recurso do tipo *room* (Anexos *Polygon*)		

Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}/rooms/{*roomId*}	
	Método:	PUT	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso correspondente ao *floor*
		roomId	Id do recurso a atualizar
	Conteúdo:	Objeto do tipo *room*	
Resposta:	Conteúdo:	Vazio	
Descrição:	Apaga um recurso do tipo *room* (Anexos *Polygon*)		
Pedido:	URI:	…/geo-json/buildings/{*buildingId}*/floors/{*floorId*}/rooms/{*roomId*}	
	Método:	DELETE	
	Parâmetros:	*buildingId*	Id do recurso correspondente ao *building*
		floorId	Id do recurso correspondente ao *floor*
		roomId	Id do recurso a apagar
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Vazio	
Descrição:	Obtém todos os recursos do tipo de *point-of-interest* (Anexos *Point*)		
Pedido:	URI:	…/geo-json/points-of-interest	
	Método:	GET	
	Parâmetros:	Vazio	
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *point-of-interest*	
Descrição:	Cria um recurso do tipo *point-of-interest* (Anexos *Point*)		
Pedido:	URI:	…/geo-json/points-of-interest/{*pointOfInterestId*}	
	Método:	POST	
	Parâmetros:	Vazio	
	Conteúdo:	Objeto do tipo *point-of-interest*	
Resposta:	Conteúdo:	Id do recurso criado	
Descrição:	Obtém um recurso do tipo *point-of-interest* (Anexos *Point*)		
Pedido:	URI:	…/geo-json/points-of-interest/{*pointOfInterestId*}	
	Método:	GET	
	Parâmetros:	*pointOfInterestId*	Id do recurso a obter
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo *point-of-interest*	
Descrição:	Atualiza um recurso do tipo *point-of-interest* (Anexos *Point*)		

Pedido:	URI:	…/geo-json/points-of-interest/{*pointOfInterestId*}
	Método:	PUT
	Parâmetros:	*pointOfInterestId* Id do recurso a atualizar
	Conteúdo:	Objeto do tipo *point-of-interest*
Resposta:	Conteúdo:	Vazio
Descrição:	Apaga um recurso do tipo *point-of-interest* (Anexos *Point*)	
Pedido:	URI:	…/geo-json/points-of-interest/{*pointOfInterestId*}
	Método:	DELETE
	Parâmetros:	*pointOfInterestId* Id do recurso a apagar
	Conteúdo:	Vazio
Resposta:	Conteúdo:	Vazio

Como é possível observar na *Tabela 4.1* e na *Tabela 4.2* os serviços encontram-se representados de duas formas:

- *GeoJSON*
- *JSON*

A primeira corresponde à representação geográfica de pontos de interesse, onde a sua representação é uma representação que segue o *standard GeoJSON* (RFC-7946, 2019).

A segunda é uma representação mais simples, onde apenas são apresentados os seus atributos de negócio, como por exemplo o nome da sala, podendo dizer-se que são os dados que o utilizador consegue interpretar e atribuir a sua conotação lógica.

Esta diferenciação acontece devido ao facto de o custo computacional ser superior nas representações em *GeoJson*, já que estas têm mais informação relativa aos recursos. Assim é possível consultar os dados sem ter a necessidade de obter a sua representação geográfica, por Ex: em listas.

Outra diferença passa pelas funcionalidades, onde nos primeiros tipos de serviços existe a possibilidade de executar pedidos POST, ou seja, criar recursos com um identificador único. Este identificador é depois utilizado para obter os recursos criados, tanto no primeiro tipo de serviços com o segundo tipo.

Tabela 4.2: Catálogo de serviços correspondentes a pontos de interesse com o formato JSON.

Meta-informação	Dados		
Descrição:	Obtém todos os recursos do tipo de *point-of-interest* (Anexos *PointOfInterest*)		
Pedido:	URI:	…/points-of-interest	
	Método:	GET	
	Parâmetros:	Vazio	
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *point-of-interest*	
Descrição:	Obtém um recurso do tipo *point-of-interest* (Anexos *PointOfInterest*)		
Pedido:	URI:	…/points-of-interest/{*pointOfInterestId*}	
	Método:	GET	
	Parâmetros:	Vazio	
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo *point-of-interest*	
Descrição:	Obtém um recurso do tipo *point-of-interest* (Anexos *Room*)		
Pedido:	URI:	…/points-of-interest/{*pointOfInterestId*}/rooms	
	Método:	GET	
	Parâmetros:	*pointOfInterestId*	Id do recurso correspondente ao *point-of-interest*
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	List de *room*	

Serviços para Navegação

Na Tabela 4.3 é possível ver o catálogo de serviços correspondente aos serviços correspondentes ao catálogo de navegação.

Tabela 4.3: Catálogo de serviços respetivos a pontos de interesse.

Meta-informação	Dados		
Descrição:	Obtém todos os recursos do tipo de *controlPoint* (Anexos *ControlPoint*)		
Pedido:	URI:	…/navigation/control-points	
	Método:	GET	
	Parâmetros:	*frequencies*	Lista de RSSIs
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *controlPoint*	
Descrição:	Cria um recurso do tipo *controlPoint* (Anexos *ControlPoint*)		
Pedido:	URI:	…/navigation/control-points	
	Método:	POST	

	Parâmetros:	Vazio	
	Conteúdo:	Objeto do tipo *controlPoint*	
Resposta:	Conteúdo:	Id do recurso criado	
Descrição:	Obtém o recurso *controlPoint* (Anexos *ControlPoint*)		
Pedido:	URI:	…/navigation/control-points/{*controlPointId*}	
	Método:	GET	
	Parâmetros:	*controlPointId*	Id do recurso solicitado
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo *controlPoint*	
Descrição:	Cria um recurso *controlPoint* (Anexos *ControlPoint*). Permite criar uma ligação entre um ponto de controlo para outro, funcionando como a criação de um novo vértice numa estrutura de grafo		
Pedido:	URI:	…/navigation/control-points/{*controlPointId*}	
	Método:	POST	
	Parâmetros:	*controlPointId*	Id do recurso fonte de este novo recurso
	Conteúdo:	Objeto do tipo *controlPoint*	
Resposta:	Conteúdo:	Id do recurso criado	
Descrição:	Obtém todos os recursos do tipo de *frequency* (Anexos *Frequency*)		
Pedido:	URI:	…/navigation/control-points/{*controlPointId*}/frequencies	
	Método:	GET	
	Parâmetros:	*controlPointId*	Id do recurso de tipo *controlPoint*
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Lista de *frequency*	
Descrição:	Cria um recurso do tipo de *frequency* (Anexos *Frequency*)		
Pedido:	URI:	…/navigation/control-points/{*controlPointId*}/frequencies	
	Método:	POST	
	Parâmetros:	*controlPointId*	Id do recurso de tipo *controlPoint*
	Conteúdo:	Objeto do tipo *frequency*	
Resposta:	Conteúdo:	Lista de *frequency*	
Descrição:	Calcula uma rota entre os parâmetros providenciados (Anexos *LineString*)		
Pedido:	URI:	…/navigation/paths	
	Método:	GET	
	Parâmetros:	*controlPointId*	Id do recurso do ponto de partida
		pointOfInterestId	Id do recurso de destino
	Conteúdo:	Vazio	
Resposta:	Conteúdo:	Objeto do tipo GeoJson LineString	

Estes acabam por ser menos extensos que os serviços apresentados na secção anterior, mas têm também uma importância bastante relevante já que estes serviços permitem que um cliente obtenha um posicionamento e consequentemente uma rota até um ponto de interesse.

É importante de referir que o modo como o WINS gera uma rota é através de uma computação em que o posicionamento do utilizador não é algo relativo. Ou seja, um cliente ao tentar obter uma rota de navegação, será posicionado sobre um ponto conhecido num caminho, similarmente a um comboio numa rede ferroviária. A Figura 4.6 permite mostrar esta situação onde o utilizador é reposicionado. O que acontece é que o sistema interpreta a informação do cliente e em vez de ter uma posição relativa do utilizador obtêm uma posição absoluta deste, posicionando-o sobre uma grelha de navegação o que permite que este navegue contido e limitado nesta. O detalhe desta implementação encontra-se com detalhe na secção 4.1.5.2.

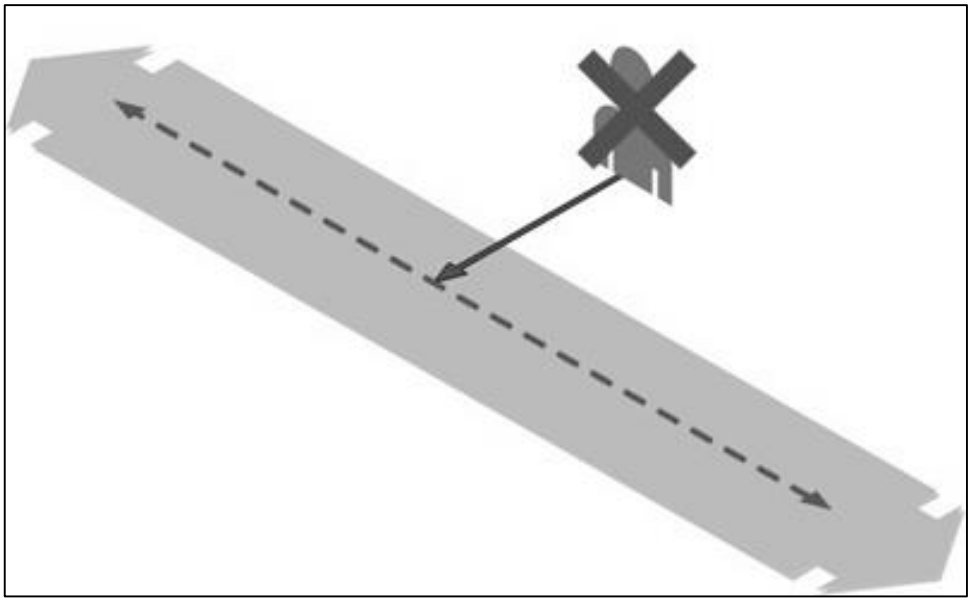


Figura 4.6: Abordagem de posicionamento do WINS.

4.1.5. Lógica Computacional

Esta secção encontra-se subdividida em duas secções:

- Arquitetura de componentes
- Posicionamento & Navegação

A primeira corresponde à arquitetura de componentes escolhida para o desenvolvimento dos diversos serviços implementados, explica que tecnologias foram utilizadas no contexto do desenvolvimento aplicacional e providencia uma explicação do comportamento genérico de como os serviços estão implementados e que tipos de padrões de desenvolvimento são possíveis de encontrar.

Na segunda secção é detalhado o modo de criação e processamento da informação no WINS. Será explicado a forma como o WINS obtêm o posicionamento de um cliente e de que forma, através deste posicionamento consegue providenciar uma rota para chegar ao ponto de interesse solicitado.

4.1.5.1. Arquitetura de Componentes

A arquitetura escolhida é bastante simples. Sendo que o objetivo do WINS é disponibilizar uma interface REST foi escolhido o padrão *Model-View-Controller* (MVC) (Microsoft, 2019). A Figura 4.7 mostra como este funciona no âmbito do WINS.

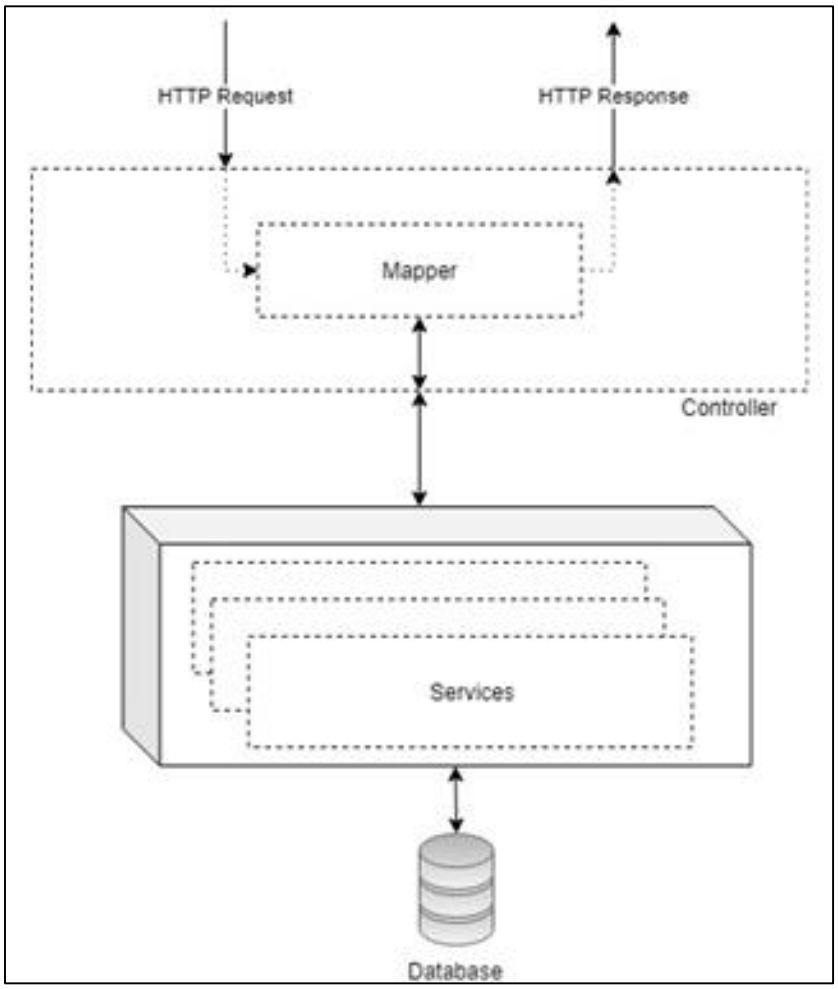


Figura 4.7: Padrão MVC do WINS.

Ao existir um pedido para um URL um *Controller* é providenciado pelo servidor aplicacional para responder a este pedido. Após a resolução interna do *Controller* o pedido HTTP passa para um contexto *Java*. Este passo consiste na serialização do pedido HTTP para objetos no contexto de uma *Java Virtual Machine* (JVM), tendo desse modo um ciclo de vida gerido por esta.

Mediante o tipo de pedido poderá ser necessário um conjunto de validações de negócio, como por exemplo validar que um objeto *GeoJson* está em conformidade com a sua especificação definida no

RFC 7946. Para tal é utilizada a norma *Java Specification Request (JSR) 303* (Emmanuel Bernard, 2019). Esta define um modo de validar um objeto do tipo *Bean* de *Java* de forma a que a validação de um objeto seja feita via anotação o que permite ter o código desacoplado.

Apos esta validação o *Controller* realiza a conversão deste objeto para um objeto de negócio. Esta conversão é realizada de modo a garantir que alguns tipos de validações necessárias ao contexto do pedido, e não de negócio, são realizados inicialmente, garantido deste modo que quando o objeto de negócio é instanciado, se encontra corretamente serializado não existindo objetos "inconsistentes" no contexto aplicacional. Este objeto de negócio é na realidade a entidade que representa os dados que irão ser guardados na base de dados – *Entity*. Para ser obtido este objeto o *Controller* invoca o *Mapper* correspondente de modo a obter o modelo (*Model* do MVC) fazendo o envio deste para uma camada aplicacional de *Services*, invocando o correspondente, mediante o contexto do pedido.

Esta separação entre camadas permite que o código se encontre desacoplado e sendo que esta comunicação é realizada via Interface, aquando novos requisitos de negócio apenas será necessário providenciar uma nova implementação da Interface.

Na camada de *Services* reside a lógica de negócio, podendo serem desencadeados vários processos, mediante a complexidade da ação solicitada. É também nesta camada que existe o acesso à *Database* (base de dados). Este acesso à base de dados é realizado utilizando *Java Persistence Api* (JPA) o que permite guardar e obter objetos serializados *Java* através da invocação de comandos abstratos sobre o modelo corrente (Oracle, 2019).

Após o processamento pela camada de *Services* é devolvido o modelo ao *Controller*. Mediante a ação solicitada no pedido HTTP este poderá ter a necessidade de chamar outro *Service* para articular outra lógica de negócio. Assim um *Controller* pode ser considerado como um orquestrador de várias lógicas de negócio. No caso do WINS funciona como uma camada que abstrai um pedido HTTP para objetos que depois são interpretados pelo sistema.

Acabando as várias iterações necessárias do pedido solicitado, o *Controller* reutiliza o *Mapper* para serializar os objetos *Java* para *GeoJSON* ou *JSON* e consequentemente devolve o pedido via HTTP para o solicitador do pedido.

No caso do WINS a componente *View* deste padrão são os próprios serviços. Ou seja, devido ao facto de não existir uma interface gráfica dos serviços, estes acabam por ser a sua própria V*iew*. Utilizando o *Swagger* e os diversos serviços REST desenvolvidos podemos visualizar os modelos e realizar operações sobre estes.

O desenvolvimento do servidor foi feito em *Java* e como tal foi escolhida a *framework Spring* © (Pivotal, 2019). A escolha desta *framework* foi devido ao facto de possibilitar um conjunto de opções de desenvolvimento que permite criar um conjunto de abstrações simplificando o processo de desenvolvimento. Sendo as principais a inversão de controlo (Inversion of Control - IOC) permitindo

dependency injection, assim como o uso de JTA de modo a não ser necessário controlar transacional da aplicação, já que estes são garantidos por esta framework e pelo servidor aplicacional (Oracle, 2019).

De uma forma geral os serviços são implementações simples de operações como criar, ler, atualizar e apagar (*Create*, *Read*, *Update*, *Delete* – CRUD) (Microsoft, 2019). Isto permite que clientes da API, como o caso da aplicação desenvolvida no âmbito deste projeto, utilizem o WINS com toda a liberdade possível.

4.1.5.2. Posicionamento & Navegação

Tanto a componente de posicionamento como a de navegação são as componentes mais complexas inseridas no contexto deste projeto. O seu funcionamento e modelo consiste numa abordagem bastante simples.

Em primeiro lugar, de modo a poder existir uma componente de posicionamento e navegação tem de existir uma calibração prévia. Ou seja, tem de existir uma estrutura que permita identificar onde um utilizador se encontra e onde se pode deslocar tendo por base a valor dos RSSI dos diversos AP.

Das várias abordagens apresentadas no Capítulo 2 foi escolhida a opção em que tanto o cálculo do posicionamento como o da navegação seja realizado através de amostragem.

Enquanto algumas soluções para o posicionamento com base em Wi-Fi se baseiam num mapa de intensidade, onde são identificadas zonas "quentes" onde a sua interceção com os RSSI dos AP registados no dispositivo permitem extrapolar a localização do utilizador, o WINS tem uma abordagem diferente.

A abordagem do WINS acaba por ser algo menos complexo, pelo menos a nível do cálculo de posicionamento. Este assume que existe uma amostragem prévia de um conjunto de pontos de controlo. Estes pontos acabam por criar uma grelha num modelo tipo grafo, grafo esse navegável pelo utilizador. Esta amostragem prévia compreende o registo dos RSSI dos AP que se encontram ao seu alcance, calibrando assim cada ponto de controlo com os dados daquele ponto em concreto. Através desta grelha e possível extrapolar e inferir a localização de um utilizador num espaço físico tendo por base os RSSI dos AP em seu redor.

É assumido que, embora a precisão seja importante, este não é o fator mais importante para o sistema. O que é importante é que um utilizador que utilize o WINS consiga deslocar-se na infraestrutura da FCUL e chegar a pontos de interesse utilizando esta aplicação, deixando o aumento da precisão da localização para futuros trabalhos.

O que o sistema deverá concretizar é o posicionamento do utilizador sobre uma grelha de calibração de diversos pontos de controlo navegáveis entre si. Este irá posicionar o utilizador o mais perto possível de um destes pontos de controlo. O WINS identifica qual o ponto de controlo mais perto do utilizador

através de um sistema de voto que consiste em saber quais os pontos de controlo que se classificam melhor com base nos diversos RSSI que o cliente identifica em seu redor. Sendo que a melhor classificação é fornecida ao cliente. Na Figura 4.8 podemos observar um cenário onde existem 3 pontos de controlo para 3 AP e um utilizador com uma aplicação cliente.

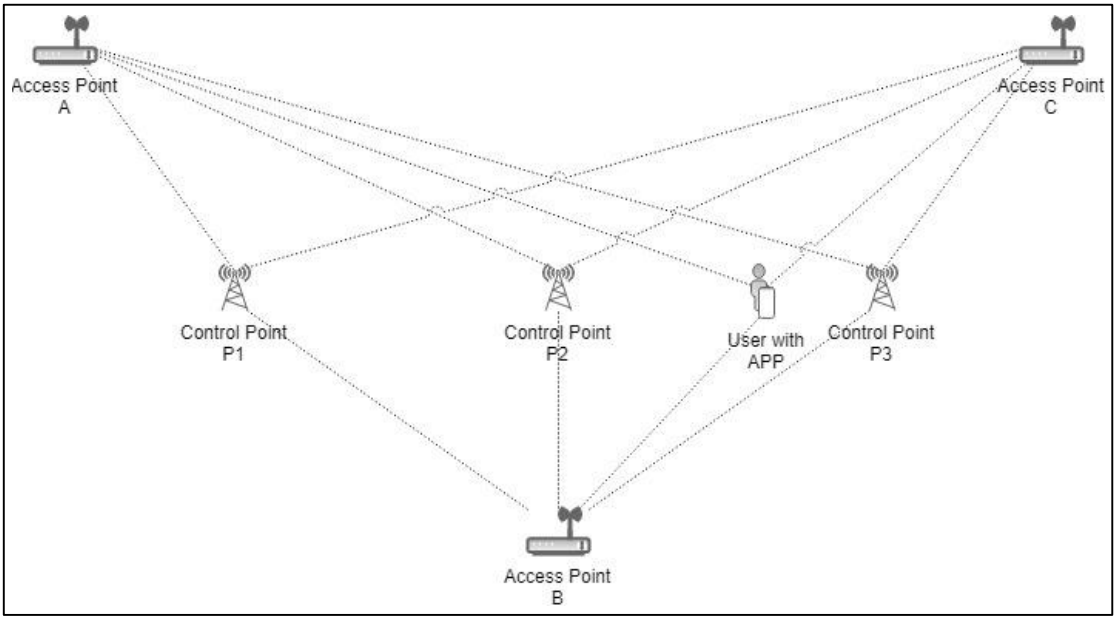


Figura 4.8: Utilizador num cenário de posicionamento WINS.

Sendo que cada ponto de controlo teve uma calibração prévia é possível simular a posição de um utilizador com base nos RSSI que o seu dispositivo consegue obter, sendo esses dados mostrados na *Tabela 4.4.*

Tabela 4.4: Medições de exemplo para um cenário de posicionamento WINS.

Ponto de Controlo	AP	RSSI
P1	A	-35
P1	B	-40
P1	C	-80
P2	A	-70
P2	B	-30
P2	C	-70
P3	A	-80
P3	B	-40
P3	C	-35
Utilizador	A	-75
Utilizador	B	-37
Utilizador	C	-37

O que o sistema realiza pode ser dividido em três passos.

O primeiro consiste em identificar quais os AP que podem ser utilizados, funcionando como um filtro aos dados que existem no WINS tornando a computação mais eficiente. Não interessa calcular distâncias entre AP que não foram identificados pelo dispositivo já que se encontram fora do alcance deste.

O segundo, passo mais complexo, consiste em iterar sobre os AP filtrados e destes identificar qual o RSSI medido que mais se aproxima ao RSSI captado pelo dispositivo, sendo que este é elegível para ser considerado. Como cada RSSI registado no sistema tem sempre uma bidirecionalidade, tanto para o AP como para o ponto de controlo, ao identificar qual o RSSI mais perto, é possível obter qual o ponto de controlo correspondente. É este ponto de controlo que depois é adicionado para uma "urna" de votação. A *Tabela 4.5* apresenta os resultados de uma simulação de iteração com base nos dados simulados da *Tabela 4.4* para um utilizador

Tabela 4.5: Processamento de medições de exemplo para um cenário de posicionamento WINS.

AP	RSSI Utilizador	Ponto de Controlo	RSSI	Diferença	Elegível
A	-75	P1	-35	40	Não
		P2	-70	5	Sim
		P3	-80	5	Sim
B	-37	P1	-40	3	Sim
		P2	-30	7	Não
		P3	-40	3	Sim
C	-37	P1	-80	43	Não
		P2	-70	33	Não
		P3	-35	2	Sim

O terceiro e último passo, corresponde em iterar sobre a "urna" gerada no segundo passo e identificar qual o ponto de controlo que foi votado mais vezes como o ponto de controlo mais elegível, sendo este devolvido para o cliente do serviço. Com base na *Tabela 4.5* corresponderia ao ponto de controlo P3, já que este teve 3 votos, seguido do P2 com 2 votos e o P1 com 0 votos.

Uma das vantagens de utilizar este tipo de abordagem é que cada ponto de controlo terá uma assinatura própria. Assim, pontos de controlo que sejam identificados com uma latitude e longitude igual, mas com altitudes diferentes, terão uma assinatura distinguível pelo WINS, já que a calibração de cada ponto terá RSSIs diferentes, fazendo com que problemas de colisões entre pontos de controlo em pisos diferentes sejam inexistentes.

Poderá dar-se o cenário em que existem colisões de pontos de controlo elegíveis, pontos de controlo em que o sistema de voto elegeu um, ou mais pontos de controlo como os mais propícios a serem o ponto da localização do utilizador. Neste caso o WINS não toma nenhuma decisão. Para todos os efeitos é

dada uma posição do cliente aproximada, o que do ponto de vista do sistema é suficiente tendo em conta os dados providenciados. Compete depois à aplicação cliente gerir estas colisões e posicionar o utilizador corretamente no espaço. Importante referir que no âmbito deste projeto tal resolução não é contemplada já que não são desenvolvidos mecanismos para contemplar este problema, sendo que uma possível solução poderia passar por *sensor fusion* ou *dead* reckoning conforme mencionado no Capítulo 2.

Outro cenário consiste no facto de o sistema não conseguir obter a localização inicial do utilizador. Este poderá surgir devido à rede ter perdido a sua calibração, já que estamos num contexto que se degrada ao longo do tempo, ou ao facto de o utilizador se encontrar numa área que não foi corretamente calibrada.

Este cenário será classificado como um *outlier* que também deverá ser resolvido em trabalhos futuros, onde poderão ser utilizadas outras tecnologias de posicionamento para garantir que tal não suceda.

Assim, partindo do pressuposto que é obtida a localização do utilizador, o contexto da navegação torna-se um processo simples de realizar. Sendo que a localização do utilizador e a sua deslocação é realizada e limitada por física, por exemplo, é impossível um utilizador que se encontra no piso 0 se deslocar imediatamente para o piso 5, a navegação e o cálculo de que rota se deve seguir para se chegar ao ponto de interesse solicitado é algo simples de calcular já que um utilizador irá deslocar-se sobre esta rede, similarmente ao modo como um comboio de desloca sobre um rede ferroviária.

Para calcular qual o caminho a utilizar é utilizado o algoritmo de *Dijkstra*. Este adapta-se bem à solução adotada, já que o utilizador se encontra sobre uma estrutura que representa um grafo (Paraschiv, 2019).

Embora possam existir outros algoritmos que possam ser mais otimizados computacionalmente, tais não são mencionados devido ao facto do algoritmo *Dijkstra* permitir encontrar um caminho sobre uma estrutura de grafo.

4.2. Cliente – *Front-end*

O Cliente, similarmente ao servidor e no contexto desta dissertação, acaba por ser uma componente principal do WINS, não sendo possível demostrar as capacidades e funcionalidades do sistema sem esta componente. Ao contrário do Servidor, esta é uma peça facilmente substituível por outra (desde que interaja conforme a especificação) já que o Servidor é agnóstico ao Cliente.

É assumido que o cliente não terá necessidade de criar dados do tipo de pontos de interesse. Estes já se encontram carregados e guardados no sistema, ação realizada na secção 4.1.2.2.

Deste modo esta seção encontra-se subdividida em três aspetos:

- Arquitetura Aplicacional
- Calibração do modelo
- Navegação

4.2.1. Arquitetura Aplicacional

Sendo que esta componente acaba por ser apenas uma aplicação para demonstrar as funcionalidades do servidor, esta não foi desenvolvida tendo em conta normas e boas práticas de desenvolvimento no SO *Android*. O que esta componente tem de realizar é a sua integração com o servidor e providenciar os requisitos para que este consiga calcular tanto a localização do utilizador como da sua rota.

É importante referir que a *Google* recomenda que o desenvolvimento *Android* seja realizado numa programação de tipo assíncrona. Em termos práticos o código que esteja a consumir serviços deve ser realizado num padrão de desenho de *callback*, onde quando o serviço acaba a sua execução, os dados obtidos são transmitidos para o *User Interface* (UI) neste momento. Deste modo foi criada uma estrutura aplicacional que garante este comportamento, de modo a que a aplicação não tenha bloqueios enquanto o utilizador a está a utilizar.

Na Figura 4.9 é possível observar de que modo esta gestão de ciclo de vida é implementada nesta componente.

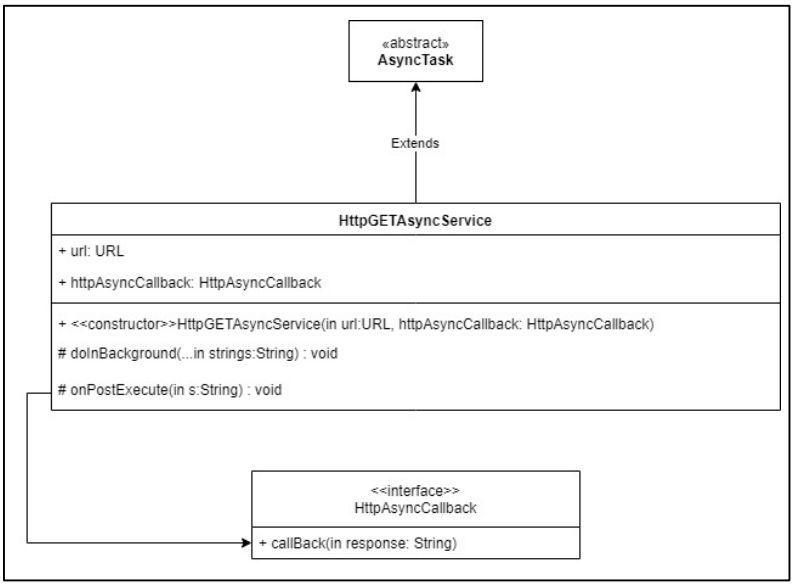


Figura 4.9: Arquitetura aplicacional de pedidos HTTP.

Conforme o tipo de pedido a realizar, é instanciada uma classe que estende a classe *AsyncTask* providenciada pela API do *Android*. No caso da Figura 4.9 é mostrado como é realizada um GET, onde a classe *HttpGETAsyncService* recebe como parâmetros o URL e uma interface *HttpAsyncCallBack*. Após a sua execução gerida pelo SO *Android*, é chamado o método *onPostExecute* que por sua vez irá chamar o método *callBack* dando assim a resposta do serviço. Sendo que se trata de uma *interface* apenas é necessário implementar esta por classes que pretendam consumir serviços REST, obtendo assim desta maneira comunicação assíncrona com o servidor WINS.

Sendo que o objetivo deste projeto é mostrar a posição e a navegação do utilizador, é importante mencionar que foi escolhido o *Standard Development Kit* (SDK) da *Mapbox* © (mapbox, 2019). Este

permite efetuar a apresentação de um mapa e subsequente amostragem de estruturas vetoriais como polígonos, linhas e pontos, podendo providenciar uma imagem ao utilizador do seu redor. Outra vantagem encontra-se também pelo facto de este produto utilizar o formato *GeoJSON*, o que torna a sua integração mais simples em função da estrutura desenvolvida pelo servidor do WINS.

Existem diversas *Activities* implementadas na aplicação Cliente do WINS. De uma maneira geral todas tem a sua importância, embora a principal seja a *MainActivity*. Esta permite ao utilizador visualizar o que está a acontecer em seu redor do ponto de vista geométrico e apresenta onde este se encontra. Caso este decida entrar em modo de navegação, ou seja, solicitar o caminho para um ponto de interesse, como uma sala, esta irá apresentar no mapa qual o caminho a percorrer.

O modo como esta *Activity* mostra os seus dados é através da iteração de pedidos HTTP ao servidor WINS de modo a que este devolva os dados previamente carregados. Sendo que esta aplicação apenas foi criada com o intuito de mostrar os dados carregados no WINS não existe uma lógica que permita ir obter os dados de forma inteligentes, como verificar a localização do utilizador e somente ir buscar estes. Funciona quase como um *dump* de informação previamente carregada.

Na Figura 4.10 é possível visualizar como os dados obtidos do servidor se encontram distribuídos geográfica e geometricamente, pelo que é a primeira vez em que a concetualização do sistema WINS apresenta resultados. A este modo de interação é dado o nome de modo *idle*.

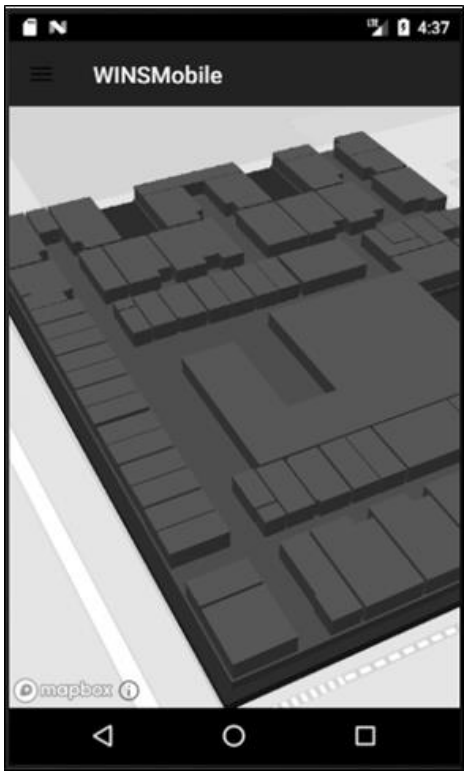


Figura 4.10: Modelo i*dle* do WINS cliente

4.2.2. Calibração do Modelo

De forma a que o sistema forneça a localização e dessa maneira possa calcular um caminho para o utilizador utilizar é necessária uma fase de calibração prévia. Uma vez que o WINS assume uma área de controlo, onde o utilizador depois navega sobre esta, é necessário calibrar a grelha correspondente a esta área.

Desse modo a aplicação tem um modo de calibração que permite ao utilizador informar onde se encontra geograficamente, e ao pressionar neste ponto recolhe os RSSI dos diversos AP em seu redor. Ao enviar estes dados para o servidor, este terá os dados necessários para fornecer a localização do utilizador mais tarde. De modo a que o utilizador consiga aceder a este modo, basta pressionar o botão *"Calibration Mode"* na *side-bar* para tal como apresentado na Figura 4.11.

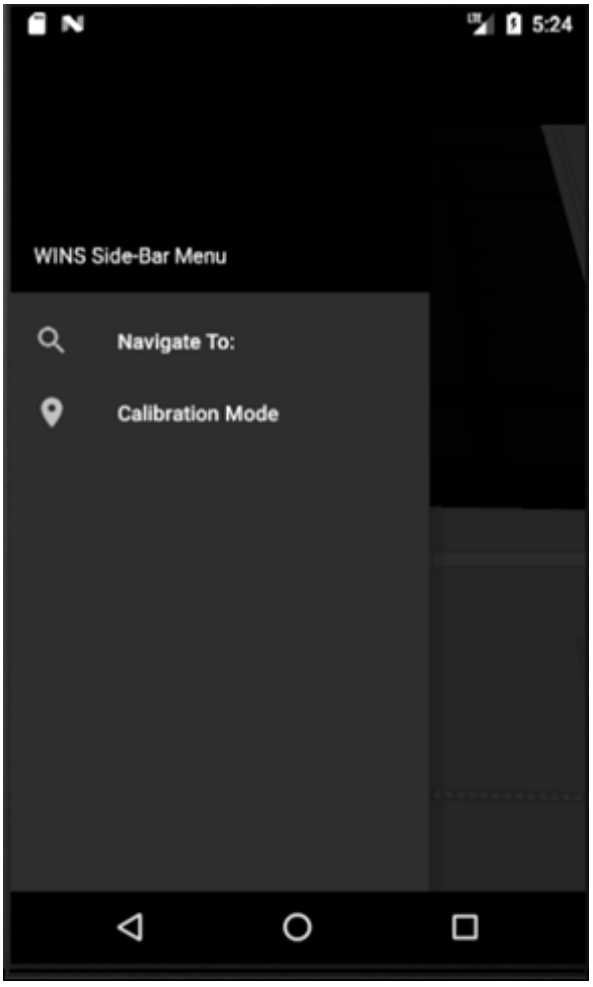


Figura 4.11: Menu para aceder ao modo de calibração.

Ao pressionar neste botão surgirá um menu bastante semelhante ao modo *idle*. A diferença encontra-se na representação dos diversos pontos de controlo, como representado na Figura 4.12

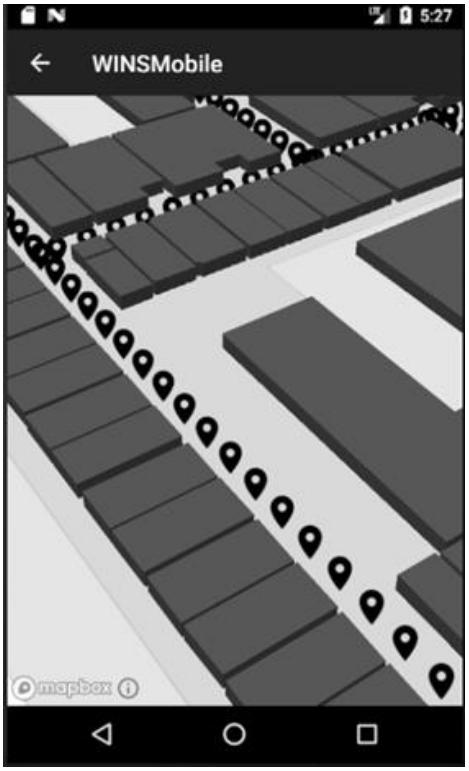


Figura 4.12: Modo calibração.

Sendo que o objetivo deste modo é calibrar a rede, ao pressionar qualquer ponto irá surgir um menu que mostra os RSSI registados neste ponto de controlo, sendo que se for a primeira vez que este é calibrado, esta lista surgirá vazia, conforme mostrado na Figura 4.13.

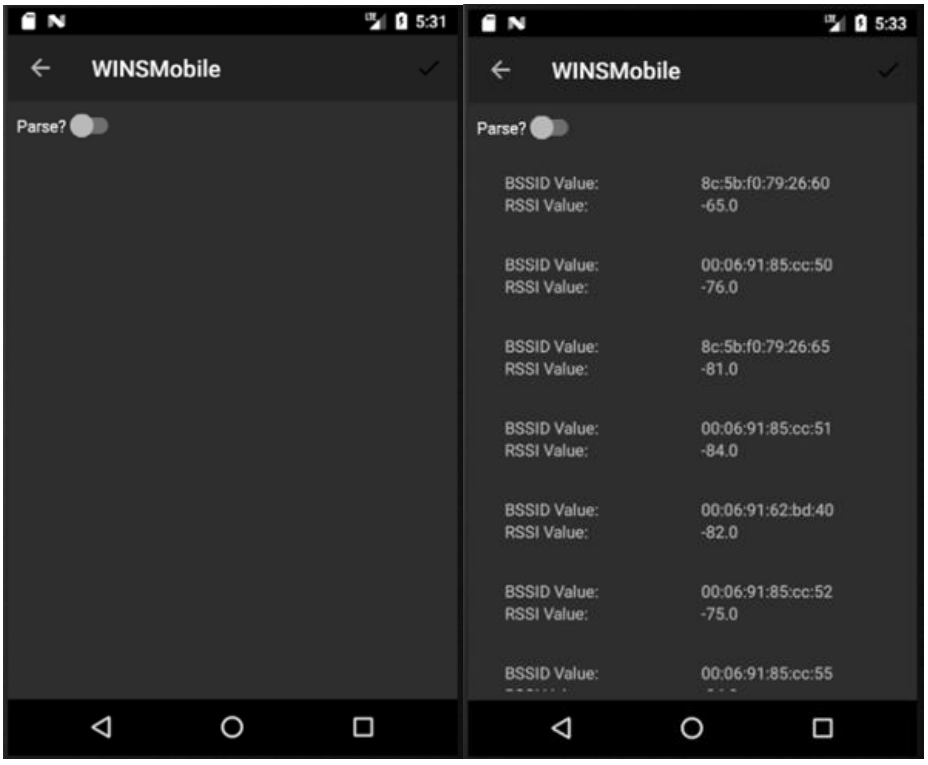


Figura 4.13: Menu de calibração de ponto de controlo. a) Pré-registo. b) Pós registo.

Funcionamento

A *Activity* que lança esta interação é a *ControlPointsMapActitviy*, que de forma semelhante à *MainActivity* apresenta os dados do sistema WINS, sendo que para além de obter dados, também os cria, nomeadamente os dados referentes aos RSSI dos diversos pontos de controlo. Para tal, a *Activity* realiza inicialmente a apresentação do Mapa, e também as salas. Tal sucede somente pelo facto de o utilizador que esteja a realizar a calibração do modelo necessita de ter algum ponto de referência do espaço físico em que se encontra.

A Figura 4.14 mostra o funcionamento desta *Activity*. Em primeiro inicia o Mapa com o método *buildMap*. Quando este acaba de ser processado e é apresentado ao utilizador o método *onMapReady* é chamado pelo SO *Android*, onde por sua vez é chamado o método *populateMapWithData.* Como estamos perante um paradigma assíncrono e é necessário obter dados do servidor é instanciada uma classe que efetua esta chamada ao servidor para obter os dados das salas e dos pontos de controlo. Quando esta classe obtém uma resposta do servidor chama o método *addControlPointsToMap*. Este método cria um *listener* (algo que fica à escuta de uma interação do utilizador para ser chamado) que quando um *ControlPoint* identificado no mapa é pressionado é executada a *Activity ControlPointsActivity*.

Esta por sua vez capta os diversos RSSI para os diversos AP e envia estes dados ao servidor. Aquando do término deste processamento a *Activity* é fechada e o utilizador volta a visionar o mapa podendo calibrar outro ponto de controlo ou voltar para a *Activity* principal.

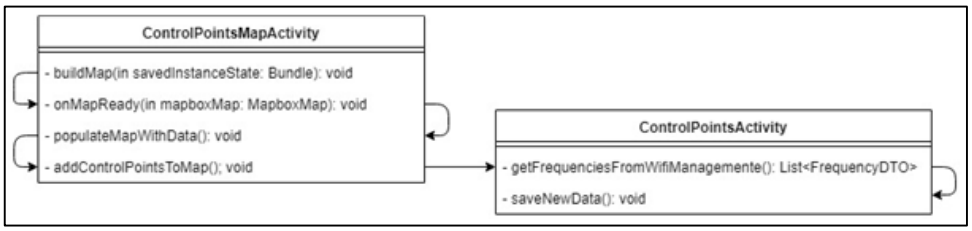


Figura 4.14: Funcionamento do modo de calibração.

4.2.3. Navegação

No modo de navegação o WINS todas as componentes do WINS interligam-se e providenciam a principal funcionalidade do sistema, que consiste em providenciar um sistema de navegação com a utilização de Wi-Fi para obter a posição do utilizador.

Para aceder a este modo o utilizador têm de indicar qual o sítio para onde pretende ir. Para tal basta pressionar o botão "*Navigate* To" na *side-bar* conforme mostrado na Figura 4.15.

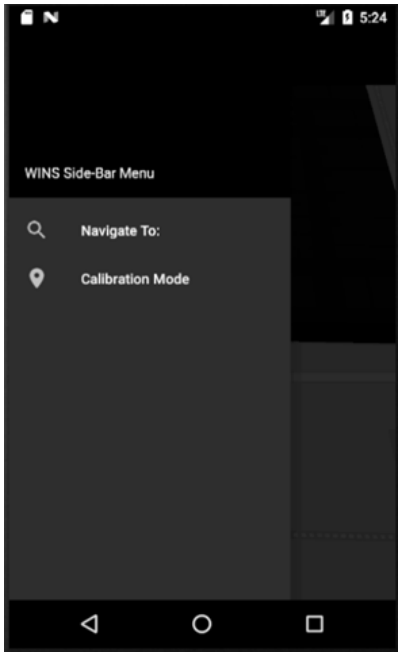


Figura 4.15: Menu para aceder ao modo de navegação.

Ao pressionar neste botão o utilizador é automaticamente enviado para o menu de procura de pontos de interesse, conforme apresentado na Figura 4.16.

Este menu consiste na lista de pontos de interesse carregados no sistema WINS. Embora sejam apenas salas, estas são entidades geográficas que providenciam uma informação ao utilizador ao qual este consegue associar a um contexto real. De modo a que o utilizador possa procurar de uma maneira eficiente também existe um *Search* ícone que permite ao utilizador procurar por um ponto de interesse tendo por base uma caixa de texto.

Figura 4.16: Menu com pontos de interesse.

Ao pressionar numa sala a aplicação entra em modo de navegação e disponibiliza ao utilizador um caminho que este deverá percorrer, tendo em conta a sua localização corrente e o destino pretendido, conforme apresentado na Figura 4.17. A rota encontra-se identificada como uma linha e o destino escolhido pelo utilizador fica com a mesma cor e uma saliência de modo a que esta perceba qual a sala solicitada.

À medida que o utilizador se desloca sobre este caminho, este vai sendo atualizado mostrando assim o caminho que o utilizador já percorreu.

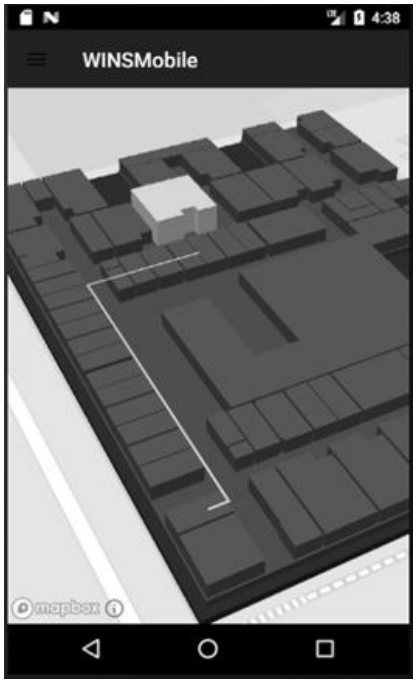


Figura 4.17: Modo navegação do cliente WINS.

Funcionamento

Embora fosse de esperar que esta componente fosse a mais complexa de implementar, tal não acontece devido ao facto de toda a complexidade se encontrar no servidor do WINS. Assim esta componente de modo a funcionar apenas tem de articular pedidos entre os serviços disponibilizados no contexto de *navigation* e apresentar os dados que o servidor devolve já que são dados com um formato *GeoJSON*.

De acordo com o catálogo de serviços disponibilizados na secção 4.1.5.2 de modo a obter um caminho é utilizado o serviço responsável para esta funcionalidade é: *.../navigation/paths*, com dois parâmetros: *controlPointId* e *pointOfInterestId*, que correspondem ao início e ao destino do caminho a percorrer.

O destino já se encontra definido, já que corresponde ao ponto de interesse que um utilizador pressiona na lista de pontos de interesses quando solicita um caminho. O início é obtido através do serviço *.../wifi-indoor-navigation-system/api/v1/navigation/paths?controlPointId={id}&pointOfInterestId={id}*. Este devolve o ponto de controlo tendo por base os RSSI providenciados.

Assim, o dispositivo cliente apenas tem de articular estas chamadas e à medida que *currentControlPointId* é alterado é pedido um novo caminho, que na realidade corresponde à deslocação do utilizador na grelha previamente calibrada e correspondente aos caminhos navegáveis.

A *Activiy* que efetua esta gestão é na realidade a *MainActivity*, já que é esta que tem o mapa inicial.

Esta apenas começa a processar os dados dos RSSI no seu redor para obter a sua localização quando obtém um ponto de interesse como destino. Para tal é necessário a *SearchActivity* providenciar o destino, conforme apresentado na Figura 4.18.

O que sucede é que ao terminar a *SearchActivity* o método *onActivityResult* da *MainActivity* é chamado pelo SO *Android*. Neste momento, este método verifique se foi identificado um ponto de interesse, já que o utilizador não é obrigado a selecionar um destino – pode simplesmente voltar para trás.

Se for detetado um ponto de interesse a aplicação assume que o utilizador está a solicitar um caminho e desse modo entra em modo de navegação.

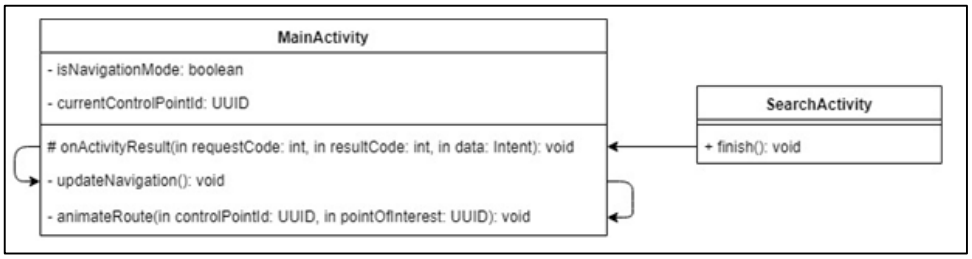


Figura 4.18: Funcionamento do modo de navegação.

Este modo consiste no envio sistemático dos RSSI dos diversos AP que se encontram no seu redor. Caso o ID do ponto de controlo obtido do servidor seja diferente do *currentControlPointId* é chamado o *updateNavigation* fazendo com que a rota que está a ser apresentado ao utilizador seja atualizada, dando a sensação de que o utilizador se encontra a percorrer a rota solicitada.

4.3. Resultados Obtidos

Conforme explicado ao longo deste documento, o WINS é um LBS que providencia a localização do utilizador através de *fingerpriting*. Assim sendo, a fase de calibração é indispensável para o sistema poder providenciar qualquer tipo de informação com um contexto geográfico em relação à posição do utilizador da aplicação. Esta seção pretende demonstrar de que modo a fase de calibração pode afetar o posicionamento do utilizador.

Em teoria, quanto maior o número de pontos de controlo calibrados, melhor será a precisão de posicionamento do WINS. Mas devido ao facto de a aplicação cliente não controlar de que modo o dispositivo se desloca no espaço, por exemplo, utilizando técnicas de *dead reckoning*, os valores de RSSI poderão não ser suficientes já que estes são algo voláteis e suscetíveis a fatores físicos relacionados com espaço e tempo. Não obstante este facto, e tendo em conta que este projeto apenas aborda o

posicionamento de um utilizador com uso do RSSI, são equacionados três ensaios para provar esta concetualização.

A diferença de ensaio para ensaio consiste numa fase de calibração diferente, onde de ensaio para ensaio, é dado um valor de "salto" diferente. Um salto, neste contexto, é considerado literalmente o salto entre cada ponto de controlo. Ou seja, sendo que cada ponto de controlo corresponde a um vértice de uma estrutura de um grafo, ao existir um salto de Nível 1 significa que existe um ponto de controlo que é calibrado, mas não existe a calibração dos primeiros pontos adjacentes. Num de Nível 2 é a mesma lógica, mas onde existe um ponto de controlo que é calibrado, não são calibrados os segundos pontos adjacentes, e por aí adiante conforme o aumento do nível.

Deste modo o WINS ao providenciar a localização de um utilizador terá menos pontos para posicionar o utilizador, conforme o aumento do nível, tornando o sistema de voto mais simples, mas menos preciso, já que existem menos localizações possíveis para colocar o utilizador da aplicação.

Assim, os ensaios podem ser classificados da seguinte maneira:

Ensaio Nível 2 – Este cenário é feito uma abordagem em que apenas são calibrados pontos de controlo com saltos de Nível 2, ou seja, a cada 3 pontos de controlo apenas 1 é calibrado.

Ensaio Nível 1 – Em segundo é realizada uma calibração de Nível 1, onde a cada 2 pontos de controlo 1 é calibrado.

Ensaio Nível 0 – Por último é realizado uma calibração de Nível 0, onde todos os pontos de controlo são calibrados.

Relativamente às condições destes ensaios, todos foram realizados numa sexta-feira à tarde, pós o horário de trabalho, o que torna a área menos propicia a ter interferências de dispositivos de outras pessoas na área de estudo.

4.3.1. Ensaio Nível 2

Este ensaio compreende uma calibração muito ponderada, onde com facilidade será dada a posição do utilizador, embora com uma precisão muito baixa, e muito aquém de qualquer LBS *indoor*. Para todos os efeitos é o cenário mais simples de validar.

Conforme é possível observar na secção dos anexos Nível 2, o WINS consegue providenciar a precisão necessária e o utilizador consegue de facto navegar na rede e chegar ao seu destino. Mas em termos práticos de usabilidade, simplesmente não funciona. Como a aplicação só atualiza a cada 3 pontos de controlo, que corresponde sensivelmente a cada 15 metros, o utilizador obtém uma experiência muito negativa, onde por vezes quase se pondera se a aplicação deixou de funcionar, e de um momento para o outro "salta" para a posição onde o utilizador se encontra. Pior é o caso em que um utilizador que

continue a andar, por vezes, quando a aplicação atualiza a posição, a posição encontrada é de 3 a 5 metros atrás, ou à frente.

Foi também observado que por vezes a posição do utilizador estava totalmente fora do contexto, colocando o utilizador numa localização totalmente dispare de onde este se encontrava. Isto acontece devido ao facto de os RSSI providenciados ao WINS corresponderem a uma assinatura de um ponto de controlo que não é o correto, o WINS acaba por interpretar a localização do utilizador como aquela.

4.3.2. Ensaio Nível 1

Neste ensaio a experiência do utilizador já é mais aceitável, embora não seja perfeita. Devido ao facto de a calibração realizada ser de Nível 1 o utilizador da aplicação já não sente que a aplicação deixou de funcionar. A experiência pode ser traduzida em algo em que a aplicação parece que está a pensar e depois obtém uma resposta. A secção dos anexos Nível 1 tenta demonstrar de que modo a aplicação realiza a apresentação ao utilizador, podendo verificar-se que ocasionalmente existe um retrocesso no caminho, quase como se o utilizador estivesse a andar para trás. De uma maneira geral, a precisão é superior, já que existem mais pontos de controlo calibrados o que permite ao WINS posicionar o utilizador mais perto da sua localização real.

Similarmente ao ensaio de Nível 2 foi também verificado que ocasionalmente o sistema colocava o utilizador numa posição totalmente dispare do que onde este se encontrava. De salientar que este comportamento ocorreu menos vezes do que as vezes verificadas no ensaio com uma calibração de Nível 2.

4.3.3. Ensaio Nível 0

Por último é realizado um ensaio que corresponde a um cenário onde é realizada a calibração de todos os pontos de controlo. Conforme seria de esperar a precisão do WINS aumenta, assim como a usabilidade por parte do utilizador.

Devido ao facto de existirem mais pontos de controlo calibrados a precisão do WINS é superior, já que consegue posicionar o utilizador no espaço físico com muito maior precisão. Isto traduz-se a uma experiência muito melhor do que a verificados nos outros ensaios. Tal acontece devido ao facto de o sistema estar constantemente a receber pedidos e a reposicionar o utilizador num ponto de controlo que se encontra mais perto da sua localização real, traduzindo-se para uma experiência onde o utilizador vê a aplicação a fluir e providenciar a sua localização corrente, embora sempre com um ligeiro atraso, já que o espaçamento de pontos de controlo é de sensivelmente 3 a 5 metros.

Como nota final, mais uma vez acontece o comportamento verificado nos outros ensaios, onde existem determinados momentos onde a aplicação diz que o utilizador se encontra numa localização totalmente dispare de onde este se encontra.

Capítulo 5 - Considerações Finais e Trabalhos Futuros

Com o avanço da tecnologia tem sido possível criar ambientes aplicacionais que permitem desenvolver aplicações que anteriormente pareciam inalcançáveis. O WINS recai neste tipo de ambientes, onde através da utilização de software é possível criar e implementar soluções alcançáveis por todos.

O objetivo deste sistema foi alcançado em particular o de criar um sistema que permita a um utilizador navegar num ambiente *indoor* recorrendo apenas à tecnologia Wi-Fi.

Pode-se afirmar que o sistema WINS é generalizável a outras áreas de interesse sendo necessário apenas inserir os dados correspondentes no sistema.

A disponibilização dos serviços num protocolo HTTP sobre um paradigma REST torna a sua integração fácil para outras aplicações, em particular em que usem dados no formato *GeoJSON*. (RFC-7946, 2019)

Contudo, o sistema apresenta algumas limitações e como tal são apresentadas melhorias e futuros trabalhos.

A primeira recai sobre o carregamento dos dados. Embora tenha sido desenvolvido um catálogo de serviços sobre uma API que permite inserir dados no sistema, este carregamento recai sobre a articulação de pedidos HTTP. Seria interessante desenvolver uma aplicação *Web* que em vez de consumir os dados, como é realizado pela aplicação cliente *Android* do WINS, os permitisse criar. Muito similarmente ao que já é realizado pela *Mapbox* © na aplicação *Studio* destes, onde é possível desenhar elementos geográficos de componentes como edifícios, andares e salas. é necessário também que o WINS ofereça este serviço, não só através das interfaces REST mas também através de uma interface gráfica simplificada para o utilizador final, podendo este desenhar a sua área de interesse, sendo que no caso em concreto seria o campus da Faculdade de Ciências da Universidade de Lisboa.

O segundo elemento recai sobre a componente de Segurança. A realidade é que esta não foi contemplada devido ao facto de não fazer parte do âmbito deste projeto. Neste momento qualquer pessoa que tenha acesso ao WINS facilmente consegue apagar todos os dados, bastando para isso chamar os serviços de DELETE que são expostos nas diversas interfaces REST. Este ponto é crucial resolver de modo a que este sistema possa crescer de uma forma resiliente.

A terceira está relacionada com o posicionamento baseado no Wi-Fi. É um facto que o posicionamento através do Wi-Fi foi concretizado, mas o modo como o WINS o realiza poderá não ser suficiente. Este ponto é crucial, porque embora o sistema de momento se encontra a funcionar com o uso do Wi-Fi, amanhã tal não poderá acontecer. Toda a aplicação foi desenvolvida sobre a premissa de que o serviço de posicionamento através de Wi-Fi está disponível. Assim o WINS está dependente da disponibilização deste serviço e caso a *Google* ©, que é a dona do SO *Android*, deseje remover este serviço do SO, o WINS deixa de funcionar já que não tem outros mecanismos que garantam o posicionamento. É por isso

que, embora a segmentação e calibração de dados realizada neste projeto seja aceitável, já que o foco principal não era uma precisão muita elevada, mas sim um mecanismo que permitisse a utilizador obter um caminho e navegar sobre este, é necessário adicionar outros mecanismos que permitam ao sistema ser mais resiliente no contexto de posicionamento, garantido que o utilizador consegue obter uma localização de onde se encontra, com a utilização de outras tecnologias que adicionem valor ao WINS.

Por último temos a própria arquitetura do WINS, onde não é tirado partido da sinergia possível entre as diversas componentes do sistema. Embora o WINS assuma que todo o posicionamento e navegação deva ser realizado pela componente do Servidor, tal poderá não ser a melhor abordagem, pelo menos da maneira tão explicita como foi implementada. Enquanto o princípio de separação de responsabilidades seja importante concetualizar e implementar em sistemas informáticos, por vezes é necessário adaptar as diversas componentes de um sistema de modo a tirarem o máximo partido de entre elas podendo criar sinergias entre si. Nos diversos ensaios realizados, esta falta de sinergia é clara, já que o utilizador da aplicação não consegue obter a melhor experiência devido aos saltos constantes à medida que o sistema descobre a nova posição do utilizador.

Em nota final, conforme mencionada ao longo do documento, o propósito deste projeto é providenciar uma aplicação onde sejam acrescentadas funcionalidades, desse modo o código fonte encontra-se disponibilizado nos seguintes links:

- Componente Cliente – https://bitbucket.org/andreAdro/ins_android/src/master/
- Componente Servidor – https://bitbucket.org/andreAdro/ins_java/src/master/

.

Referências Bibliográficas

Apache. (2019). Obtido de Apache: http://tomcat.apache.org/

Basiri, A., Lohan, E. S., Moore, T., Winstanley, A. C., Peltola, P., Hill, C., . . . Silva, P. F. (2017). Indoor location based services challenges, requirements and usability of current solutions. *Computer Science Review, 24*, 1-12. Obtido em 2 de 4 de 2019, de https://sciencedirect.com/science/article/pii/s1574013716301782

Beaconstac. (10 de 05 de 2019). *Beaconstac*. Obtido de Beaconstac: https://www.beaconstac.com/what-is-a-bluetooth-beacon

Beaconstac. (10 de 05 de 2019). *Beaconstac*. Obtido de Beaconstac: https://www.beaconstac.com/ibeacon-and-eddystone

Brislen, P. (25 de 8 de 2000). *Computer World*. Obtido de Computer World: https://www.computerworld.com.au/article/74325/wap_dead_water_says_australian_analyst/

Cangeloso, S. (25 de 3 de 2013). *Extremetech*. Obtido de Extremetech: https://www.extremetech.com/extreme/151068-forget-wifislam-bytelight-uses-leds-for-indoor-positioning

Carrera, J., Zhao, Z., Braun, T., Li, Z., & Neto, A. (2 de 2018). A real-time robust indoor tracking system in smartphones. *Computer Communications*, pp. 104-115.

Cisco. (2008). *Wi-Fi Location-Based Services 4.1 Design Guide.* San Jose: Cisco System, Inc. Obtido de Cisco: https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Mobility/WiFiLBS-DG/wifich2.html_

Codecademy. (16 de 08 de 2019). *Codecademy*. Obtido de Codecademy: https://www.codecademy.com/articles/what-is-rest

CRFS. (2019). *CRFS*. Obtido de CRFS: https://www.crfs.com/blog/how-accurate-tdoa-geolocation/

Davidson, R., Akiba, Cufi, C., & Townsend, K. (2014). Getting Started with Bluetooh Low Energy. Em R. Davidson, Akiba, C. Cufi, & K. Townsend, *Getting Started with Bluetooh Low Energy* (pp. 1-3). O'Reilly Media, Inc., 1005 Garvenstaring Highway North, Sebastopol, CA 95472.

Dogtiev, A. (11 de 5 de 2018). *Business Of Apps.* Obtido de http://www.businessofapps.com/data/app-revenues/

Emmanuel Bernard, S. P. (18 de 08 de 2019). *Beanvalidation*. Obtido de Beanvalidation: https://beanvalidation.org/1.0/spec/

Fireflylifi. (14 de 05 de 2019). *Fireflylifi*. Obtido de Fireflylifi: https://www.fireflylifi.com/visible-light-communications.html

Gibbs, S. (8 de 2 de 2015). *The Guardian.* Obtido de The Guardian: https://www.theguardian.com/technology/2015/feb/08/google-maps-10-anniversary-iphone-android-street-view

Google. (2019). *Activities.* Obtido de https://developer.android.com/reference/android/app/Activity.

Grozdanic, L. (22 de 10 de 2015). *Archipreneur*. Obtido de Archipreneur: https://archipreneur.com/wired-city-how-technology-is-remapping-the-urban-environment/

Haines, E. (25 de 3 de 2019). *wordstream*. Obtido de wordstream: https://www.wordstream.com/blog/ws/2018/10/04/beacon-technology

Higginbotham, J. (13 de 6 de 2018). *Nordicapis*. Obtido de Nordicapis: https://nordicapis.com/is-rest-still-a-relevant-api-style/

http://www.explainthatstuff.com/wirelessinternet.html. (2016). *www.explainthatstuff.com/wirelessinternet.html*. Obtido de www.explainthatstuff.com.

Infsoft. (10 de 05 de 2019). *Infsoft*. Obtido de Infsoft: https://www.infsoft.com/technology/sensors/bluetooth-low-energy-beacons

Khan, L. U. (5 de 2017). Visible light communication: applications, architecture, standardization and research challenges. *Digital Communications and Networks*, pp. 78-88.

Kim, J.-H., Kang, S.-Y., & Lee, S.-T. (2017). VLC-based location data transferal for smart devices. *Optical Switching and Networking*, pp. 250-258.

Kontakt. (10 de 05 de 2019). *Kontakt*. Obtido de Kontakt: https://kontakt.io/beacon-basics/what-is-a-beacon/

Lomas, N. (17 de 05 de 2017). *Tech Crunch.* Obtido de Tech Crunch: https://techcrunch.com/2017/05/17/google-has-an-indoor-positioning-tech-in-the-works-called-vps/

Lomas, N. (17 de 05 de 2017). *techcrunch*. Obtido de techcrunch.com: https://techcrunch.com/2017/05/17/google-has-an-indoor-positioning-tech-in-the-works-called-vps/

mapbox. (01 de 09 de 2019). *mapbox*. Obtido de mapbso: https://docs.mapbox.com/android/maps/overview/

Markets, R. a. (20 de 12 de 2018). *Indoor Positioning System Market - Forecasts From 2018 to 2023.* Knowledge Sourcing Intelligence LLP. Obtido de Business Wire: https://www.researchandmarkets.com/research/k7kpbt/the_global_indoor?w=4

Microsoft. (18 de 08 de 2019). *Microsoft*. Obtido de Microsoft: https://docs.microsoft.com/en-us/previous-versions/aspnet/dd381412(v=vs.108)

Microsoft. (19 de 08 de 2019). *Microsoft*. Obtido de Microsoft: https://docs.microsoft.com/en-us/iis-administration/api/crud

Moshe, D. (19 de 1 de 2019). *Business Insider*. Obtido de Business Insider: https://www.businessinsider.com/how-fast-speed-light-travels-earth-moon-mars-nasa-2019-1

MySQL. (24 de 09 de 2019). *MySQL*. Obtido de MySQL: https://www.mysql.com/

Oracle. (16 de 08 de 2019). Obtido de Oracle: https://www.oracle.com/technetwork/java/index.html

Oracle. (18 de 08 de 2019). *Oracle*. Obtido de Oracle: https://www.oracle.com/technetwork/java/javaee/jta/index.html

Oracle. (18 de 08 de 2019). *Oracle*. Obtido de Oracle: https://docs.oracle.com/javaee/6/tutorial/doc/bnbpz.html

Paraschiv, E. (18 de 08 de 2019). *Baeldung*. Obtido de Baeldung: https://www.baeldung.com/java-dijkstra

Pichler, S. (28 de 01 de 2019). *indoors.* Obtido de indoors: https://indoo.rs/indoor-positioning-with-wifi/

Pivotal. (18 de 08 de 2019). *Spring*. Obtido de Spring: https://spring.io/

Premack, R. (10 de 6 de 2018). *Business Insider.* Obtido de Business Insider: https://www.businessinsider.com/it-jobs-information-technology-2018-5

Pullen, J. P. (24 de 4 de 2015). *Time*. Obtido de Time: http://time.com/3834259/wifi-how-works/

RFC-2616. (16 de 08 de 2019). Obtido de Internet Engineering Task Force: https://tools.ietf.org/html/rfc2616#section-1.1

RFC-6749. (17 de 18 de 2019). *Internet Engineering Task Force*. Obtido de Internet Engineering Task Force: https://tools.ietf.org/html/rfc6749

RFC-7946. (16 de 08 de 2019). *Internet Engineering Task Force*. Obtido de Internet Engineering Task Force: https://tools.ietf.org/html/rfc7946

Rouse, M. (03 de 2010). *Search Mobile Computing*. Obtido de Search Mobile Computing: https://searchmobilecomputing.techtarget.com/definition/Cell-of-Origin

Rouse, M. (21 de 9 de 2018). *Search Networking*. Obtido de Search Networking: https://searchnetworking.techtarget.com/definition/Ethernet

Schiller, J., & Voisard, A. (2004). Location Based Services. San Francisco: Morgan Kaufmann Publishers Inc.

Srivastava, S. (5 de 12 de 2018). *Leverege.* Obtido de Leverege: https://www.leverege.com/blogpost/wifi-indoor-positioning

Swagger. (17 de 08 de 2019). *Swagger*. Obtido de Swagger: https://petstore.swagger.io/#/

Swagger. (17 de 08 de 2019). *Swagger*. Obtido de Swagger: https://swagger.io/

Techopedia. (16 de 08 de 2019). Obtido de Techopedia: https://www.techopedia.com/definition/432/application-server

Turgut, Z., Aydin, G., & Sertbas, A. (2016). Indoor Localization Techniques for Smart Building Environment. *Elsevier*, pp. 1176-1181.

UCI. (05 de 08 de 2019). Obtido de Donal Bren School of Information & Computer Sciences: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Verma, A. (18 de 8 de 2017). *Foss Bytes*. Obtido de Foss Bytes: https://fossbytes.com/wifi-beacons-better-location-based-services-lbs/

Wang, Y., Yue, L., & Xu, Y. (2013). Bluetooth positioning using RSSI and triangulation methods. *Processing of the 2013 IEEE 10th Consumer Communications and Networking Conference (CCnC*, (pp. 837-842). Las Vegas.

Woodford, C. (5 de 7 de 2018). *Explain that stuff*. Obtido de Explain that stuff: https://www.explainthatstuff.com/howbluetoothworks.html

Woodford, C. (22 de 5 de 2018). *Explain that Stuff*. Obtido de Explain that Stuf: http://www.explainthatstuff.com/wirelessinternet.html

Zahradnik. (19 de 8 de 2018). *Life Wire*. Obtido de Life Wire: https://www.lifewire.com/wifi-positioning-system-1683343

Zheng, Z., Liang, L., Lei, Z., & Yong, F. (2 de 10 de 2018). Indoor Positioning Based on Blueotth Low-Energy Beacons Adopting Graph Optimization. *sensors*.

Zhuang, Y., Syed, Z., Georgy, J., & El-Sheimy, N. (11 de 2 de 2015). Autonomous smartphone-based WiFi positioning system by using acces points localization and crowdsourcing. *Elsevier*.

Anexos

Entidades *GeoJson*

Point

```
{
  "id": "c0a80107-6d0e-1b28-816d-0e2b46fd018c",
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [
      -9.157159029999946,
      38.75679878300008,
      0
    ]
  },
  "properties": {
    "name": "point name"
  }
}
```

LineString

```
{
  "type": "Feature",
  "geometry": {
    "type": "LineString",
    "coordinates": [
      [
        -9.157255499999962,
        38.75685043400006,
        0
      ],
      [
        -9.15676028799993,
        38.757002973000056,
        0
      ]
    ]
  },
  "properties": {}
}
```

Polygon


```
{
  "id": "c0a80107-6d0e-1b28-816d-0e2b46b70160",
  "type": "Feature",
  "geometry": {
    "type": "Polygon",
    "coordinates": [
      [
        [
          -9.156673627999965,
          38.75711557500006,
          0
        ],
        [
          -9.156705408999926,
          38.757105790000026,
          0
        ],
        [
          -9.156932414999972,
          38.757556986000054,
          0
        ],
        [
          -9.156673627999965,
          38.75711557500006,
          0
        ]
      ]
    ]
  },
  "properties": {
    "name": "polygon name"
 }
}
```


ControlPoint


```
{
  "id": "id correspondnete",
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [
      -9.157209817999956,
      38.75686450000006,
      0
    ]
  },
  "properties": {
    "name": "Nome do ponto de controlo"
  }
}
```

Entidades *JSON*

PointOfInterest


```
{
  "identifier": "c0a80107-6d0e-1b28-816d-0e2b46fd018c",
  "updatedDate": "2019-09-07T23:59:40.296Z",
  "createdDate": "2019-09-07T23:59:40.296Z",
  "name": "8.1.38"
}
```


Room

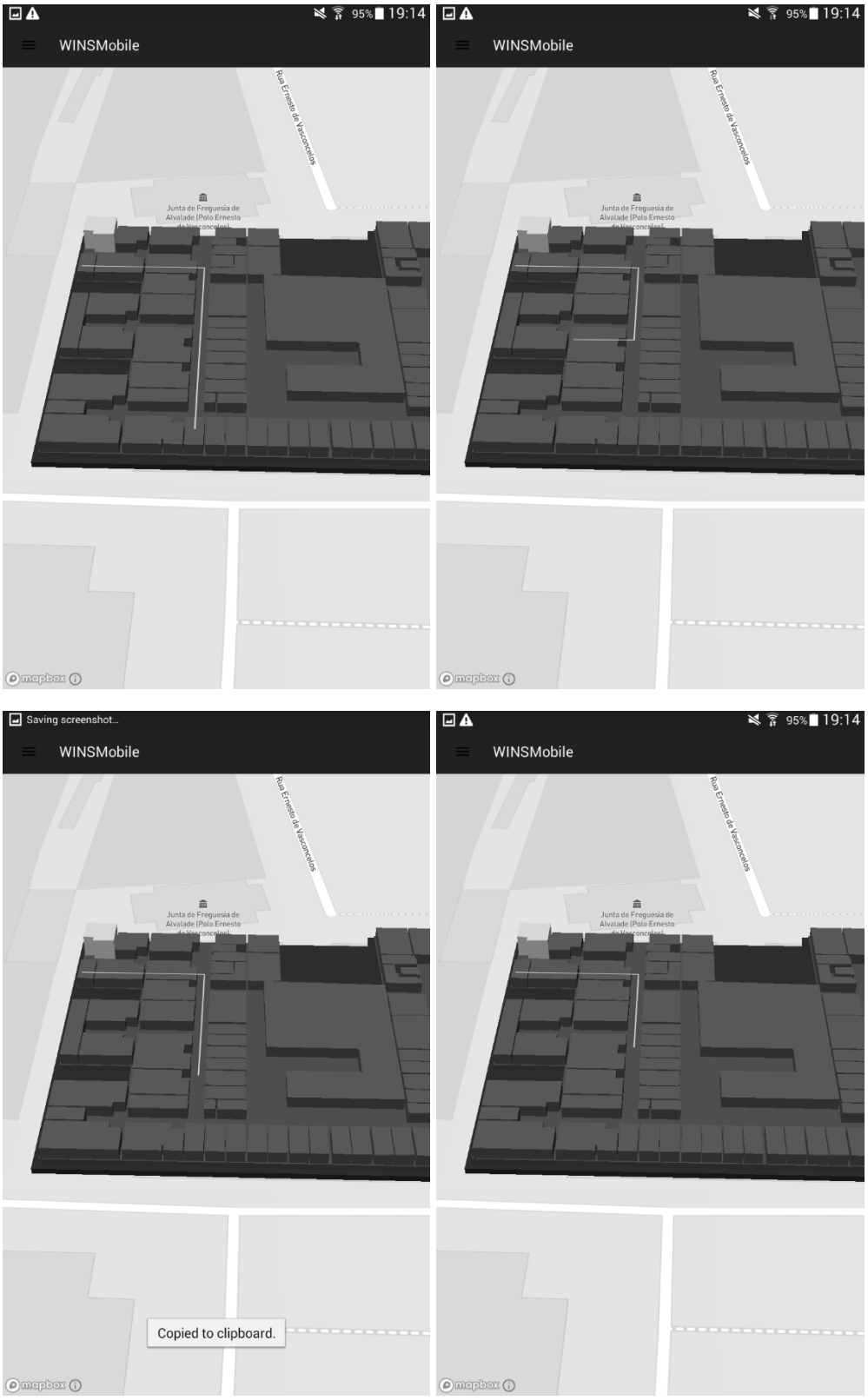

```
{
  "identifier": "c0a80107-6d0e-1b28-816d-0e2b46fc017c",
  "updatedDate": "2019-09-07T23:59:40.285Z",
  "createdDate": "2019-09-07T23:59:40.285Z",
  "name": "8.1.38"
}
```

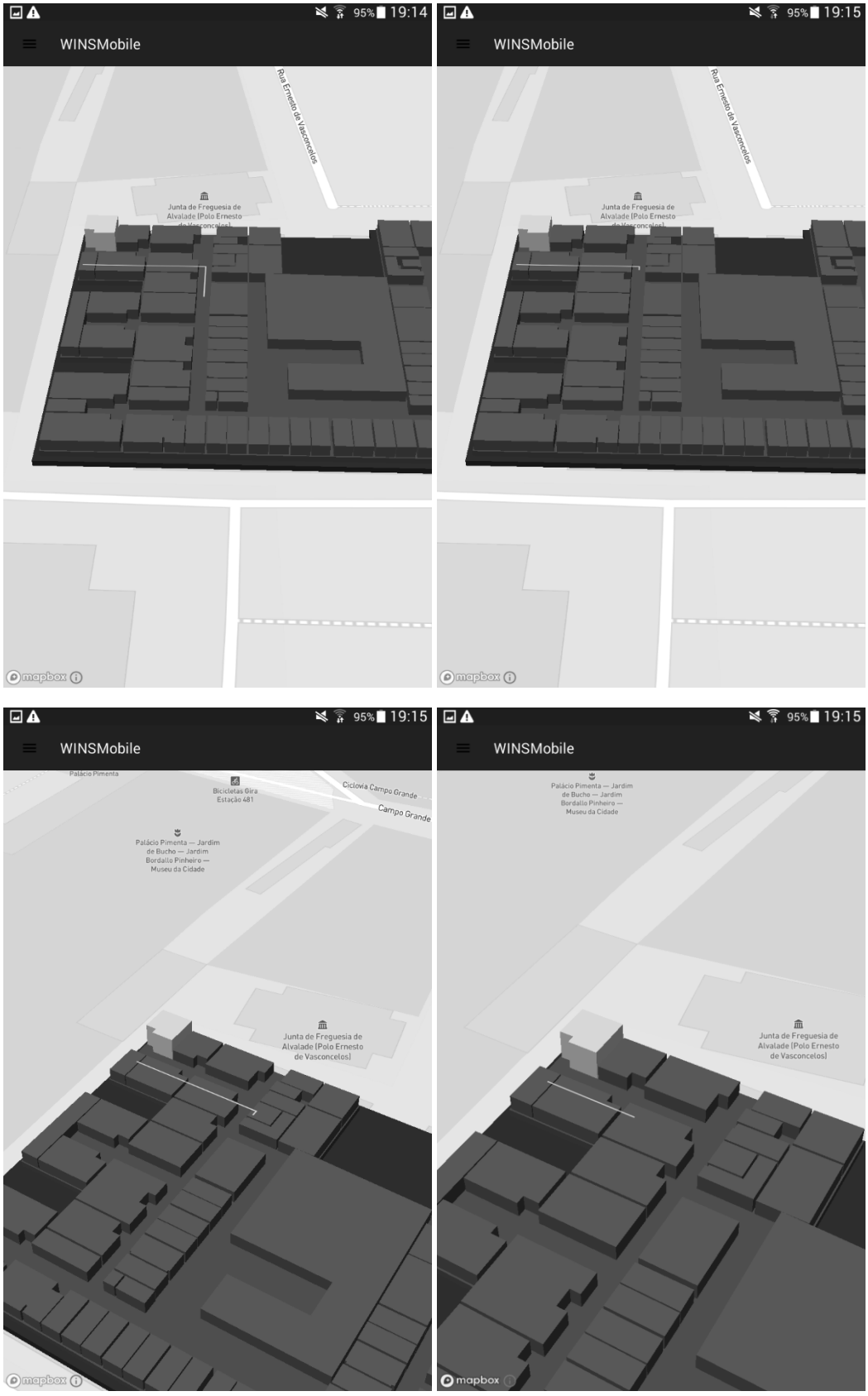

Frequency


```
{
  "identifier": "c0a80107-6d0e-1b28-816d-0e2bd4ec05ce",
  "updatedDate": "2019-09-08T00:00:16.620Z",
  "createdDate": "2019-09-08T00:00:16.620Z",
  "rssi": -56,
  "ssid": "NOS-2660",
  "bssid": "8c:5b:f0:79:26:60"
}
```

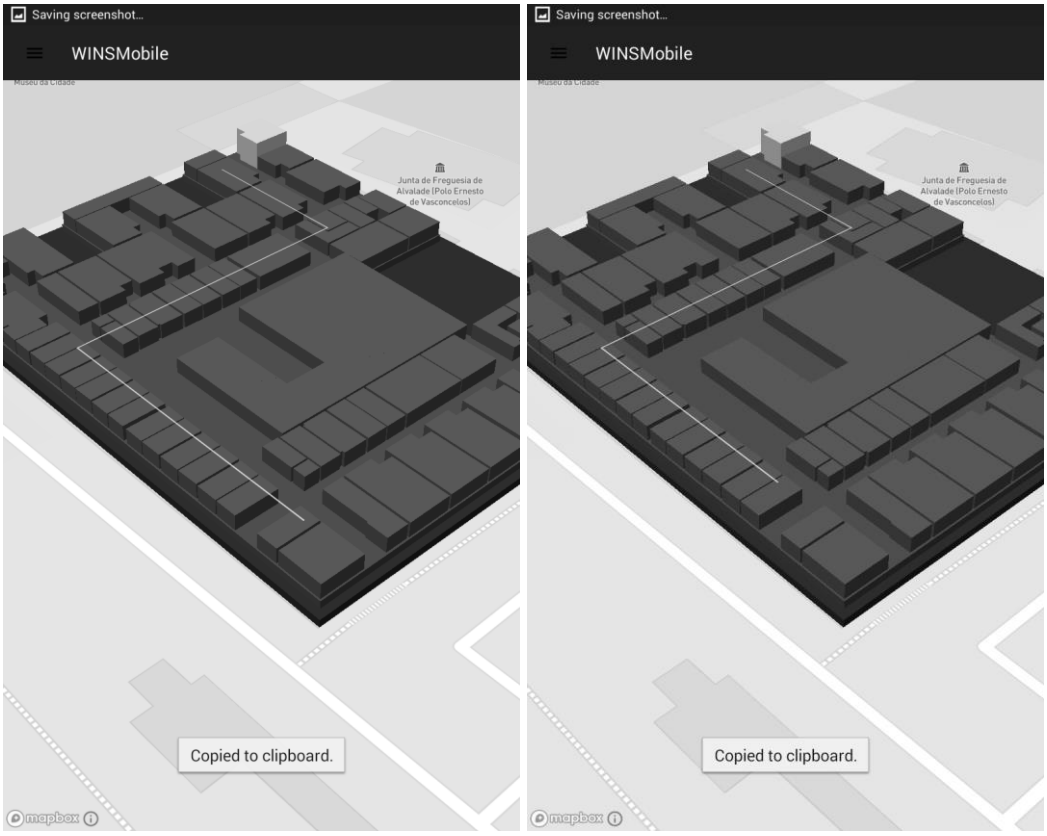
Ensaios

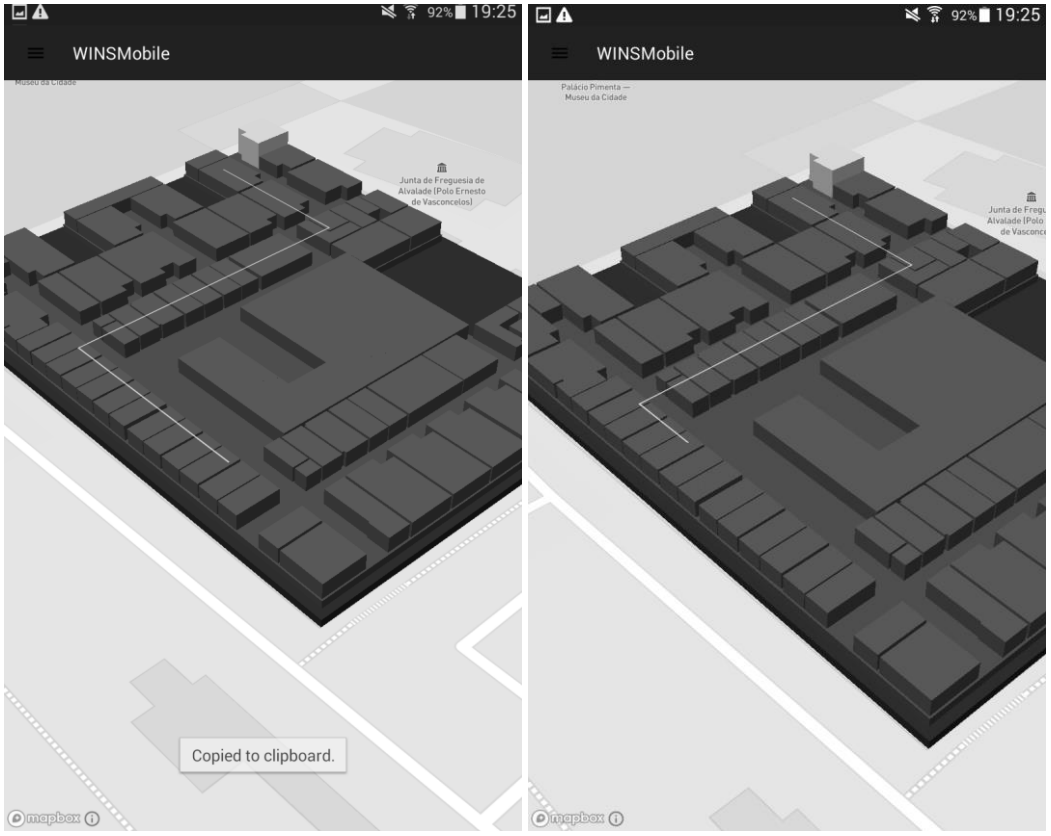
Nível 2

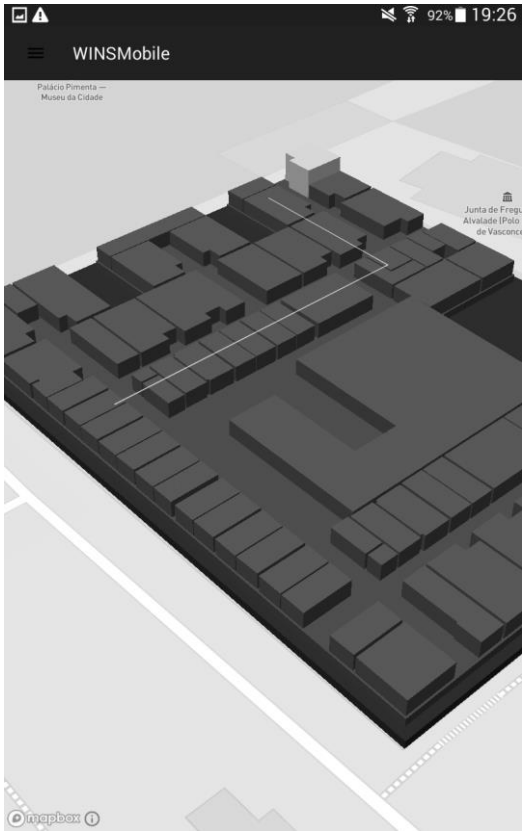



95% 19:14
WINSMobile
Rua Ernesto de Vasconcelos
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox
95% 19:15
WINSMobile
Rua Ernesto de Vasconcelos
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox
95% 19:15
WINSMobile
Palácio Pimenta
Bicicletas Gira Estação 481
Ciclovia Campo Grande
Campo Grande
Palácio Pimenta — Jardim de Bucho — Jardim Bordallo Pinheiro — Museu da Cidade
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox
95% 19:15
WINSMobile
Palácio Pimenta — Jardim de Bucho — Jardim Bordallo Pinheiro — Museu da Cidade
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox

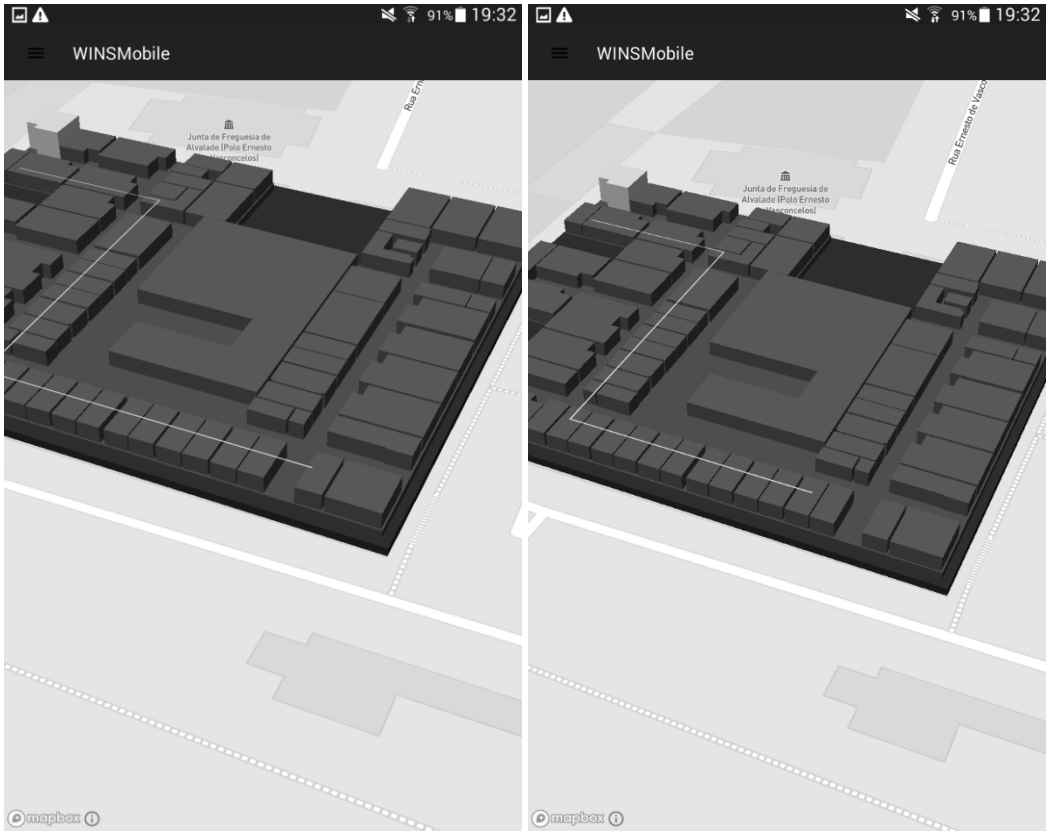
Nível 1

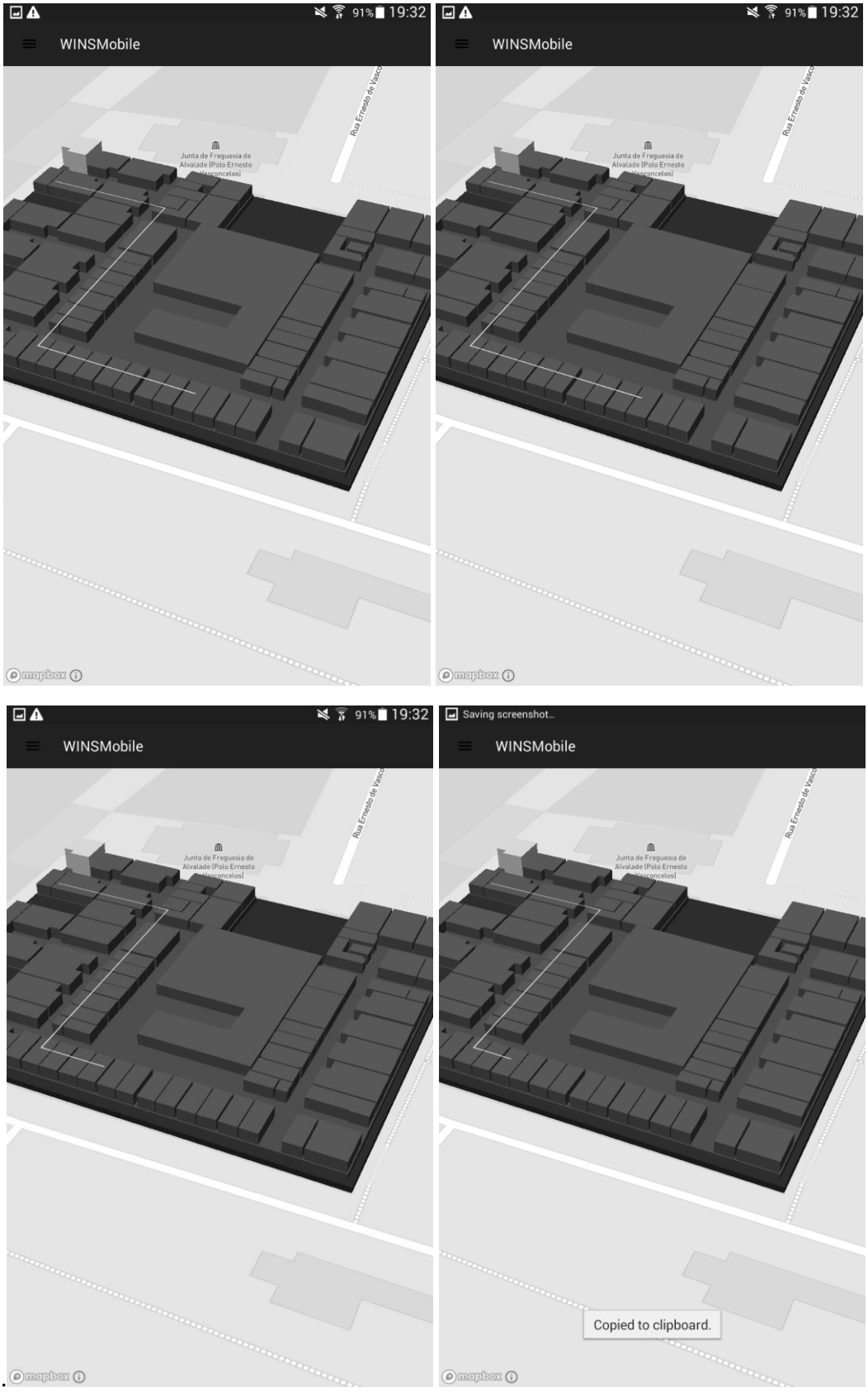

Saving screenshot...
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
Copied to clipboard.
Saving screenshot...
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
Copied to clipboard.

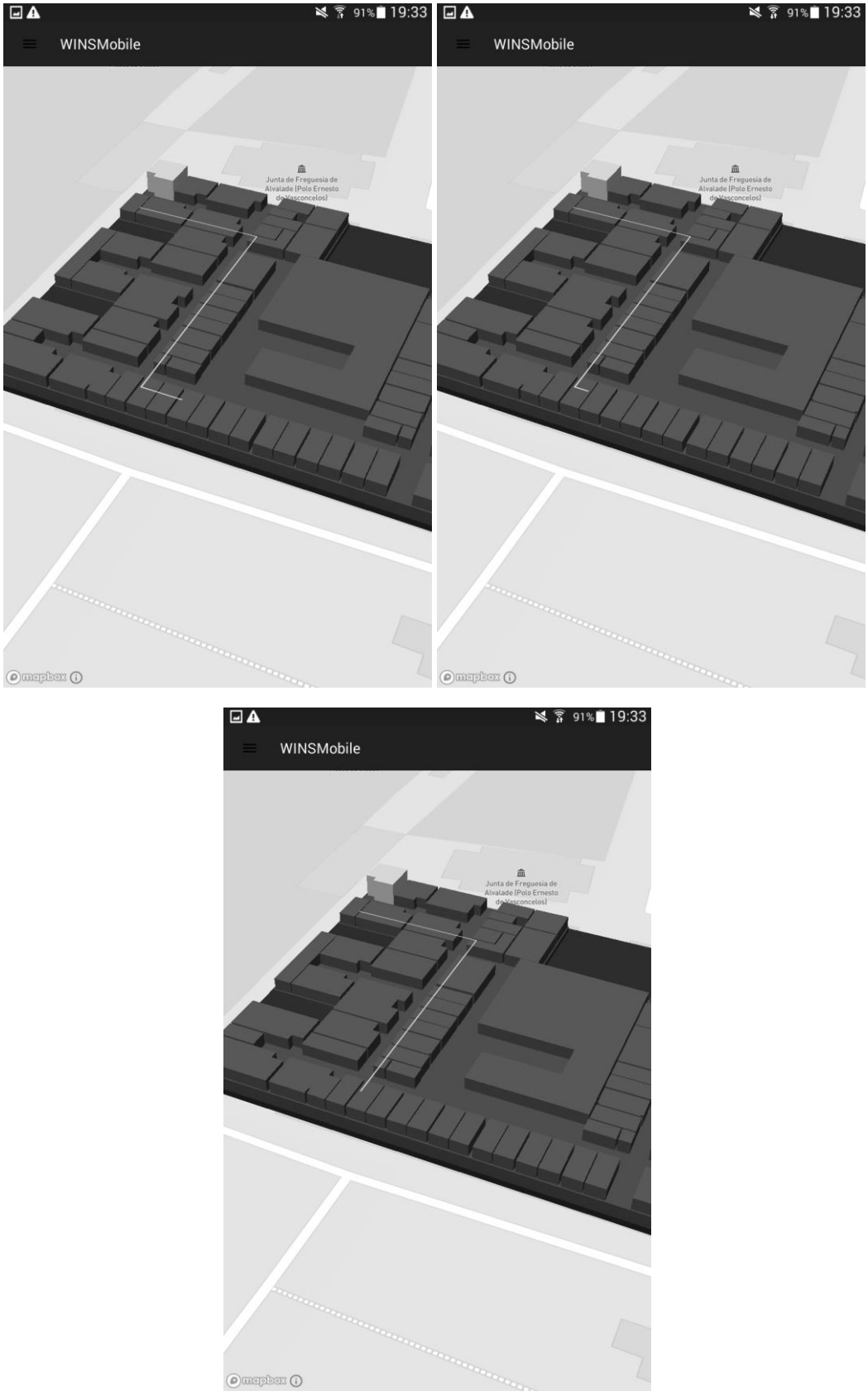

92% 19:25
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
Copied to clipboard.
92% 19:25
WINSMobile
Palácio Pimenta — Museu da Cidade


92% 19:26
WINSMobile
Palácio Pimenta — Museu da Cidade
mapbox

Nível 0


91% 19:32
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
mapbox
91% 19:32
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
mapbox


91% 19:32
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
Rua Ernesto de Vasc
mapbox
91% 19:32
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
Rua Ernesto de Vasc
mapbox
91% 19:32
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
Rua Ernesto de Vasc
mapbox
Saving screenshot...
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto
Rua Ernesto de Vasc
Copied to clipboard.
mapbox


91% 19:33
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox
91% 19:33
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox
91% 19:33
WINSMobile
Junta de Freguesia de Alvalade (Polo Ernesto de Vasconcelos)
mapbox