FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

MoviMetro

Rui Guilherme Silva Barros

VERSÃO DE TRABALHO

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientadora: Professora Maria Teresa de Andrade

Co-orientadora: Engenheira Raquel Vieira

27 de Junho de 2016

Resumo

A necessidade de proporcionar a um utilizador dos transportes públicos, nomeadamente o Metro do Porto, fácil acesso à informação de circulação, em tempo real, através de aplicativos móveis é algo essencial nos dias de hoje. Juntando isso à utilização de uma interface gráfica que forneça ao utilizador simplicidade e um ambiente amigável, torna evidente o motivo do desenvolvimento deste projeto.

O presente trabalho, proposto pela empresa EFACEC, foca-se no desenvolvimento de uma aplicação móvel para o Metro do Porto, mas com a premissa de ser facilmente adaptável a outras redes de metropolitano, que disponibilize ao utilizador a informação de circulação, em tempo real, assim como outras funcionalidades de uma forma bastante amigável para o utilizador destes meios de transporte.

A linguagem escolhida foi a linguagem Android nativo. A razão desta escolha prende-se com o facto das API's necessárias, tais como o Google Maps, sejam utilizadas apenas pela linguagem Android também por ser este o sistema móvel mais utilizado no presente. A opção pela adaptação da aplicação ao sistema de IOS ou multi-plataforma, obrigava a custos que a empresa não pretendia ter neste momento, sendo que essa hipótese ficou logo à partida excluída.

Numa fase inicial, realizou-se um estudo aprofundado sobre a Base de Dados utilizada, assim como o Web Server Apache e Android. Após esse estudo, foi construido um Sistema que fosse capaz de fazer a comunicação entre estes três elementos de forma simples mas eficaz.

Após a realização do projeto, foram encontradas algumas limitações ao nível do Web Server, sendo este de baixo e médio uso. Isso implica que para uma elevada frequência de utilizadores, poderá haver conflitos que bloqueiem o sistema. Tendo em conta este constrangimento e tendo em vista a resolução e adequação deste problema, foi feito um estudo para realização de trabalho futuro.

Relativamente aos objetivos principais, sendo estes a disponibilização da informação de circulação, em tempo real, criação de favoritos onde é possível guardar estações, calcular a distância entre a posição GPS atual e a estação de metro mais próxima, e de uma interface gráfica bastante acessível e com fácil usabilidade para o utilizador, foram cumpridos tal como era expectável nas ideias iniciais.

Abstract

The necessity to provide public transports users, namely Metro of Porto, an easy access to real time information through mobile applications is something essential nowadays. Gathering this with the use of a graphical interface which provides simplicity and an user-friendly environment to the user, clarifies the importance of this project development.

The present work, proposed by EFACEC Company, focuses on developing a mobile application for Metro of Porto, but with the premise of being easily adaptable to other metropolitan networks, providing information to the user, in real time, as well as others features, in a very friendly way, for the user of these transports.

The chosen programming language was native Android. The reasons for this choice are related to the fact that the API's needed, such as Google Maps, are used by Android only and also because this is the most widely used mobile system. The option to use the IOS system or multiplatform would require some costs that the company did not intend to have ate this moment, being this hypothesis excluded from the outset.

Initially, it was held an in-depth study about the database used, as well as the Web Server Apache and Android. After this study, was developed a System capable of making the communication between these three elements simple but effective.

After the accomplishments of the project, some limitations about the Web Server were found, being this for lower and medium use. This implies that for a big amount of users, there could exist conflicts that will block the system. Taking into account this constraint and having in view the resolution and suitability of this problem, it was made a study about future work.

Regarding the main goals, being these the tracking of the locations in real time, the creation of favorites where stations can be saved, the calculation of the distance between current position and the nearest station, and a graphic interface very easily understandable with easy usability to the user, were accomplished as it was expected in the initial ideas.

Agradecimentos

Ao longo da elaboração deste projeto tive a possibilidade de contar com várias ajudas, tanto a nível organizacional como no desenvolvimento deste projeto. Como tal, gostaria de demonstrar a minha gratidão por essa ajuda e pelo tempo que as pessoas dispensaram.

Antes de mais gostaria de agradecer a toda a minha família, tanto os meus pais como o meu irmão por me terem sempre apoiado e terem sido os meus pilares nesta fase importante da minha vida. Referindo, como principal pilar, o meu pai Manuel Barros sendo, sem dúvida, uma pessoa que sempre esteve presente e demonstrou um enorme apoio nas minhas decisões, orientando-me sempre para a escolha de um melhor caminho. Neste agradecimento incluo também a minha madrinha Glória e ao meu primo Rui Dinis pela apoio emocional que me deram.

Quero também prestar o meu maior agradecimento à minha orientadora, Professora Maria Teresa de Andrade, docente na Faculdade de Engenharia da Universidade do Porto e membro do centro de investigação INESC, por ter demonstrado sempre disponibilidade na ajuda para ultrapassar dificuldades, e no apoio sempre incondicional.

Um enorme obrigado à minha orientadora da empresa EFACEC, Eng.ª Raquel Vieira, por todo o tempo dispensado, toda a informação fornecida e empenho na resolução dos diversos problemas encontrados ao longo do projeto. Agradeço ainda ao Engº Paulo Paixão o acolhimento e as condições excelentes de trabalho, que me permitiram o desenvolvimento do projeto de forma relevante e entusiástica, e ao Engº Bruno Gil pelo apoio e disponibilidade proporcionada.

Por fim, deixo um agradecimento especial a todos os meus amigos que me foram acompanhando durante a minha vida académica, ajudando-me a ultrapassar os momentos mais complicados. Nestes amigos incluo o Nuno Pereira, o Sérgio Moreira, o Miguel Moreira, o Pedro Sonié e o André Vitorino, para eles o meu sentido agradecimento por todos os momentos vividos e partilhados ao longo do nosso percurso académico, momentos esses, que ficarão para sempre guardados na minha memória.

"A mente que se abre a uma nova ideia,
jamais volta ao seu tamanho original."

Albert Einstein

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

SDK	Software Development Kit
WM	Windows Mobile
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol address
WWW	*World Wide Web*
IIS	Internet Information Services
FTP	File Transfer Protocol
URL	Uniform Resource Locator
ASPX	Active Server Page Extended
ASF	Apache Software Foundation
API	Application Programming Interface
PHP	Hypertext Preprocessor
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
SMTP	Simple Mail Transfer Protocol
JMS	Java Message Service
XML	Extensible Markup Language
SQL	Structured Query Language
OPT	Optimização e Planeamento de Transportes
app	Application
Wi-Fi	Wireless Fidelity
ID	Identity
ASCII	American Standard Code for Information Interchange
GPS	Global Positioning System
GUI	Graphical User Interface
UML	Unified Modeling Language
IOS	iPhone Operating System

Capítulo 1

Introdução

Este projeto de dissertação ocorre no âmbito do Mestrado Integrado em Engenharia Eletrotécnica e Computadores (MIEEC), major de Telecomunicações, da Faculdade de Engenharia da Universidade do Porto. Vai ser realizado em ambiente empresarial na empresa EFACEC, supervisionado pela Engenheira Raquel Vieira e pela orientadora interna, Professora Maria Teresa de Andrade.

1.1 Apresentação do Problema

Nesta dissertação coloca-se o desafio de construção de uma aplicação móvel que tem como objetivo proporcionar aos utilizadores do Metro do Porto uma ajuda no usufruto deste serviço, criando nestes, fidelização e satisfação. Para tal, a aplicação deverá comportar, logo à partida, as seguintes funcionalidades: apresentar a informação de circulação, em tempo real, permitir ao utilizador o cálculo de distância entre o ponto em que se encontra e a estação de metro mais próxima, bem como, a possibilidade de o utilizador definir uma lista de estações favoritas para consulta futura. Esta aplicação a desenvolver, terá que dispor de uma ligação apenas de leitura, sobre uma base de dados fornecida pela EFACEC. Pretende-se, ainda, que tenha uma boa usabilidade, apresentando uma interface de fácil compreensão e controlo por parte do utilizador.

1.2 Motivação

Ao longo dos anos temos testemunhado grandes avanços tecnológicos, principalmente ao nível dos dispositivos móveis. Nos dias de hoje, existem aplicações diversas e com múltiplas funcionalidades destacando-se como exemplos aplicações para jogos ou aplicações para utilização das funcionalidades bancárias.

A base principal para o seu funcionamento é a programação, sendo esta uma área de grande interesse pessoal. Como utilizador frequente de transportes públicos, a oportunidade de estudar e

desenvolver um projeto que oferece um valor acrescentado para os utilizadores deste tipo de transportes, envolvendo também a oportunidade de, em paralelo, ser aplicada programação, estabeleceu em mim uma maior motivação no desenvolvimento do referido projeto.

Deparamo-nos várias vezes com situações de chegar a uma paragem de metro e este ter acabado de passar, originando uma espera pelo seguinte, por vezes longa.

Através do desenvolvimento de uma aplicação em que seja possível visualizar o tempo de chegada do metro à paragem em questão, podia-se previamente ajustar os tempos antes da deslocação para a paragem, ou na paragem, esse tempo utilizado de forma produtiva, criando assim no utilizador uma empatia com a empresa disponibilizadora do serviço.

1.3 Objetivos

O desenvolvimento da aplicação Android que estará na base do trabalho, pretende ajudar os utilizadores dos serviços de transporte, neste caso o Metro do Porto, facilitando a sua mobilidade e a gestão de tempo, fazendo com que o grau de satisfação dos utentes deste serviço, seja aumentado.

Como objetivos principais pretende-se que a *app* apresente, em cada paragem, quais os Metros que irão passar, os seus destinos e qual o tempo que falta para estes chegarem à paragem, bem como, dar ao utilizador informação da sua localização via GPS e, através desta, este calcular a distância a que se encontra da estação de metro mais próxima.

Um outro objetivo deste trabalho será a criação de uma lista de favoritos que apresente, consoante o interesse do utilizador, os horários já devidamente organizados da linha de Metro, que este pretende utilizar.

No desenvolvimento deste projeto foi sugerido pelos orientadores a inclusão de um objetivo secundário que é a possibilidade de ser criada uma rota, entre duas estações de metro.

Com isto pretendemos que o utilizador faça de forma autónoma a sua gestão do tempo de espera na paragem, podendo seguir assim, ao minuto o Metro que deseja, tendo acesso à informação de forma simples e rápida, num ambiente gráfico amigável.

1.4 Estrutura do Documento

Para além da introdução, esta dissertação contém mais 5 capítulos. No capítulo 2, é descrito o estado da arte. Aí, são explicados os Sistemas Operativos existentes e/ou mais utilizados, havendo uma análise breve sobre cada um. Também neste capítulo é realizado um estudo dedicado à arquitetura web que se mostrará necessária para o envio das informações, sendo analisados os vários tipos existentes. Essas informações são extraídas de uma Base de dados, havendo também a necessidade de um estudo sobre a mesma. Por fim a última parte é dedicada às aplicações atualmente existentes no mercado, e sobre elas é realizado uma estudo aferindo a sua usabilidade e informações disponibilizadas.

O capítulo 3 inicia-se com a especificação funcional, sendo dividido nas seguintes secções:

- Na secção 3.1 são descritos os modelos de caso de uso utilizados;

- Nas secções 3.2 são explicados cada caso de uso, através do evento de fluxos de casos de uso;
- Por fim na secção 3.3 é apresentado o *design* da aplicação e a forma como foi desenvolvida a estrutura para obtermos o produto final desejado.

No capítulo 4 é descrita a especificação técnica, sendo dividido nas seguintes secções:

- Na secção 4.1 é descrito o Sub-sistema Base de dados, apresentando todos os passos seguidos para o desenvolvimento do mesmo;
- Na secção 4.2 é descrito o Sub-sistema Android, apresentando um diagrama de classes, explicando cada classe e demonstrando a ligação existente entre os casos de uso e o diagrama de classes.

No capítulo 5 é apresentado o produto final, onde são demonstrados os resultados obtidos. Como se trata de um produto, é necessário fazer testes para provar que as funcionalidades se encontram conforme é desejado.

No último capítulo, o capítulo 6, são discutidos os resultados obtidos e dificuldades sentidas ao longo do desenvolvimento do projeto. Por fim é realizada uma reflexão sobre todo o trabalho desenvolvido, onde são elencadas as expectativas e pistas para o trabalho futuro.

1.5 Diagrama de organização do Documento

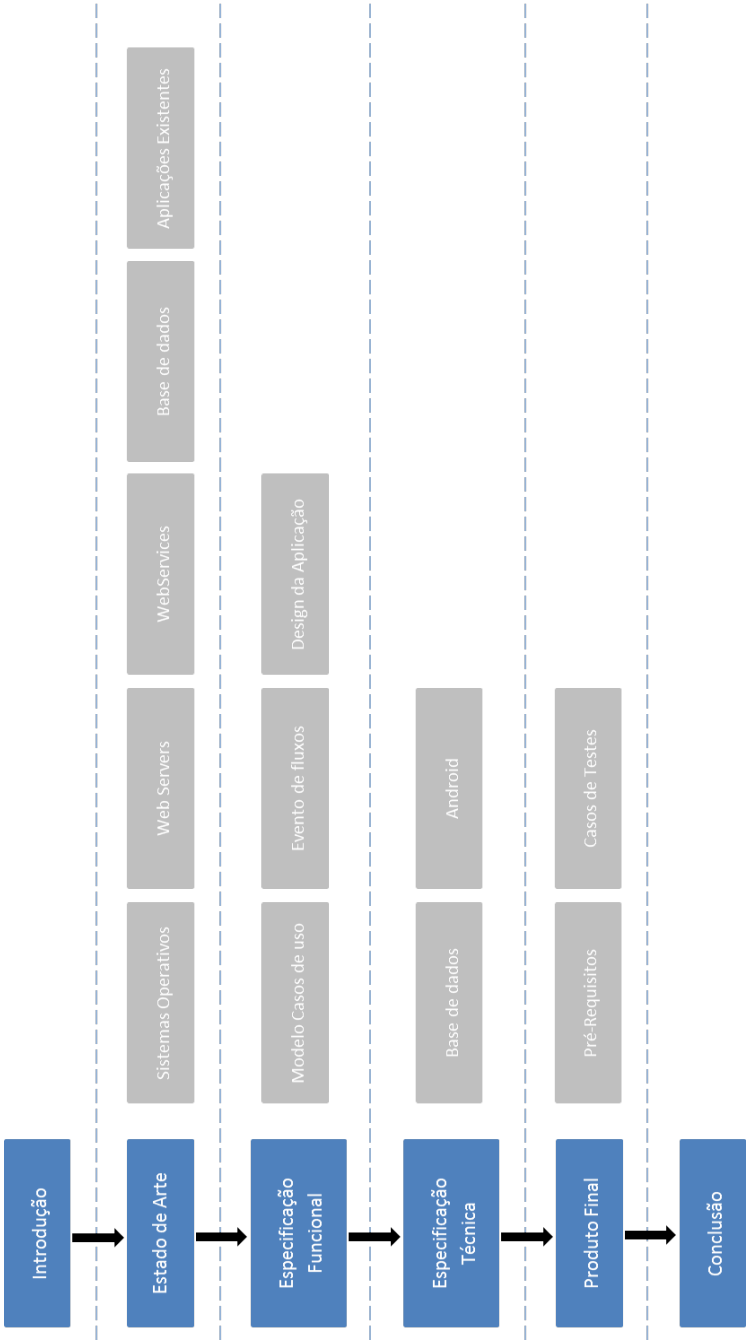


Figura 1.1: Diagrama do conteúdo da dissertação

No diagrama presente, Figura 1.1, podemos observar a constituição e organização desta dissertação.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é descrito o estado de arte, apresentando também trabalhos relacionados. São inscritas diversas referências bibliográficas.

Serão abordados os Sistemas operativos para dispositivos móveis disponíveis no mercado, assim como servidores web, webservices e Base de dados. Este estudo deve-se ao facto de cada sector ter influência no desenvolvimento do projeto.

2.1 Sistemas Operativos Móveis

2.1.1 Android

O sistema operativo Android é um sistema desenvolvido e suportado pela *Google Inc*, sendo a sua principal utilização em dispositivos móveis [9]. Este sistema operativo é baseado em Linux, sendo as licenças *open source*.

Este sistema operativo surgiu nos finais de 2008. Inicialmente não teve tanto sucesso em comparação com os sistemas operativos concorrentes tal como é possível observar na Figura 2.1. No entanto, a sua utilização ao longo dos anos foi aumentando exponencialmente, tendo ultrapassado mesmo a Apple em 2012. A Figura 2.2 demonstra esta alusão.

O Android baseia-se numa interface multi toque, esforçando-se por manter a interface, entre o dispositivo e o utilizador, o mais simples possível. Essa interação com o utilizador é feita, em parte, através de sensores integrados no dispositivo tais como acelerómetros e giroscópios, tornando a sua usabilidade mais atrativa. Isto é um facto bastante importante visto que um utilizador procura sempre algo que coloca à sua disposição o que pretende da forma mais simples e atrativa possível.

Uma das grandes vantagens deste sistema operativo é o seu desenvolvimento ser feito por qualquer utilizador, sendo possível a distribuição do produto final de forma gratuita, coisa que não acontece com outros sistemas operativos. O Software Development Kit que é recomendado, é o "Android Studio". Este Software é o único que de momento é disponibilizado pela Google com suporte em atualizações [10].

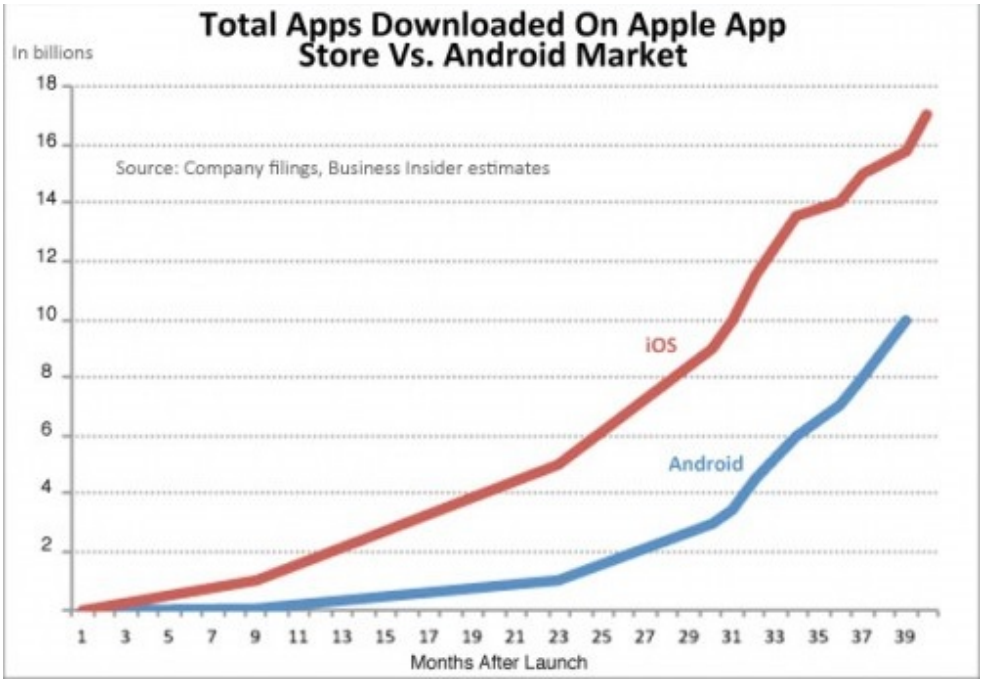


Figura 2.1: Gráfico do crescimento do número de downloads feitos quando Android e iOS foram lançados [1]

2.1.2 iOS

O sistema operativo iOS é um sistema criado e desenvolvido exclusivamente pela *Apple Inc*, sendo utilizado para dispositivos móveis [11]. Por ser um sistema operativo exclusivo, este tem a particularidade de apenas poder ser usado em dispositivos da Apple tal como iPods, iPads e iPhones. A sua utilização ou popularidade teve desde então um crescimento bastante acentuado.

Este sistema operativo surgiu no início de 2007. O seu crescimento desde então foi bastante acentuado. Tal é possível observar através da Figura 2.3, onde é visível o crescimento de novas aplicações para este sistema operativo, o que torna evidente o seu crescimento desde o seu surgimento.

À semelhança do Android, o iOS foca-se no desenvolvimento de dispositivos móveis tácteis, como tablets e *smartphones*. Os movimentos feitos pelo utilizador são baseados no manuseamento de objetos num ecrã táctil.

As aplicações que são desenvolvidas com o sistema operativo iOS como *main source*, são baseadas na linguagem C, sendo desenvolvidas especificamente em *Objective-C*.

A *Apple* fornece aos seus utilizadores a possibilidade de criar as suas próprias aplicações tal como o Android, usando um *Software Development Kit*, o *Apple Developer* [12]. Apesar de ser fornecido de forma gratuita, caso o programador pretenda disponibilizar as suas aplicações terá de pagar uma taxa "*Apple Developer Program fee*" para ser certificada pela *Apple Store*, sendo um dos maiores inconvenientes relativamente ao desenvolvimento de aplicações para iOS.

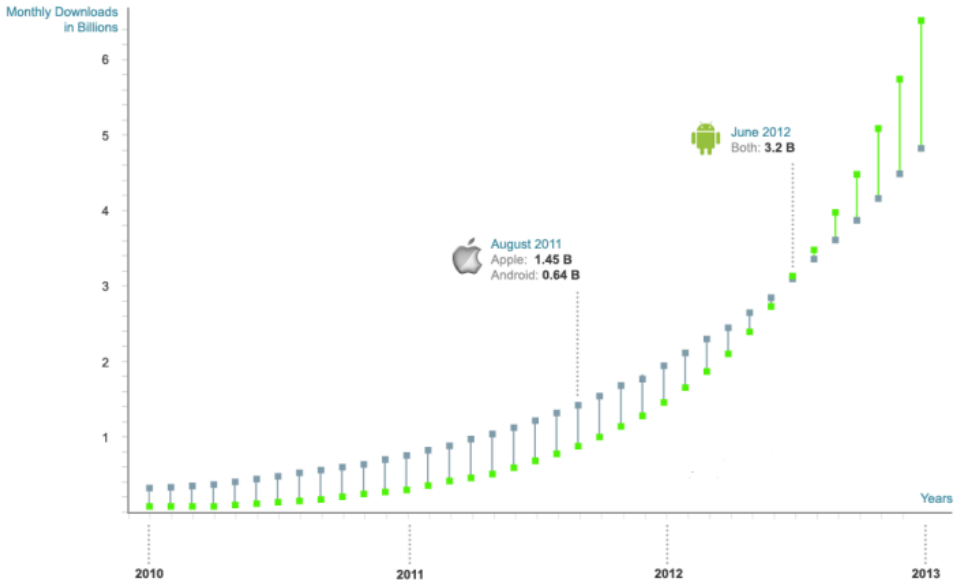


Figura 2.2: Gráfico do número de downloads entre 2010 e 2013 [2]

2.1.3 Windows Phone

O sistema operativo Windows Phone é o utilizado pela *Microsoft Inc* para os dispositivos móveis. Antes do aparecimento deste sistema, a Microsoft usava o *Windows Mobile* e o *Zune*. Estes dois foram os responsáveis pelo surgimento do Windows Phone, aliando-se as capacidades executivas do WM às capacidades multimédia do *Zune*, ou seja, usando as capacidades que cada um destes tinha de melhor para oferecer [13].

O surgimento deste sistema operativo foi nos finais de 2010, sendo nos dias de hoje um dos três sistemas operativos mais utilizados. Como o seu aparecimento foi dos últimos a surgir no mercado, este é o que tem a menor quota de implantação, em comparação com os seus principais concorrentes, Figura 2.4.

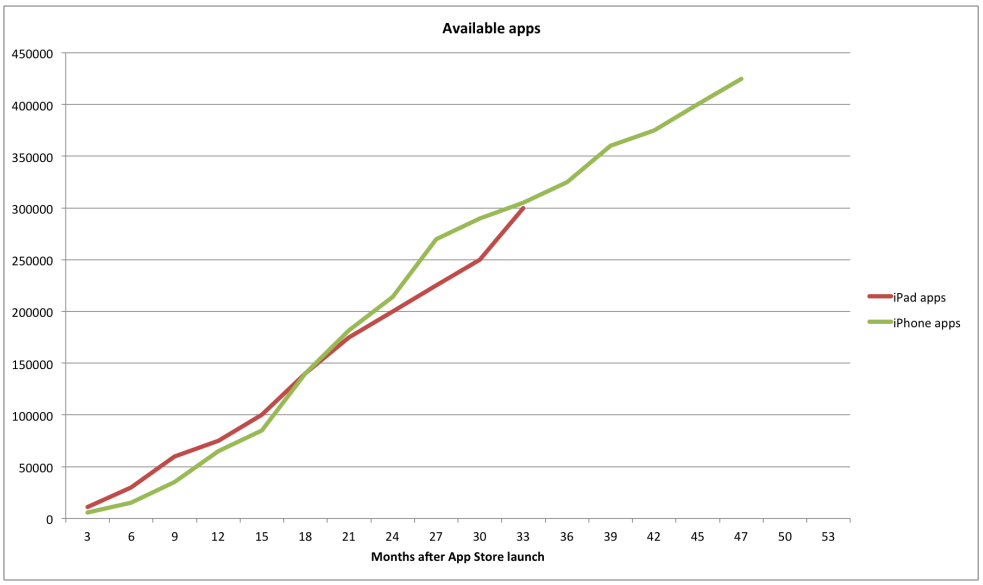
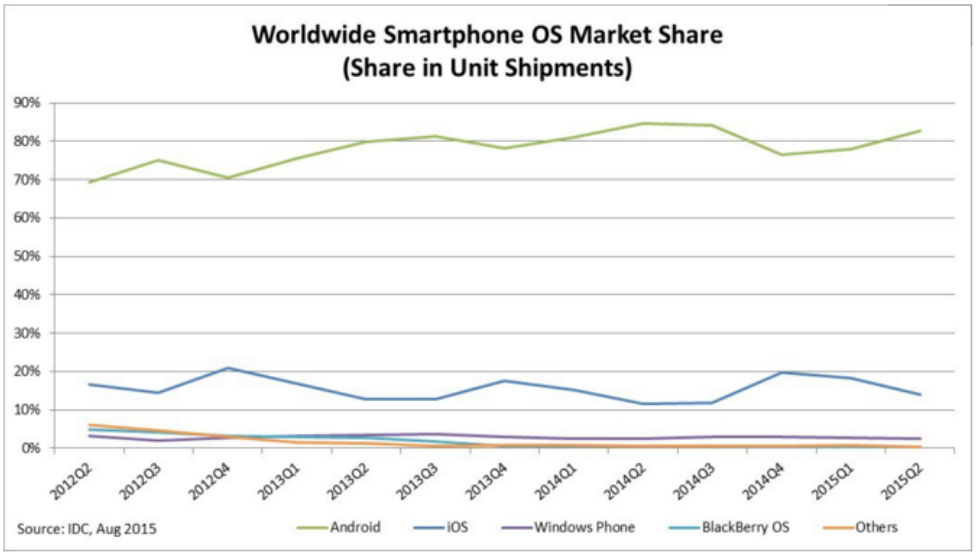


Figura 2.3: Gráfico das aplicações disponíveis quando iOS foi lançado [3]



Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	69.3%	16.6%	3.1%	4.9%	6.1%

Figura 2.4: Quotas de mercados dos Sistemas Operativos para dispositivos móveis [4]

2.2 Servidor Web (Web Server)

Um Servidor Web é um sistema que fornece conteúdo ou serviços para os utilizadores finais (clientes) através da Internet, respondendo a solicitações *Hypertext Transfer Protocol*. O Servidor Web é baseado num servidor físico, um Sistema Operativo (OS) de servidor e um software usado para facilitar a comunicação, como o HTTP [14].

Esta comunicação é possível porque cada dispositivo tem uma identificação única, chamado de Internet Protocol address, permitindo que os dispositivos comuniquem entre si através da rede.

Cada página web tem um endereço único, chamado de Uniform Resource Locator (URL). Quando o utilizador digita um URL num navegador web, envia uma solicitação para o Servidor Web e este responde enviando todo o conteúdo de volta para o endereço IP.

2.2.1 Internet Information Services

Internet Information Services é um *Servidor Web* criado pela Microsoft. Com o IIS, a Microsoft inclui um conjunto de programas para a construção e administração de sites *web*, um motor de busca, e suporte para aplicativos baseados em *web* que têm acessos a base de dados [15]. Este funciona apenas nos sistemas operativos Windows onde está incluído, sendo o seu uso gratuito. É um produto de *Software* fechado e suportado unicamente pela Microsoft.

A primeira versão este *Servidor Web* surgiu em 1996, o IIS 1.0. O IIS utiliza extensões de *web* externos para implementar algumas funcionalidades como por exemplo publicação *File Transfer Protocol*, roteamento de solicitação de aplicações, serviços de comunicação e regravação de URL que foram introduzidas com a versão IIS 7.5. Fornece também um grande apoio para o produto.NET (plataforma) e *Active Server Page Extended*. A versão lançada mais recentemente é a IIS 10 suportada no Windows 10, sendo a versão IIS 7 a mais utilizada nos dias de hoje tal como é possível observar na Figura 2.5.

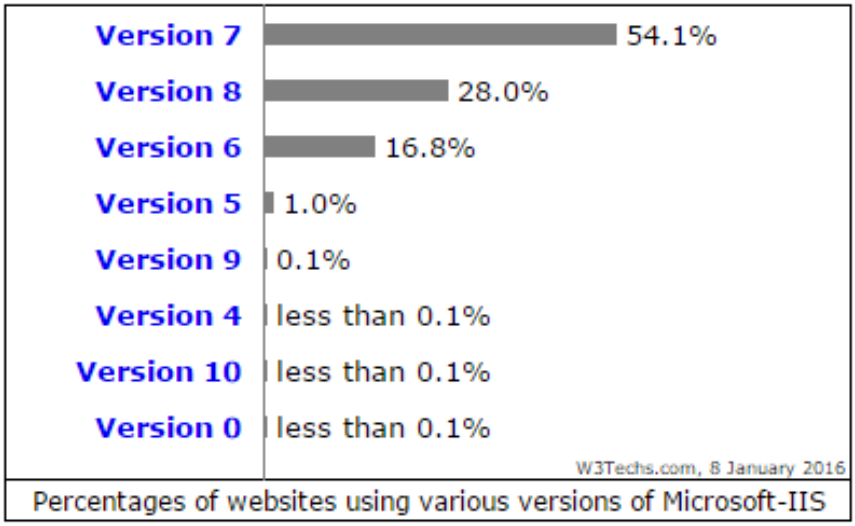
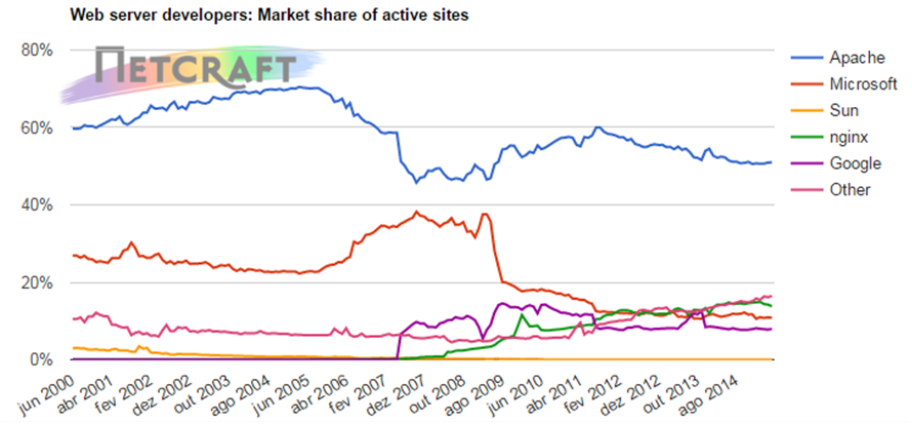


Figura 2.5: Percentagem das versões IIS usadas pelos websites [5]



Developer	April 2015	Percent	May 2015	Percent	Change
Apache	90,001,047	50.91%	89,387,114	50.98%	0.07
nginx	25,174,837	14.24%	24,204,572	13.81%	-0.43
Microsoft	19,353,327	10.95%	18,984,654	10.83%	-0.12
Google	13,712,694	7.76%	13,931,238	7.95%	0.19

Figura 2.6: Gráfico e tabelas dos websites ativos que utilizam os respetivos Servidor Webs [6]

2.2.2 Apache HTTP Server

Apache HTTP Server, mais conhecido por *Apache*, é um software open source Servidor Web de criação, implementação e gerenciamento. Foi desenvolvido inicialmente por um grupo de programadores mas é gerido atualmente pela empresa *Apache Software Foundation* [16]. Uma das características notáveis deste *Servidor Web* é ser capaz de suportar varias linguagens de programação, scripting do lado do servidor, mecanismo de autenticação e suporte de bases de dados.

Apache foi lançado para o mercado em 1995. Desde então, várias versões foram lançadas sendo a mais recente a versão 2.4.

Como o código fonte está disponível de forma gratuita, qualquer utilizador que pretenda utilizar pode adaptar o servidor para as suas necessidades específicas, existindo uma grande biblioteca de Apache add-ons. Pode ser utilizado em várias plataformas como Linux, Windows, Unix e Mac OS.

Atualmente é o *Servidor Web* mais utilizado neste mercado, tendo como maior concorrente o IIS referido anteriormente. Na Figura 2.6 podemos constatar esses mesmos dados.

2.2.2.1 Plataforma Slim

SLIM [17] é uma plataforma PHP utilizada para o desenvolvimento de aplicações web tal como descrito na figura 2.7. Utiliza API's RESTful na qual podemos construir aplicações web simples, seguras e eficazes. Tipicamente esta plataforma prevê o recebimento e encaminhamento de um pedido HTTP para um controlo especial, que retorna uma resposta HTTP. Esta plataforma

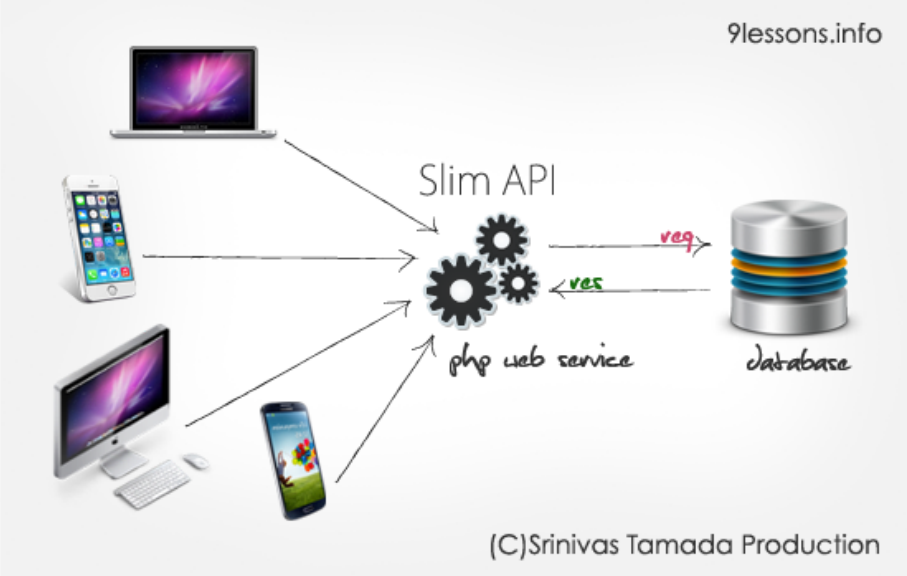


Figura 2.7: Descrição do processo fornecido pelo SLIM framework [7]

tem ferramentas adicionais tais como manipular a resposta HTTP como cache HTTP, redirecionar atualizações de status com implementação de *middleware*. No fundo, o SLIM lida principalmente com solicitações e respostas HTTP.

1. Arquitetura

    - Os dispositivos como os telemóveis, tablets ou computadores enviam um pedido HTTP para as API's RESTful. Neste caso, as API's RESTful fornecem uma interface entre a base de dados e os dispositivos;
    - As API's RESTful usam métodos da aplicação SLIM que serão usados para o processo da solicitação HTTP enviado pelos clientes;
    - AS API's RESTful recebem as solicitações HTTP a partir dos dispositivos e processam-nos na base de dados;
    - A base de dados recebe a indicação da informação solicitada e envia respostas de volta para os dispositivos através de API's RESTful como respostas HTTP.

2.3 WebServices

2.3.1 REST

A lógica por detrás do design desta arquitetura *web* pode ser descrita através de um estilo arquitetónico que consiste num conjunto de restrições aplicadas aos elementos dentro dessa arquitetura. Ao examinar o impacto de cada restrição podemos identificar as propriedades induzidas

pelas restrições *web*. Restrições adicionais podem então ser aplicadas para formar uma nova arquitetura que melhor reflita as propriedades desejadas, de uma arquitetura moderna *web*. Isto demonstra uma visão geral da definição de *REST* [18].

REST é constituída por três elementos: os componentes (proxy, gateway, entre outros), os conectores (cliente, servidor) e os dados (recursos, representação, entre outros).

Existem características específicas que compõem o estilo *REST*:

- **Cliente-Servidor** — A separação de interesses é o princípio por detrás das restrições de cliente-servidor. Ao separar as preocupações da interface do utilizador a partir das preocupações de armazenamento de dados, podemos melhorar a portabilidade da interface através de múltiplas plataformas e melhorar, assim, a escalabilidade, simplificando os componentes do servidor. Um dos aspetos importantes ao nível da *web* é o facto de os servidores e clientes poderem ser desenvolvidos e substituídos de forma independente, não alterando a interface entre eles;

- **Stateless** — Cada solicitação do cliente para o servidor deve conter todas as informações necessárias para este compreender o pedido, sem tirar vantagem de qualquer contexto armazenado no servidor. Ou seja, o estado da sessão é mantido inteiramente no cliente. Esta restrição induz a três propriedades. A visibilidade é melhorada porque um sistema de monitorização não tem de observar para além de um único pedido, de forma a determinar a natureza completa do pedido. A fiabilidade é também melhorada visto que facilita a tarefa de falhas parciais. A última propriedade é a escalabilidade que é também melhorada por não ter de armazenar o estado entre as solicitações, permitindo o acesso a recursos livres à componente servidor. A implementação desta é também simplificada porque não tem de gerir o uso de recursos entre solicitações.

  A desvantagem do *Stateless* é o facto de poder diminuir o desempenho da rede, aumentando os dados repetitivos (sobrecarga por interação) enviando uma série de pedidos, uma vez que os dados não podem ser deixados no servidor num contexto partilhado;

- **Cache** — As restrições da *Cache* exigem que os dados dentro de uma resposta a um pedido sejam rotulados como *cacheable* ou *non-cacheable*. Se a resposta é cacheable, então a uma *cache* do cliente é dado o direito de reutilizar os dados de respostas para mais tarde, a partir de pedidos equivalentes.

  A vantagem de utilizar restrições do tipo *cache* é que têm potencial para eliminar parcialmente ou completamente algumas interações, melhorar a eficiência e escalabilidade através da redução da latência média de uma série de interações;

- **Interface Uniforme** — Todos os recursos são acedidos através de uma interface genérica como por exemplo *HTTP GET*, *POST*, *PUT* e *DELETE*;

- **Sistema de camadas** — O estilo de sistema em camadas permite uma arquitetura composta por camadas hierárquicas ao restringir o comportamento dos componentes de tal forma que,

cada componente não possa "ver" além da camada imediatamente a seguir, com as quais estão a interagir. Ao restringir o conhecimento do sistema para uma única camada, coloca um limite de complexidade geral do sistema, promovendo a independência do substrato. As camadas podem ser usadas para encapsular os legados dos serviços e para proteger os novos serviços dos legados dos clientes, simplificando os componentes ao mover funcionalidades raramente usadas por um intermediário partilhado.

A principal desvantagem deste sistema é o facto de adicionar sobrecarga e latência para o tratamento de dados, reduzindo o desempenho. Este problema poderá ser resolvido através das restrições de *cache*.

2.3.2 SOAP

SOAP é um protocolo de mensagens que permite que programas sejam executados em diferentes Sistemas Operativos como Windows ou Linux, e comuniquem usando *HTTP* e *XML*.

Desde que os protocolos Web foram instalados e estão disponíveis para uso em todos os sistemas operativos, o *HTTP* e *XML* fornecem uma solução que permite aos programas correrem em diferentes sistemas operativos na rede de forma a comunicarem entre si. SOAP especifica exatamente como codificar o *HTTP header* e o ficheiro XML de forma a que um programa num computador possa chamar outro programa de um outro computador e transmita essa informação. Também especifica como um programa é chamado, podendo retornar a resposta. Apesar da sua união frequente ao *HTTP*, *SOAP* suporta também outros protocolos de transporte.

SOAP define o formato da mensagem baseado em *XML* que são usados para comunicar e interoperar pelos serviços web uns com os outros. O ambiente heterogéneo da Web exige que as aplicações suportem um protocolo de codificação de dados comum e formato de mensagem. Resumidamente, o *SOAP* é um padrão para a codificação de mensagens em *XML* que invocam funções noutros aplicativos.

Uma mensagem *SOAP* é um documento *XML* comum, contendo os seguintes elementos: um envelope que identifica o documento *XML* como uma mensagem *SOAP*; um *header* (cabeçalho) que contém informações de cabeçalho; um corpo que contém a chamada e resposta de informações e um elemento chamado *fault* (falha) que contém erros e informações do estado.

As principais características da *SOAP* são: extensibilidade, sendo que a segurança e o roteamento de webservices estão entre as extensões em desenvolvimento; neutralidade, podendo operar em qualquer protocolo de transporte tal como http, o *Simple Mail Transfer Protocol*, *Java Message Service*, bem como outros; e independência, permitindo qualquer modelo de programação [19].

Algumas das vantagens fornecidas pela *SOAP* são:

- É uma plataforma e uma linguagem independente;
- Fornece comunicações simplificadas através de proxies e firewalls;
- Tem capacidade de influenciar diferentes protocolos de transportes, referidos anteriormente.

Algumas das desvantagens fornecidas pela *SOAP* são:

- É tipicamente mais lenta do que outros tipos de arquitetura. Isto acontece devido ao facto de usar um formato *XML* detalhado, implicando que seja necessário entender bem as limitações de desempenho, antes de construir aplicações em torno da SOAP;
- Tem diferentes níveis de suporte, dependendo da linguagem de programação utilizada. Por exemplo, o suporte da *SOAP* com Python e *PHP* é bastante inferior ao suporte com Java e .NET.

2.4 Base de Dados

Uma base de dados é uma coleção de informações, sendo organizada de forma a que seja facilmente acedida, gerenciada e atualizada. Em computação, as bases de dados são classificadas de acordo com a sua abordagem organizacional. Existem dois tipos de abordagem mais comuns: *XML* e relacional.

2.4.1 XML

Extensible Markup Language é usado para descrever dados. O padrão *XML* é uma maneira flexível para criar formatos de informação e eletronicamente partilhar dados estruturados via Internet. É possível afirmar-se que um documento *XML* é uma base de dados no seu sentido mais estrito, ou seja, como uma coleção de dados. Quando este é encarado como um formato de base de dados, apresenta algumas vantagens como a portabilidade (um sistema *Unicode* de representação), ser auto-descrito (os nomes dados às marcações descrevem a estrutura) e a sua capacidade de organizar os dados numa estrutura em árvore. Apesar das suas vantagens, apresenta também algumas desvantagens como o acesso aos dados ser lento e a verbosidade[1]. Mesmo sendo possível usar um documento *XML* como uma base de dados em ambientes com pequenas quantidades de dados, verificou-se que esse suporte falha na maioria dos ambientes de produção como o armazenamento eficiente, índices, transações, triggers, segurança, interrogações, etc. Como tal, foram criados dois tipos de mecanismos de tratamento de documentos *XML*, designados por base de dados *XML* enabled e *XML* nativas.

2.4.1.1 XML enbaled

XML enabled não é nada mais do que uma extensão fornecida para a conversão de documentos *XML*. Estas bases de dados não trabalham diretamente com a representação de informação em *XML*. Apesar disso, conseguem armazenar documentos *XML* e extrair informação deles. A informação é armazenada em tabelas que consistem em linhas e colunas contendo um conjunto de registos, os quais por sua vez consistem em campos. Existem três grandes problemas destas bases de dados, sendo uma delas a escalabilidade, manifestando-se na degradação do desempenho

[1] Uso excessivo de palavras.

com o aumento do documento *XML* ou quando é necessário lidar com vários documentos em simultâneo. Um outro problema é a falta de *queries* estruturadas, sendo que os motores de pesquisa não reconhecem a estrutura da sintaxe *XML*, não conseguindo distinguir elementos de atributos e respetivos conteúdos. Por último temos a falta de possibilidade de tirar partido dos mecanismos das bases de dados como a segurança, concorrência, entre outras. A solução encontrada para estes problemas foi o uso de bases de dados *XML nativas*.

2.4.1.2 XML nativas

Estes tipos de bases de dados baseiam-se no recipiente, em vez de formato de tabela. Conseguem armazenar vários documentos *XML* e dados.

As características que se destacam relativamente às *XML enabled* são:

- Existe separação de informação no carregamento;
- Após inserção de um documento *XML*, é efetuado de imediato a separação e classificação dos dados, sendo separados os elementos, atributos e respetivos valores. A indexação é também efetuada no momento do carregamento;
- É feita uma interrogação direta com tecnologias associadas à XML. As interrogações são feitas através da consulta de índices. As linguagens de interrogações podem variar, sendo que a *Xquery* tem sido a mais implementada ao nível de soluções comerciais e open-source. A tecnologia XPath assume um papel importante visto que permite identificar grupos de elementos num documento;
- Tem a capacidade de manipulação de grandes documentos;
- As bases de dados *XML* nativas não implicam utilização de esquemas fixos, conseguindo manipular os mesmos, tendo uma maior flexibilidade em acompanhar a evolução do esquema [20][21][22][23].

2.4.2 Relacionais

Uma base de dados relacional é um conjunto coletivo de vários conjuntos de dados organizados por tabelas, registos e colunas. Estas tabelas comunicam e compartilham informações, o que facilita a procura de dados, organização e elaboração de relatórios. Estas bases de dados utilizam uma estrutura chamada de *Structured Query Language*, sendo esta, um aplicativo do utilizador-padrão que fornece uma interface de programação fácil para a interação com a base de dados.

A organização dos dados é feita de várias formas. Cada tabela é conhecida como uma relação, contendo um ou mais dados na categoria colunas. Cada registo da tabela (ou linha) contém uma instância de dados únicos definidos para uma categoria da coluna correspondente. Um ou mais dados dizem respeito a um ou mais registos, formando dependências funcionais. Estes podem ser classificados da seguinte forma:

- **Um para um (1:1)** — Um registo de uma tabela refere-se a outro registo de uma outra tabela;
- **Um para muitos (1:N)** — Um registo de uma tabela refere-se a mais do que um registo de uma outra tabela;
- **Muitos para um (N:1)** — Mais do que um registo de uma tabela referem-se a um registo de uma outra tabela;
- **Muitos para muitos (N:N)** — Mais do que um registo de uma tabela referem-se a mais do que um registo de uma outra tabela.

Uma base de dados Relacional executa três tipos de operações: Selecionar, Projeto e Juntar. Selecionar é usado para recuperar dados, Projeto é usado para identificar atributos de dados e Juntar é usado para relações combinadas entre atributos de diferentes tabelas.

As grandes vantagens deste tipo de base de dados são:

- Extensibilidade fácil, podendo um novo dado ser adicionado sem modificar os registos já existentes, também conhecido por escalabilidade;
- Maior potência e flexibilidade com múltiplas capacidades de requisitos de dados;
- Segurança de dados, o que é crítico quando a partilha de informação é baseada em privacidade. Um exemplo simples de demonstrar isto é quando a administração de uma empresa pretende fornecer certos privilégios de dados e de acesso e, ao mesmo tempo, bloquear os funcionários de outros dados, como informação privilegiado ou salários confidenciais [24][25][26].

2.5 Aplicações Existentes

Existem várias aplicações utilizadas em redes de metropolitano. Neste estudo, serão analisadas quatro aplicações de diferentes cidades.

2.5.1 Tune Map London Underground

Após abrir a aplicação, o layout apresentado inicialmente é o representado na Figura 2.8 do lado esquerdo. Como é possível observar, encontram-se representadas todas as linhas do metro e as respectivas paragens.

Clicando no botão Menu do dispositivo móvel, serão apresentadas várias funcionalidades existentes na aplicação, tal como retrata a Figura 2.8 do centro. Se pretender ir de uma estação para outra, indicando qual a inicial e a final, clica no botão "*Route*" e ele indicará quais as linhas em que deverá viajar e o tempo estimado de viagem. O "Locate me" abrirá o *GoogleMaps* indicando a sua posição atual, o "*Clear Map*" apagará qualquer alteração feita no mapa das linhas de metro. Os outros botões são apenas informativos.

No canto superior da aplicação, existe um botão com um sinal de informação que disponibiliza informações em tempo real (Figura 2.8 do lado direito). Na primeira opção poder-se-á ver o tempo de espera até à partida do próximo metro de cada linha, indicando o seu destino final. No resto das opções poder-se-á ver as linhas que se encontram fechadas de momento, o primeiro e ultimo metro de cada linha e a meteorologia dos próximos três dias.

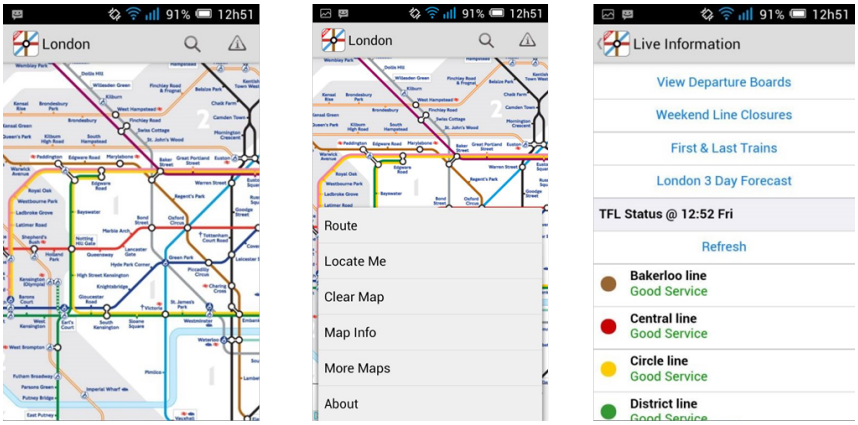


Figura 2.8: Apresentação da aplicação Tube Map London Underground

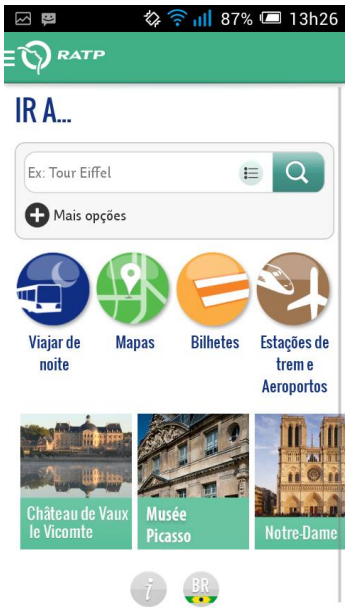


Figura 2.9: Apresentação inicial da aplicação RATP

2.5.2 RATP - Visit Paris by Metro

Esta aplicação está disponível em várias línguas como inglês, francês, espanhol, brasileiro, entre outras. É uma aplicação que funciona em modo *Offline*.

Tem várias funcionalidades como é possível observar na Figura 2.9, como o botão "Viajar de Noite" que fornece os horários de funcionamento, mapas, preços e como aceder às estações de comboio e aeroportos. No botão "Bilhetes" é possível fazer uma simulação para ver qual o bilhete indicado para o utilizador comprar. No botão de pesquisa é possível colocar o local para onde se pretende deslocar e, através do *GPS* do dispositivo móvel, ele indica a viagem que terá de ser percorrida.

2.5.3 Metro de Madrid

Esta aplicação está disponível em várias línguas como inglês, francês, espanhol, entre outras. É uma aplicação que funciona em modo *Offline*.

A aplicação tem quatro funcionalidades como é possível observar na Figura 2.10. A primeira funcionalidade "Lines and Stations" indica todas as linhas existentes do metro e quais as estações que fazem parte dessas linhas. "*Routes*" indica ao utilizador o caminho a percorrer entre dois pontos, previamente assinalados pelo mesmo. No "*Maps*" existem dois tipos de mapas disponíveis, o "*Metro Map*" que fornece o mapa completo de todas as linhas existentes, enquanto que o "*Tourist Map*" fornece também pontos turísticos, maioritariamente históricos. Por último temos o "*Information*" que indica ao utilizador informação útil para a utilização da aplicação, que tipo de bilhetes comprar e o horário de funcionamento.

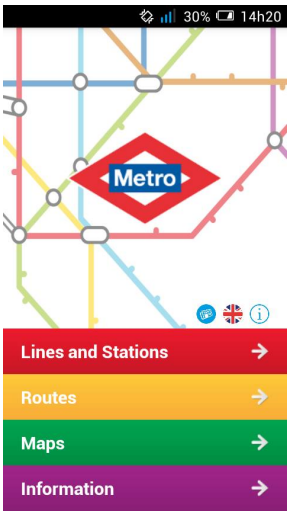


Figura 2.10: Apresentação inicial da aplicação Metro de Madrid

2.5.4 MOVE-ME

É uma aplicação Portuguesa desenvolvida pela empresa OPT (Otimização e Planeamento de Transportes) em parceria com operadoras como STCP, Metro do Porto, entre outros. É uma aplicação que funciona em modo Online para os sistemas operativos Android e iOS.

Tal como é possível constatar na Figura 2.11 do lado esquerdo, após escolher a operador "Metro do Porto", o utilizador terá de escolher a linha e de seguida, em que sentido pretende deslocar-se. Por fim, escolhe a paragem em que se encontra. Na figura 2.11 do lado direito podemos analisar os resultados obtidos. Estes reportam todos os metros que por ali irão passar e o tempo de espera até à chegada do mesmo dentro de um período de sessenta minutos.

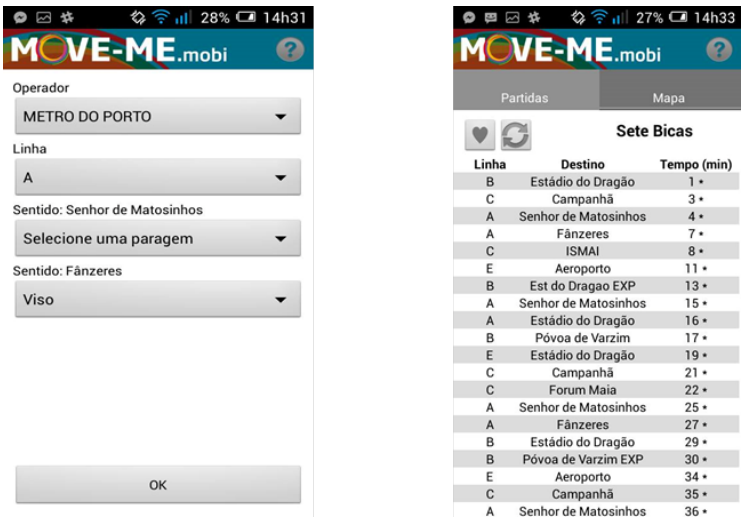


Figura 2.11: Apresentação do funcionamento do Metro do Porto na aplicação MOVE-ME

2.5.5 Moovit

Esta aplicação foi desenvolvida para os transportes públicos, tanto metro como comboios e autocarros. Tem vários idiomas disponíveis, sendo que está a ser utilizado em vários países, num total de 65 países, 850 cidades.

É uma aplicação bastante completa e com várias funcionalidades tais como as que surgem na figura 2.12. Poderá visualizar o mapa e determinar um percurso que pretende percorrer, indicando o caminho mais apropriado. Se estiver dentro de um transporte público, poderá seguir a viagem em direto, sabendo por onde vai passar e qual o tempo previsto. Tem Favoritos, sendo possível escolher linhas ou estações para guardar. Para além disso, pode observar o Mapa das linhas, um melhor caminho a percorrer na viagem do dia em questão. Tem definições para personalizar a app de forma a estar ao gosto do utilizador e um suporte a explicar a sua utilização.

Esta aplicação foi testada durante quase um ano numa versão Beta, tendo sido lançada oficialmente em Portugal a 31 de Março de 2016.

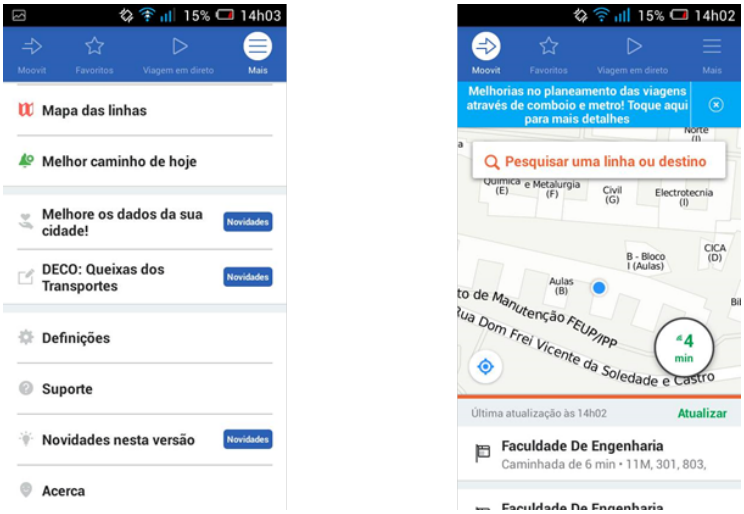


Figura 2.12: Apresentação do funcionamento da aplicação Moovit

2.6 Sumário

Através deste estudo aprofundado em cada sector, deparou-se com a necessidade de determinar qual ou quais os percursos possíveis a percorrer para o desenvolvimento deste trabalho.

No caso dos Sistemas Operativos verificou-se que a possibilidade de desenvolver para ambos os sistemas (Android e IOS) não era de facto possível, uma vez que a utilização do mapa em sistemas multi-plataforma comportaria custos. Assim, houve necessidade de se escolher ou sistema Android ou IOS, tendo-se optado pelo Android uma vez que este é o mais abrangente no marcado.

A utilização de webservices implicava uma complexidade bastante acentuada no desenvolvimento de programação Android para a ligação entre ambos, e como o tempo é limitado, optou-se pela utilização de uma plataforma chamada SLIM que utiliza como servidor web, o Apache.

A Base de dados fornecida pela empresa é do tipo relacional, daí a importância do estudo prévio realizado para a sua utilização.

O estudo realizado sobre as aplicações existentes permitiu entender que tipo de funcionalidades seriam importantes implementar para tornar a aplicação o mais completa e funcional possível, do ponto de vista do utilizador. Com isso apercebeu-se que criar uma rota desde uma estação Origem até uma estação Destino seria bastante útil, assim como a criação de Favoritos.

Capítulo 3

Especificação Funcional

Este capítulo fornece a definição e descrição funcional da arquitetura da aplicação desenvolvida. Descreve o projeto inicial de cada componente do sistema e a funcionalidade que cada um oferece. Demonstra ainda a contribuição de cada um e a forma como esses componentes cooperam entre si para fornecer a funcionalidade global do sistema. Isto é feito recorrendo à notação UML (Unified Modelling Language). UML é uma abordagem ou linguagem que permite "especificar, visualizar e estabelecer dependências entre os artefactos de sistemas de software".

Neste é incluído os casos de uso em diagramas UML, ilustrando os casos de usos considerados, do ponto de vista conceptual, mais relevantes para a definição da arquitetura desenvolvida, que contemplam um maior número de requisitos e que por isso são mais complexos, envolvendo um maior número de funcionalidades do sistema. A figura 3.1 ilustra o Sistema, indicando todos os seus componentes. Como é possível observar, nas instalações da empresa EFACEC está instalado o servidor Web numa máquina disponibilizada, e a Base de dados Oracle que foi devidamente alterada para o desenvolvimento deste projeto. O dispositivo móvel será o objeto utilizado para testar a aplicação, sendo utilizado o sistema operativo Android. Existem também API's externas como Google API e Android API's para aceder aos sensores do dispositivo móvel.

A figura 3.2 ilustra como o sistema é utilizado, quais os atores envolventes e os sistemas externos utilizados. Existem um total de quatro atores: admin, utilizador, interface gráfica (GUI) e Web Server, que fará de ator e será também utilizado como um caso de uso.

O ator "GUI" representa a interface gráfica do sistema e serve de intermediário entre o utilizador (humano) e o resto do sistema. Desta forma o ator "Utilizador" representa o utilizador humano (utilizador fina do sistema) e interage com o sistema através da GUI, executando tarefas tais como premir botões e navegar nas páginas oferecidas pela GUI. O Web Server é responsável pela ligação à Base de dados, permitindo que seja obtida a informação necessária. Por fim, o admin será o principal responsável pela manutenção de todo o sistema.

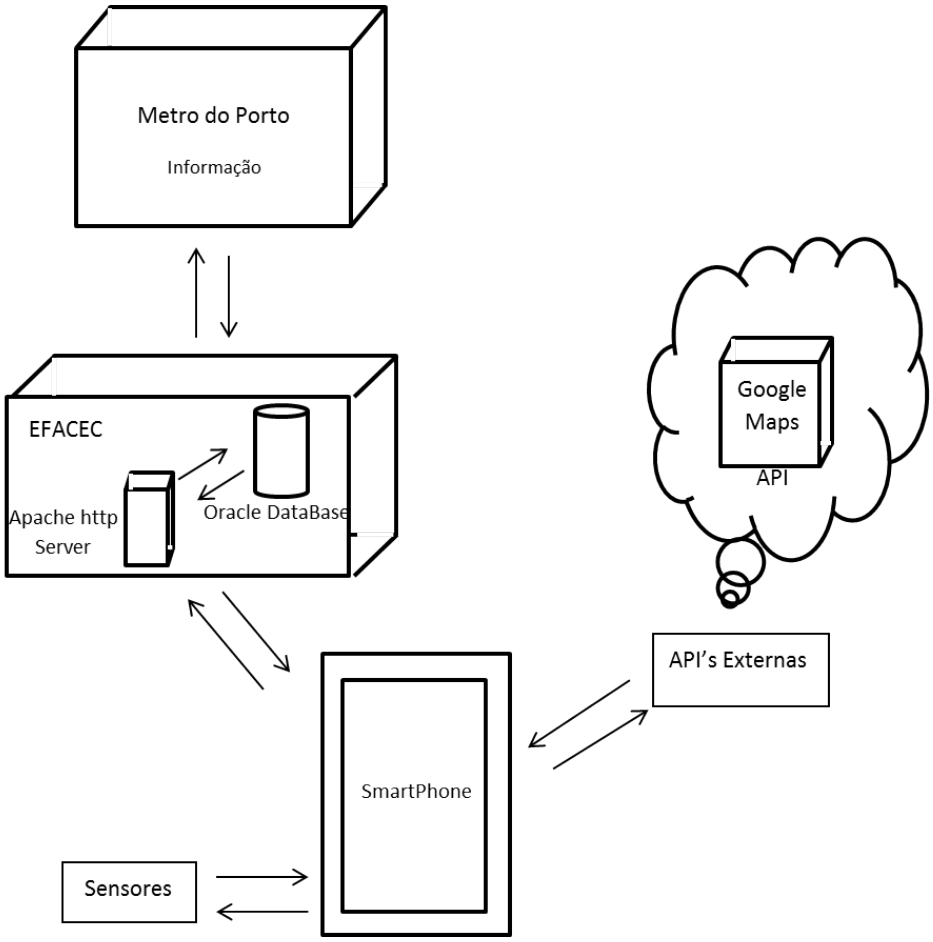


Figura 3.1: Sistema implementado com os seus componentes

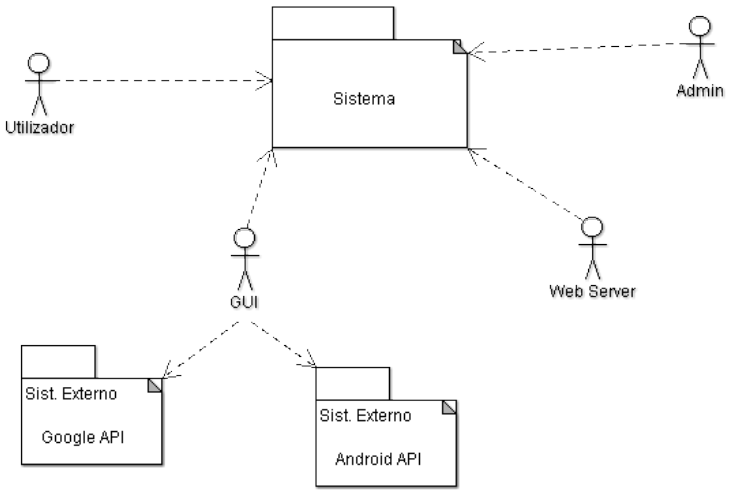


Figura 3.2: UML - O Sistema e os atores envolventes

3.1 Modelo Casos de uso

A figura 3.3 demonstra uma visão geral do modelo de casos de uso. Contém apenas os casos de uso considerados mais relevantes no ponto de vista da arquitetura. Os atores identificados na figura 3.2 são os considerados iniciadores, não implicando que sejam os únicos. Um ator é uma entidade externa para o Sistema que tipicamente estimula o Sistema com entrada de eventos, ou que recebe algum tipo de informação deste. Para um determinado caso de uso existe um ator iniciador que gera um estímulo inicial, podendo existir vários outros atores participantes. Na figura 3.3 são apresentados apenas os atores iniciadores. Este tipo de diagrama fornece uma visão de baixo nível já que, para cada ator, apresenta o sistema de uma forma parcial, evidenciando apenas as partes do sistema que são relevantes (ou que contribuem) para a realização de cada caso de uso e assim pela implementação da funcionalidade desejada, tal como é possível observar na Tabela 3.1.

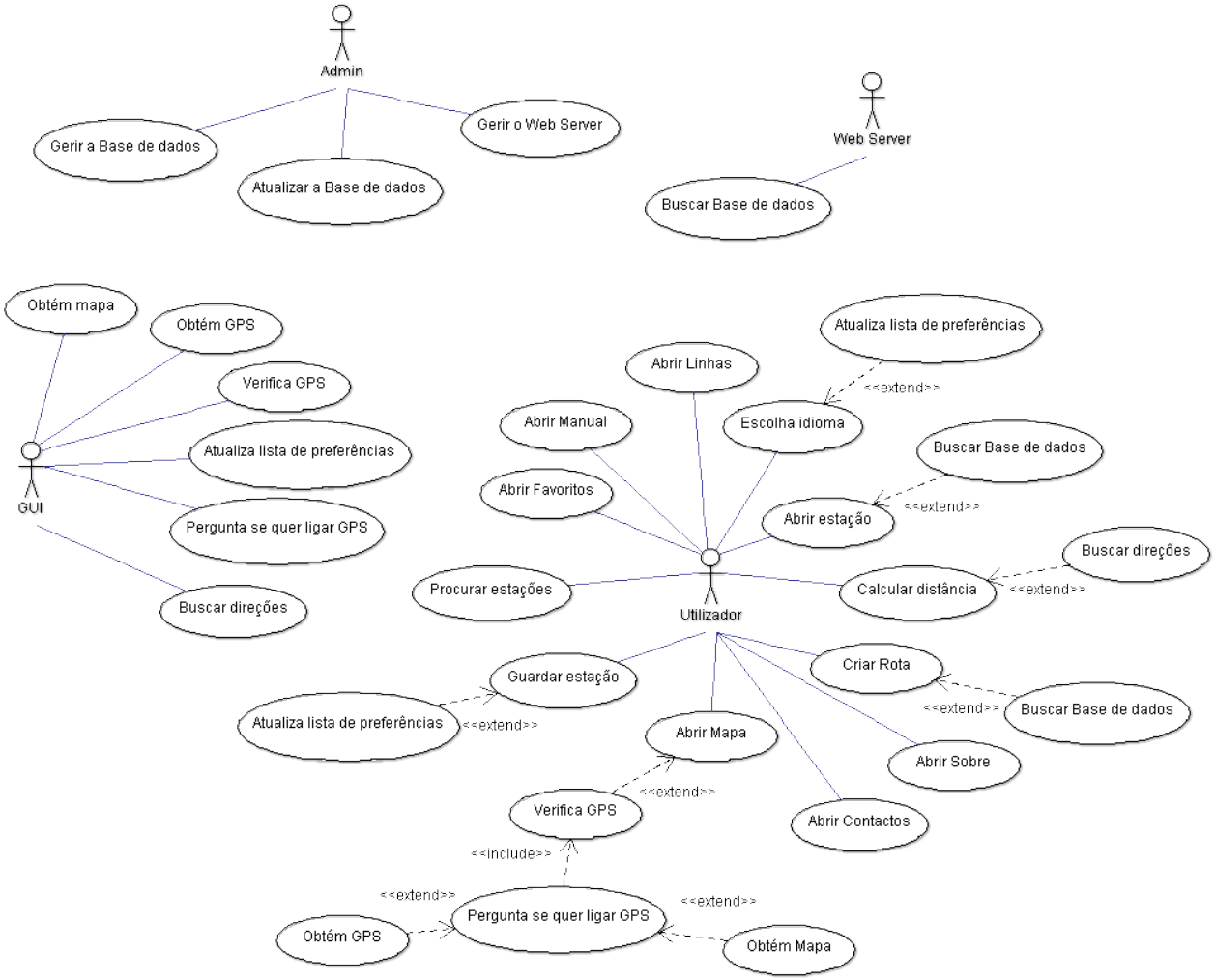


Figura 3.3: Atores e os seus Casos de uso

Nome Caso de Uso	Valor devolvido ao Sistema	Ator
Gerir a Base de dados	Criar, editar e eliminar tabelas da BD.	Admin
Atualizar a Base de dados	Atualizar a informação contida na BD.	Admin
Gerir o Web Server	Adicionar, editar ficheiros contidos no Web Server.	Admin
Buscar Base de dados	200OK no caso de a ligação ser bem-sucedida.	Web Server
Obtém Mapa	Fornecer o mapa da Google ao Sistema.	GUI
Obtém GPS	Fornecer valores de coordenadas GPS.	GUI
Verifica GPS	Indicar se está ou não ligado o sensor GPS.	GUI
Atualiza lista de preferências	Criar, editar, eliminar valores num ficheiro de texto.	GUI
Pergunta se quer ligar GPS	Aparecer uma mensagem a indicar se pretende ligar.	GUI
Buscar direções	Fornece informação entre dois pontos, como distância e tempo que demora a percorrer	GUI
Abrir Linhas	Fornecer informação das linhas de Metro.	Utilizador
Abrir Favoritos	Fornecer todas as estações guardadas.	Utilizador
Guardar estação	Guardar estação para visualizar nos Favoritos.	Utilizador
Abrir estação	Aparecer informação de circulação da estação, em tempo real.	Utilizador
Criar Rota	Escolher Origem e Destino e criar uma rota.	Utilizador
Escolha idioma	Escolher um dos idiomas disponíveis.	Utilizador
Abrir Mapa	Aparecer o mapa no Sistema.	Utilizador
Abrir Manual	Aparecer informação sobre as funcionalidades.	Utilizador
Abrir Sobre	Aparecer informação sobre a aplicação.	Utilizador
Abrir Contactos	Aparecer informação sobre os contactos disponíveis.	Utilizador
Procurar estações	Localização GPS da estação procurada.	Utilizador
Calcular distância	Caminho possível a percorrer entre dois pontos.	Utilizador

Tabela 3.1: Tabela com os Casos de Uso

3.2 Evento de fluxos dos Casos de uso

3.2.1 Gerir a Base de dados

Este caso de uso descreve a ligação entre o admin e a Base de dados, como é feito a gestão do mesmo.

Caso de Uso:	**Gerir a Base de dados**
Ator:	Administrador
Objetivo:	Criação, edição ou eliminação de tabelas existentes na Base de dados.
Visão Geral:	O administrador é o único capaz de alterar a estrutura da Base de dados, podendo criar novas tabelas, editar ou até eliminar tabelas existentes sempre que necessário.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o admin faz a autenticação na conta da Base de dados.	**2**	Uma interface é apresentada com todas as tabelas disponíveis.
3	O admin escolha qual das operações pretende executar, como por exemplo criar nova tabela ou até mesmo eliminar uma existente.	**4**	A operação escolhida pelo admin é executada, guardando as alterações realizadas.
5	Quando terminado, o admin fará log out.		

Tabela 3.2: Caso de uso: Gerir a Base de dados

3.2.2 Atualizar a Base de dados

Este caso de uso descreve a ligação entre o admin e a Base de dados, como é atualizado e de que forma.

É de notar que, apesar do ator ser o admin, a Base de dados é também atualizada automaticamente relativamente à informação de circulação, em tempo real. Esta é feita através de balizas ao longo do percurso de Metro e, sempre que um metro por lá passar, essa informação é recebida e guardada na Base de Dados.

Caso de Uso:	**Atualizar a Base de dados**
Ator:	Administrador
Objetivo:	Atualização da informação contida nas tabelas da Base de dados.
Visão Geral:	O administrador é o único capaz de alterar a informação contida nas tabelas existentes na Base de dados. Por vezes a informação fica desatualizada e é necessário o administrador alterar essa mesma informação de forma a atualizá-la.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o admin faz a autenticação na conta da Base de dados.	**2**	Uma interface é apresentada com todas as tabelas disponíveis.
3	O admin escolhe a tabela que pretende alterar.	**4**	A tabela é aberta, apresentando toda a informação nela contida.
5	O admin poderá eliminar alguma linha desejada, adicionar uma nova ou até mesmo alterá-la.	**6**	A operação escolhida pelo admin é executada, guardando as alterações feitas.
7	Quando terminado, o admin fará log out.		

Tabela 3.3: Caso de uso: Atualizar a Base de dados

3.2.3 Gerir o Web Server

Este caso de uso descreve a ligação entre o admin e o Web Server, explicando como este é gerido.

Apesar de o Web Server ser um ator, neste caso funciona como Caso de uso visto que é necessário ser gerido por um ator externo, de forma a estar funcional.

Caso de Uso:	**Gerir o Web Server**
Ator:	Administrador
Objetivo:	Adicionar, editar ou eliminar ficheiros contidos no Web Server.
Visão Geral:	O administrador é o único capaz de gerir a informação/ficheiros contidos no Web Server. Para isso tem de aceder à máquina onde se encontra o Web Server, tendo de se autenticar. Após isso pode alterar os ficheiros, podendo adicionar, editar ou até mesmo eliminar.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o admin faz a autenticação na máquina que tem instalado o Web Server.	**2**	Uma interface é apresentada com a pasta correspondente ao Web Server.
3	O admin poderá configurar o Web Server, acedendo ao ficheiro php.ini.	**4**	A informação alterada é guardada.
5	O admin poderá inserir páginas php.	**6**	O Web Server disponibiliza essas páginas, fornecendo um URL para que seja possível aceder a essas mesmas páginas.
7	Quando terminado, o admin fará log out.		

Tabela 3.4: Caso de uso: Gerir o Web Server

3.2.4 Buscar Base de dados

Este caso descreve a ligação entre o Web Server e a Base de dados, explicando o que é preciso fazer e como.

Caso de Uso:	**Buscar Base de dados**
Ator:	Web Server
Objetivo:	Gerir a ligação à Base de dados.
Visão Geral:	É feito um pedido de ligação à Base de dados, sendo que esta emite uma resposta (200OK se for bem sucedido). Após isso, fica autenticado à Base de dados, para posterior ligação ao Sistema. Qualquer alteração feita em relação à ligação à BD será aplicada aqui.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o Web Server tentar fazer a ligação com a Base de dados.	**2**	Surgirá 200OK no caso de a ligação ser bem-sucedida.
3	De seguida será feito o pedido da informação que se deseja.	**4**	Será devolvida a informação pedida.
5	O Web Server recebe essa informação e encaminha para o Sistema.	**6**	O sistema utiliza essa informação para quando esta é pedida pelo utilizador.

Tabela 3.5: Caso de uso: Buscar Base de dados

3.2.5 Obtém Mapa

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Google API.

Caso de Uso:	**Obtém Mapa**
Ator:	GUI
Objetivo:	Obter o mapa da Google para ser utilizado no Sistema.
Visão Geral:	A GUI utiliza um sistema externo, Google API, para aceder ao Google Maps. Após isto, é enviado informação ao Sistema para posterior utilização.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o GUI utiliza funções da Google API.	**2**	Acede a funções da Google de forma a obter o resultado pretendido.
3	Esse resultado é enviado para o Sistema.		

Tabela 3.6: Caso de uso: Obtém Mapa

3.2.6 Obtém GPS

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Android API's.

Caso de Uso:	**Obtém GPS**
Ator:	GUI
Objetivo:	Obter informação das coordenadas GPS do utilizador.
Visão Geral:	A GUI utiliza um sistema externo, Android API's, para aceder ao sensor GPS inserido no Sistema. Ao aceder, poderá recolher informação sobre todo o tipo de informação que este sensor fornece, como a localização do utilizador para posterior utilização do Sistema.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o GUI utiliza funções da Google API.	**2**	Acede ao sensor GPS, recebendo toda a informação fornecida pelo sensor.
3	Essa informação é enviada para o Sistema.		

Tabela 3.7: Caso de uso: Obtém GPS

3.2.7 Verifica GPS

Tal como o caso de uso "Obtém GPS", este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Google API.

Caso de Uso:	**Verifica GPS**
Ator:	GUI
Objetivo:	Verificar se o GPS do Sistema se encontra ativo.
Visão Geral:	A GUI utiliza um sistema externo, Android API's, para aceder ao sensor GPS e verificar se se encontra ativo ou não.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o GUI utiliza funções da Google API.	**2**	Acede ao sensor GPS, recebendo toda a informação fornecida pelo sensor.
		3	A informação é utilizada para saber se está ativo o sensor ou não.
4	De seguida, essa informação é enviada para o Sistema.		

Tabela 3.8: Caso de uso: Verifica GPS

3.2.8 Atualiza lista de preferências

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Android API's, utilizando funções oferecidas por este mesmo sistema.

Caso de Uso:	**Atualiza lista de preferências**
Ator:	GUI
Objetivo:	Cria ficheiros de textos para guardar informações.
Visão Geral:	A GUI utiliza um sistema externo, Android API's. Utiliza uma função do Android para criar ficheiros de textos com informação necessária para guardar a fim de ser utilizada posteriormente.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		Resposta do Sistema	
1	O caso de uso começa quando o GUI utiliza funções da Android API.	**2**	Acede a funções do Android para criar ficheiros de textos.
3	Envia informação para os ficheiros de texto.	**4**	É guardado a informação nesses ficheiros de texto.

Tabela 3.9: Caso de uso: Atualiza lista de preferências

3.2.9 Buscar direções

Este caso descreve a ligação entre a interface gráfica e dois dos sistemas externos, Android API's e Google API, utilizando funções oferecidas por estes de forma a aceder aos dados necessários.

Caso de Uso:	**Buscar direções**
Ator:	Administrador
Objetivo:	Obtém toda a informação desde distâncias, tempos e percurso a fazer entre dois pontos.
Visão Geral:	A GUI utiliza um sistema externo, Google API's. Utiliza funções do Android para recolher essa informação. É utilizado Volley e JSON para recolher essa mesma informação.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		Resposta do Sistema	
1	O caso de uso começa quando o GUI utiliza funções da Android API, enviando os dois pontos (inicio e fim do percurso).	**2**	Recebe a informação desses dois pontos e, através de funções do Android, envia essa informação utilizando URL (https://maps.googleapis.com/maps/api/directions/json/...) em que "/..." será a informação desses dois pontos.
		3	A informação recebida é tratada de forma a ser utilizada mais tarde.

Tabela 3.10: Caso de uso: Buscar direções

3.2.10 Pergunta se quer ligar GPS

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Android API's, utilizando funções oferecidas por este mesmo sistema para apresentação de uma nova janela.

Caso de Uso:	**Pergunta se quer ligar GPS**
Ator:	GUI
Objetivo:	Apresenta uma janela ao utilizador a questionar se pretende ligar o GPS.
Visão Geral:	A GUI utiliza um sistema externo, Android API's, para aceder ao sensor GPS e ao sistema de forma a aceder às definições do dispositivo móvel. Apresenta uma janela que, em caso de pretender ligar o GPS, redireciona para as definições, definições essas que poderão ser ligadas pelo utilizador.
Tipo:	Secundário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o GUI utiliza funções da Android API.	**2**	Acede a funções do Android para apresentar uma interface questionando se pretende ligar o GPS.
		3	Se pretender ligar, abrirá uma nova interface com as definições do dispositivo móvel, podendo ativar o GPS.

Tabela 3.11: Caso de uso: Pergunta se quer ligar GPS

3.2.11 Abrir Linhas

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Linhas**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com informação das linhas de Metro.
Visão Geral:	O utilizador deverá clicar no botão "Linhas", aparecendo informação sobre todas as linhas de Metro existentes. Em cada linha deverão ser apresentadas todas as estações presentes nessa linha, assim como a ordem desde o início até ao fim da linha.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		Resposta do Sistema	
1	O caso de uso começa quando o utilizador clicar no botão “Linhas”.	**2**	Uma interface é apresentada ao utilizador com todas as linhas existentes do Metro.

Tabela 3.12: Caso de uso: Abrir Linhas

3.2.12 Abrir Favoritos

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Favoritos**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com todas as estações guardadas anteriormente.
Visão Geral:	O utilizador deverá clicar no botão “Favoritos”, sendo apresentadas todas as estações guardadas anteriormente pelo utilizador. Essas estações funcionam como botões para se poder visualizar informação de circulação, em tempo real.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		Resposta do Sistema	
1	O caso de uso começa quando o utilizador clicar no botão “Favoritos”.	**2**	Uma interface é apresentada ao utilizador com todas as estações que foram guardadas pelo utilizador.
3	O utilizador poderá clicar numa das estações.	**4**	Abrirá uma nova interface com a informação de circulação, em tempo real, dessa mesma estação.
5	Poderá eliminar a estação clicando no botão “Eliminar”.	**6**	A estação é apagada dos Favoritos.

Tabela 3.13: Caso de uso: Abrir Favoritos

3.2.13 Criar Rota

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Criar Rota**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador para criar a Rota, desde a sua Origem até ao Destino pretendido.
Visão Geral:	O utilizador deverá clicar no botão "Criar Rota", sendo apresentadas duas opções de escolha, a estação de Origem (onde se encontra) e a estação de Destino (onde se pretende deslocar). No fim terá de clicar no botão "Pesquisar", obtendo o resultado mais favorável.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão "Criar Rota".	**2**	Uma interface é apresentada ao utilizador com duas opções, podendo escolher a estação de Origem e Destino.
3	O utilizador deverá escolher a estação Origem.	**4**	A estação Origem fica selecionada.
5	O utilizador deverá escolher a estação Destino.	**6**	A estação Destino fica selecionada.
7	O utilizador deverá clicar no botão "Pesquisar".	**8**	Será apresentado o/os caminho/os a percorrer para chegar ao Destino final.
		9	As estações apresentadas à esquerda (origens dos percursos) funcionam como botões, apresentado a informação de circulação, em tempo real.

Tabela 3.14: Caso de uso: Criar Rota

3.2.14 Abrir Mapa

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Mapa**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com o mapa e a sua localização GPS.
Visão Geral:	O utilizador deverá clicar no botão "Mapa" e, em caso de sucesso, abrirá o mapa da Google com a sua localização GPS sinalizada no mapa. Para que o caso seja de sucesso, terão de ocorrer casos de uso externos tal como apresentado na figura 3.3 de casos de uso.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão “Mapa”.	**2**	Abrirá depois dos sistemas externos serem chamados e serem bem-sucedidos.
		3	Uma interface é apresentada ao utilizador com o mapa da Google.

Tabela 3.15: Caso de uso: Abrir Mapa

3.2.15 Abrir estação

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir estação**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com a informação de circulação, em tempo real.
Visão Geral:	O utilizador deverá clicar numa das estações apresentadas no mapa, apresentando-se uma janela com toda a informação de circulação dessa mesma estação.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar numa estação apresentada no Mapa.	**2**	Será utilizado um caso de uso externo para aceder à Base de dados.
		3	Uma interface é apresentada ao utilizador com informação de circulação da estação, em tempo real.

Tabela 3.16: Caso de uso: Abrir estação

3.2.16 Guardar estação

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Android API's, utilizando funções oferecidas por este mesmo sistema.

Caso de Uso:	Guardar estação
Ator:	Utilizador
Objetivo:	Guarda estações nos Favoritos.
Visão Geral:	Ao abrir uma estação do mapa, abrirá uma janela onde será apresentado um botão "Guardar". Ao clicar no botão, poderá guardar estações nos Favoritos de forma a facilitar a sua visualização.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão "Guardar".	**2**	A estação é guardada nos Favoritos.
		3	Poderá abri-la quando aceder ao Menu "Favoritos".

Tabela 3.17: Caso de uso: Guardar estação

3.2.17 Calcular distância

Este caso descreve a ligação entre a interface gráfica e utilizador, apresentando dados fornecidos por casos de uso externos.

Caso de Uso:	Calcular distância
Ator:	Utilizador
Objetivo:	Apresenta uma opção ao utilizador para calcular distância, o tempo que demora a percorrer e um caminho possível.
Visão Geral:	É apresentada uma opção ao utilizador para selecionar o local para onde pretenda deslocar-se. De seguida é apresentada a distância em quilómetros, o tempo em horas e minutos e um caminho possível.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador pressiona cerca de dois segundos no local pretendido.	**2**	É apresentado um *marker* no mapa azul, indicando que o local ficou selecionado.
3	De seguida deve clicar no botão “carro” ou “a pé” para calcular o caminho.	**4**	São utilizados Casos de uso externos da GUI.
		5	Após serem executados os Casos de uso externos, é apresentado um caminho no mapa, a distância e o tempo que demora a percorrer.

Tabela 3.18: Caso de uso: Calcular distância

3.2.18 Procurar estações

Este caso descreve a ligação entre o utilizador e a interface gráfica, utilizando funções do Android para pesquisar a estação procurada pelo utilizador.

Caso de Uso:	**Procurar estações**
Ator:	Utilizador
Objetivo:	Apresenta uma opção ao utilizador para procurar pelas estações existentes.
Visão Geral:	É apresentada uma opção ao utilizador para procurar pelas estações existentes no mapa, escrevendo parte do nome são fornecidas dicas de possíveis nomes de estações. De seguida, o mapa desloca-se para a posição dessa mesma estação.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador escreve o nome da estação.	**2**	São apresentadas dicas de possíveis estações enquanto escreve o nome.
		3	Fica selecionada a estação escrita.
4	O utilizador deverá clicar no botão “Pesquisar”.	**5**	O mapa deslocar-se-á para a posição GPS da estação selecionada, ficando esta precisamente no centro do mapa.

Tabela 3.19: Caso de uso: Procurar estações

3.2.19 Escolha idioma

Este caso descreve a ligação entre a interface gráfica e um dos sistemas externos, Android API's, utilizando funções oferecidas por este mesmo sistema.

Caso de Uso:	**Escolha idioma**
Ator:	Utilizador
Objetivo:	Apresenta uma opção ao utilizador para escolher o idioma.
Visão Geral:	É apresentada uma opção ao utilizador para escolher o idioma, sendo este idioma apresentado durante toda a aplicação. Numa próxima vez que abrir a aplicação, esse idioma ficará como pré-definido, sendo possível alterá-lo se o utilizador o desejar.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar na aplicação.	**2**	Será apresentada uma interface da aplicação, com a opção de escolha de idioma.
3	O utilizador poderá escolher um dos idiomas apresentados.	**4**	O idioma escolhido ficará selecionado.
4	O utilizador deverá clicar no botão seguinte.	**5**	Será apresentada uma nova interface com todos os menus disponíveis.

Tabela 3.20: Caso de uso: Escolha idioma

3.2.20 Abrir Sobre

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Sobre**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com informação sobre a aplicação.
Visão Geral:	O utilizador deverá clicar no botão "Sobre", aparecendo informação sobre a aplicação, o porquê da sua criação e por quem foi criada.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão "Sobre".	**2**	Uma interface é apresentada ao utilizador com informação sobre a aplicação.

Tabela 3.21: Caso de uso: Abrir Sobre

3.2.21 Abrir Contactos

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Contactos**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com os contactos disponíveis.
Visão Geral:	O utilizador deverá clicar no botão “Contactos”, aparecendo informação de todos os contactos disponíveis e que poderá utilizar em caso de alguma anomalia.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão “Contactos”.	**2**	Uma interface é apresentada ao utilizador com todos os contactos disponíveis.

Tabela 3.22: Caso de uso: Abrir Contactos

3.2.22 Abrir Manual

Este caso descreve a forma como a interface gráfica comunica com o utilizador, apresentando o resultado esperado.

Caso de Uso:	**Abrir Manual**
Ator:	Utilizador
Objetivo:	Apresenta uma janela ao utilizador com informação sobre as funcionalidades da aplicação.
Visão Geral:	O utilizador deverá clicar no botão “Manual”, aparecendo informação sobre as funcionalidades da aplicação, como é utilizada e que informação é fornecida.
Tipo:	Primário

Percurso típico dos eventos			
Ação do Ator		**Resposta do Sistema**	
1	O caso de uso começa quando o utilizador clicar no botão “Manual”.	**2**	Uma interface é apresentada ao utilizador informação sobre as funcionalidades da aplicação.

Tabela 3.23: Caso de uso: Abrir Manual

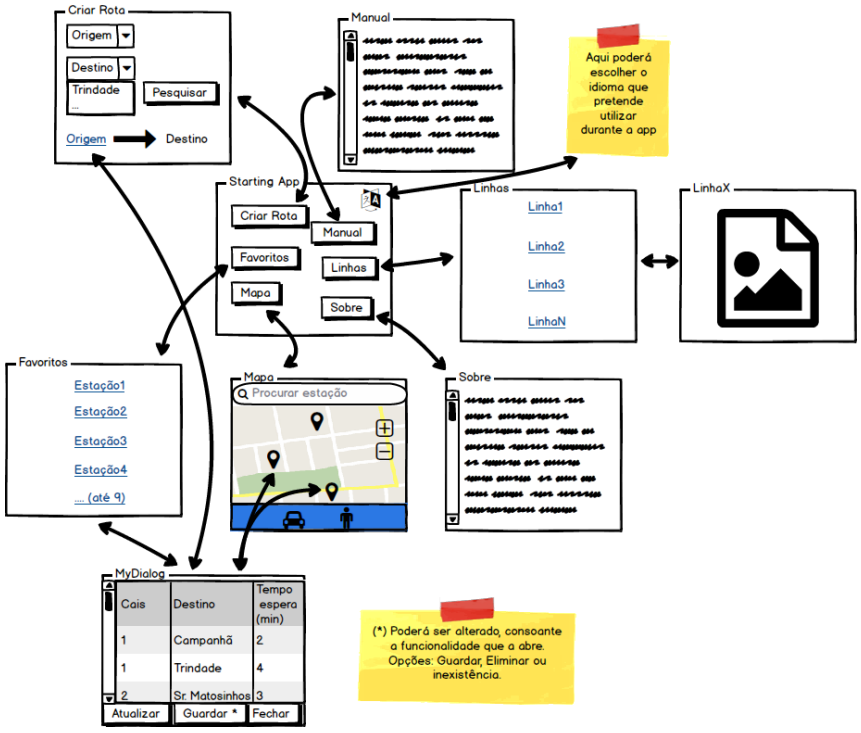


Figura 3.4: Mockup Inicial da Aplicação

3.3 Design da Aplicação

O sistema a desenvolver nesta dissertação destina-se a ser utilizado diretamente por utentes do serviço de Metro, pelo que, alguns dos aspetos a ter em conta no desenvolvimento deste projeto, são a usabilidade e o aspeto gráfico. Em particular procurou dar-se prioridade à simplicidade da sua interface gráfica, tal como já foi referido, mas também criar uma interface que fosse atraente para os seus utilizadores. Para se conseguir alcançar estes objetivos sem ter a necessidade de reescrever código, tomou-se a decisão de se começar por desenvolver *mockups*, capazes de ilustrar de uma forma sugestiva a funcionalidade e operação do sistema. Tal, permitiria detetar aspetos menos positivos de usabilidade do sistema, rapidamente introduzir modificações e verificar a adequação dessas alterações aos objetivos em vista.

Para além disso, visto que existem no mercado um conjunto enorme de aplicações gráficas para Android desenvolvidas por equipas de profissionais incorporando "designers" gráficos, optou-se por efetuar uma análise inicial às estruturas e abordagens gráficas, mais utilizadas em aplicações para a plataforma Android.

A primeira abordagem adotada está esquematizada no *mockup* apresentado na figura 3.4, o qual apresenta as funcionalidades pretendidas para o sistema e a forma como são oferecidas ao utilizador. Estas funcionalidades serão explicitadas mais à frente, na apresentação do produto final.

Embora esta primeira abordagem fosse relativamente simples, considerou-se que seria possível

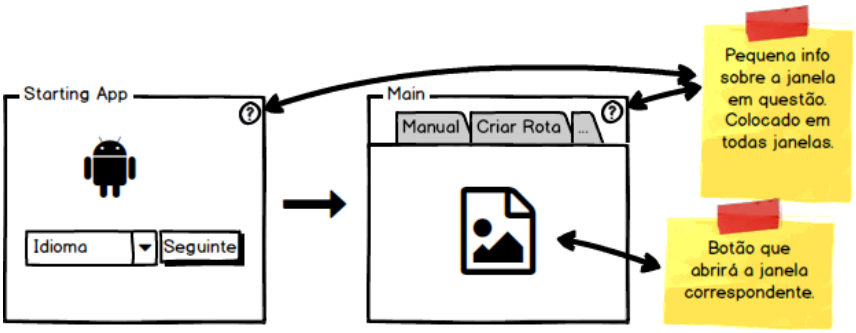


Figura 3.5: Parte do Mockup modificado

chegar a uma solução mais atrativa e na qual as funcionalidades seriam expostas ao utilizador de uma forma mais organizada, facilitando a navegação. Na figura 3.5 é possível observar as alterações que foram realizadas no *mockup* da figura 3.4.

A app inicializa com uma imagem e um *Combo-Box*, podendo-se selecionar o idioma que se pretende. Na fase seguinte, será apresentado um tipo de TabLayout, em que cada *tab* corresponde ao que anteriormente seria um botão. Em cada *tab* é apresentada uma imagem clicável, correspondente à funcionalidade pretendida. Para além disso, cada janela apresenta um ícone de informação no canto superior direito dando indicação de algumas notas sobre as funcionalidades apresentadas. De resto foi mantido a ideia do *Mockup* inicial, não havendo alterações significativas a retratar.

Capítulo 4

Implementação do Sistema

O sistema está dividido em dois sub-sistemas, parte servidor ou back-end que promove a ligação com a base de dados da Efacec e parte cliente instalada no smartphone android, que interage com APIs externas e com os sensores do smartphone.

Este capítulo inclui os dois sub-sistemas necessários para o desenvolvimento do projeto, sendo estes os referidos anteriormente. É feito uma referência ao código desenvolvido e a forma como foi estruturado cada sub-sistema.

4.1 Sub-sistema Base de dados - Processo

Este sub-sistema foi concebido para a ligação à Base de Dados Oracle, de forma a obtermos a informação necessária para ser utilizada posteriormente, pela aplicação Android. Utiliza PHP como linguagem de programação. O referido sub-sistema é colocado no Web Server Apache, tendo um URL específico para poder ser acedido.

Este processo está descrito na figura 4.1 na qual o sub-sistema baseado no Slim Framework aparece integrado no servidor Apache. É um processo que assegura um ambiente Web estável mas, como seguinte dessa segurança, cada pedido HTTP é isolado dos outros, não permitindo troca de informação entre pedidos. Ou seja, cada processo filho (cliente) realiza a tarefa pedida e, quando terminar, é destruído. São esses processos filhos que vão aceder à Base de dados e extrair a informação necessária.

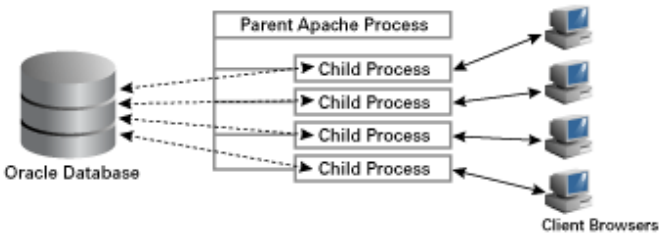


Figura 4.1: Ligação da Base de dados aos vários clientes [8]

Para se realizar a ligação com a Base de dados, foi utilizado funções do PHP. Inicialmente foi também necessário inserir os dados da conta utilizada da Base de dados, tal como descrito em baixo.

```
$username_db = "xxxxxx";
$password_db = "xxxxxx";
$database = "(DESCRIPTION=
            (ADDRESS=(PROTOCOL=TCP)(HOST=xxxx)(PORT=xxxx))
            (ADDRESS=(PROTOCOL=TCP)(HOST=xxxx)(PORT=xxxx))
            (CONNECT_DATA=
                  (SERVER=DEDICATED)
                  (SERVICE_NAME=xxxx)
            )
)";
```

Por fim, foi utilizada a função "oci_connect()" do PHP para efetuar a ligação.

Foram ainda criadas algumas *queries* para obter a informação pretendida. Serão apresentadas duas dessas *queries* como demonstração.

A primeira fornece informação das estações da rede de Metro: ID, nome, nível e coordenadas; A segunda fornece informação de circulação para uma estação específica: plataformas, tempo de espera e destino. Esta informação é obtida através do envio do ID da estação em questão.

1ª Query:

```
SELECT T_ESTACOES.fn_id as id,
            T_ESTACOES.ft_nome as name,
            T_ESTACOES.FT_EXTERNAL_LEVEL as Statlevel,
            T_IG_STATIONSSIP.fn_geox as gX,
            T_IG_STATIONSSIP.fn_geoy as gY,
            T_IG_STATIONSSIP.FT_GUI_NAME as extname
FROM T_ESTACOES
LEFT JOIN T_IG_STATIONSSIP ON
            T_ESTACOES.fn_id=T_IG_STATIONSSIP.fn_id_station
ORDER BY T_ESTACOES.fn_id
```

2ª Query:

```
SELECT FN_IDESTACAO as id ,
            FN_PLATAFORMA as platform,
            FN_TEMPOCHEGADA as timeleft,
```

FT_DESTINATION as destiny
FROM t_circulacao
WHERE fn_servicoPassageiros= 1
AND fn_paraNaEstacao=1
AND FN_IDESTACAO='$nome'
ORDER BY FN_PLATAFORMA, FN_TEMPOCHEGADA

A variável "$nome" corresponde ao ID da estação.

Tal como foi já referido anteriormente, o Web Server foi instalado numa máquina que, por sua vez, foi utilizado como repositório para o Slim Framework. A forma de aceder é através de um URL, indicando o caminho a percorrer desde o IP da máquina até ao ficheiro PHP do Slim. Para o primeiro caso das 'queries', o URL utilizado é o seguinte:

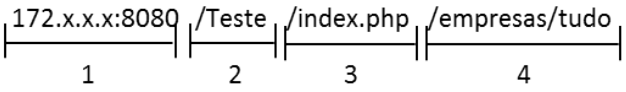


Figura 4.2: Exemplo de um URL utilizado

Na Figura 4.2 está descrito um exemplo de um URL utilizado. Cada ponto corresponde:

1. IP da máquina utilizada;
2. Pasta onde foi inserido o Slim Framework;
3. Ficheiro PHP do Slim que foi desenvolvido;
4. Nome da função e sub-função criadas para distinguir qual a *query* que pretende ser utilizada.

4.2 Sub-sistema Android

4.2.1 Diagrama de classes

Esta secção descreve as classes que foram desenvolvidas para implementar o sub-sistema Android, de forma a oferecer as funcionalidades identificadas durante a especificação funcional do sistema. O conjunto de classes desenvolvidas está ilustrado na figura 4.3.

A classe *BeginActivity* é a que inicia quando a aplicação arranca. Tem como funcionalidade a escolha do idioma que se pretende utilizar durante a execução da aplicação, tendo disponível de momento, apenas dois idiomas, português e inglês.

A classe *MainActivity* é executada imediatamente a seguir à classe *BeginActivity*. É considerada a classe principal, pois todas as outras classes divergem a partir desta. Esta apresenta todas as funcionalidades existentes.

A classe *RouteActivity* tem como funcionalidade a possibilidade de o utilizador escolher a rota que pretende efetuar e receber a informação necessária para chegar ao seu destino final. É utilizada a função getLines(int FN_ID) que irá aceder, através do *SLIM framework*, à tabela T_LINES da Base de Dados. Aí, será verificado se a linha de origem e de destino, são iguais ou diferentes. Caso sejam diferentes, esta deverá encontrar qual/quais as estações em comum e escolher a que se encontra mais próxima da estação de origem para, no fim, apresentar a solução pretendida.

A classe *MapsActivity* apresenta o mapa da Google, já referido anteriormente neste documento, onde irão ser inicializados todos os *markers*, um para cada estação. A informação de cada estação será obtida através da função getStations() que irá aceder ao *SLIM framework*, "chamando" a classe *Connection*. Associada a esta classe existe uma outra classe denominada *DialogActivity* que apresentará, em tempo real, a informação de circulação da estação assinalada pelo utilizador. Esta informação é conseguida através da utilização da função getForecast(ID_STATION) que irá aceder ao *SLIM framework* tal como na função anterior. Uma outra funcionalidade apresentada nesta classe é a possibilidade de pesquisar uma estação específica, introduzindo o nome, ou parte do nome, na caixa de texto disponível para o efeito. Quando o operador tiver introduzido o nome da estação, o ecrã do dispositivo móvel mover-se-á para a posição GPS dessa mesma estação.

A classe *FavActivity* apresentará as estações pré-definidas pelo utilizador de forma a facilitar o acesso à informação pretendida, sem existir a necessidade de procurar essa estação no Mapa, sempre que se pretende voltar a visualizar informação dos metros. Por uma questão de gestão da aplicação, nesta fase, foi opção limitar a memorização até um máximo de 10 registos. É utilizada a função 'SharedPreferences' fornecida pelo Android que cria um ficheiro no próprio dispositivo móvel e que apenas é possível ser utilizado pela app, que criou esse mesmo ficheiro. Caso a app seja desinstalada, esse ficheiro será automaticamente eliminado do dispositivo. Esta função funciona como um ficheiro xml. Quando é apresentada ao utilizador a informação em tempo real da estação escolhida pelo próprio, é apresentado no canto inferior direito um botão "Guardar", que permite guardar essa estação nos 'Favoritos'. A informação depois poderá ser consultada no *MainActivity*, aquando do clique do botão "Favoritos".

A classe *LinesActivity* apresentará todas as linhas existentes. A forma de apresentar cada linha será através de uma outra classe associada *ShowLineActivity* que exibe a linha completa, com todas as estações desde a partida até à chegada à estação final. No *layout* da classe *LinesActivity* será apresentado um botão para cada linha sendo que, quando clicado, apresentará a informação dessa mesma linha. A forma de apresentação desta informação será realizada através de uma imagem (ficheiro '.png') carregado na aplicação, ilustrando todas as estações e o seu caminho desde a estação inicial até à final.

A classe *ConnectionDetect* foi criada com a finalidade de ser utilizada por várias outras classes, sendo chamada a função isConnectingToInternet() para verificar se o dispositivo móvel tem alguma ligação à Internet ativa (Wi-Fi ou dados móveis). Esta função é importante devido ao facto de a aplicação precisar de ligação à Internet para funcionar.

As classes *AboutActivity* e *ContactsActivity* são apenas informativas, apresentando uma pequena explicação da motivação para a criação desta aplicação e os contactos para a comunicação

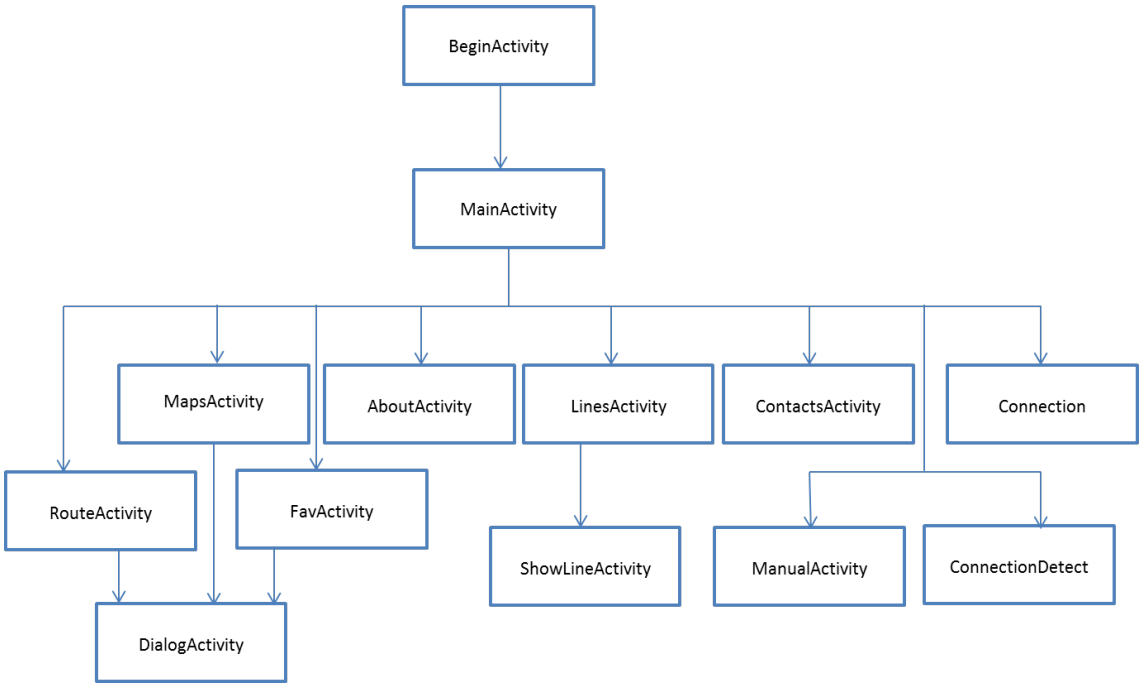


Figura 4.3: Diagrama de classes

de algum problema ou esclarecimento de dúvidas, respetivamente.

Por fim, temos o *ManualActivity* que explica de forma resumida a aplicação, o que é e que funcionalidades são fornecidas. Este, foi elaborado tendo em conta as dúvidas que poderão surgir ao utilizador durante a exploração da aplicação, utilizando-se como método, o de pergunta-resposta.

4.2.2 Especificação das classes

Cada classe na sua maioria tem um ficheiro .xml associado, quando é preciso apresentar um *layout*, ou seja, uma janela de interação com o utilizador. Esse ficheiro .xml serve para formatar essa janela apresentada, chamando-se de *layout* estático. Denomina-se de layout dinâmico, a formatação que é feita diretamente na classe, havendo exclusivamente código. Deve-se ter em atenção que um tipo não exclui o outro, podendo ser utilizados os dois tipos de *layout* em simultâneo para a mesma janela. Um exemplo disso será descrito na atividade "MyDialog".

- BeginActivity

Tal como explicado anteriormente, a funcionalidade principal desta atividade é a escolha do idioma para toda a utilização da app. Para tal é utilizada uma função do tipo SharedPreferences [27] disponibilizada pelo Android. Esta função cria um ficheiro no próprio dispositivo móvel, podendo ser guardado valores (tanto strings como variáveis numéricas). Cada valor é associado a

um objeto. A forma de pesquisar por esses valores é através do nome do objeto.

Objeto	Valor
Estação1	Trindade
Estação2	Bolhão
Total	86

Tabela 4.1: Exemplo da organização da informação da função SharedPreferences

Na tabela 4.1 é apresentado um exemplo da organização da informação. Para extrair essa informação utiliza-se a função getString(), por exemplo getString(Total), obtendo o valor 86.

- MainActivity

O que se destaca desta atividade é a criação de um TabLayout em que é utilizado uma extensão do Android FragmentStatePagerAdapter [28]. Esta extensão cria três funções, uma que indica o número total de *tabs*, outra em que é colocado o nome de cada *tab* e, por fim, uma outra que cria as *tabs*. Cada *tab* tem uma imagem, variável ImageButton [29] fornecida pelo Android que, após clicado, abrirá a atividade associada.

- ManualActivity / AboutActivity / ContactsActivity

Apenas foram criados *layouts* (xml) para que a informação fosse apresentada com formatação e organização. O texto descrito é apenas informativo, com algumas indicações de utilização e/ou informação sobre a app em si.

```
<TextView
        android: layout_width= "wrap_content
        android: layout_height= "wrap_content
        android: textAppearance= "?android:attr/textAppearanceLarge
        android: id= "@+id/tabout
        android: textSize= "18dp
        android: text= "@string/textAbout
/>
```

Este excerto de código é aqui apresentado para ilustrar a forma como o texto é formatado. As duas primeiras linhas de código indicam que o espaço ocupado na janela será ajustado ao tamanho do texto, a terceira linha corresponde ao aspeto do texto, a quarta linha é um ID utilizado para o

caso de se querer fazer alguma alteração na classe, a quinta corresponde ao tamanho do texto e, por fim, na última linha o nome da string com o texto que será apresentado.

- LinesActivity / ShowLineActivity

Para a apresentação das imagens de cada linha, foi criada um tipo de função a que se chamou de TouchImageView. Para isso foi criada uma classe com esse nome e a forma de utilização será criar uma variável deste tipo. O motivo desta opção deve-se à necessidade de existir zoom na imagem, tanto de aproximação como de afastamento (zoom in e zoom out).

Esta funcionalidade é considerada uma extensão do ImageView [30], suportando todas as funcionalidades deste.

Exemplo:
TouchImageView img = new TouchImageView(this);

- FavActivity / MyDialog

As principais funções utilizadas para esta atividade já foram referidas anteriormente, como o SharedPreferences.

Para ser possível abrir uma determinada estação, foi necessário criar uma classe chamada *MyDialog*. Esta classe é uma extensão do DialogFragment [31]. Esta extensão permite que uma janela não tenha de ser fechada para que outra seja apresentada, sobrepondo-se uma sobre a outra. Isto faz com que a informação da janela que se encontra debaixo da sobreposta, não seja perdida. Por vezes, é bastante útil quando a informação não precisa de ser constantemente atualizada.

Exemplo:
MyDialog myDialog = new MyDialog(this);
myDialog.show(manager, "MyDialog);

Neste exemplo poderemos constatar a forma de ser chamada a função, para que seja apresentada a nova janela sobre a atividade em questão. Esta nova janela utiliza dois novos tipos de *layout*, TableLayout [32] e TableRow [33]. Estes foram necessários para criar as tabelas que apresentam a informação de circulação, em tempo real. Estas tabelas são o exemplo de que os layouts estático e dinâmico podem coexistir. Para a criação das tabelas foi necessário fazer código, visto não saber logo à partida o número específico de linhas, sendo este um layout dinâmico. A apresentação da tabela foi feito no ficheiro .xml, sendo este um layout estático.

- MapsActivity

Nesta atividade, várias funções foram criadas de forma a tornar possível a colocação em prática de determinadas funcionalidades.

As funções principais são: onZoom(), onSearch(), changeType(), showAllStations(), getCurrentLocation() e drawPath().

A onZoom(), tal como o nome indica, foi criada para existir uma interação entre o utilizador e a interface gráfica, podendo aproximar ou afastar o mapa. Ao aproximar, é facultada a visualização mais pormenorizada das ruas assim como, ao afastar, a visualização torna-se mais abstrativo.

A onSearch() serve para procurar por estações existentes no mapa. Foi utilizado o AutoCompleteTextView [34] para fornecer dicas ao utilizador quando este estiver a escrever o nome duma estação.

A changeType() tem como funcionalidade o facto de mudar o tipo de mapa, oferecendo ao utilizador a hipótese de escolha entre via satélite ou vai terrestre.

A função showAllStations() é uma das mais importantes funções, pois é a que implementa no mapa os *markers* [35], cada um indicando uma estação de metro diferente. Estes *markers* são objetos que surgem sobre o mapa com a forma de uma espécie de balão, sendo possível clicar no mesmo. Ao clicar, abrirá uma nova janela através da classe *MyDialog*, tal como referido anteriormente.

A getCurrentLocation() serve para utilizar o sensor GPS do dispositivo móvel para disponibilizar essa informação no mapa. É apresentado no próprio mapa a posição GPS do utilizador. Para isso utiliza-se a função do Android, Location [36].

Por fim, a função drawPath() é utilizada para desenhar no mapa o caminho a percorrer desde a posição GPS até um ponto escolhido pelo utilizador. Para tal utiliza-se API's da Google, funcionando como um URL [37].

O URL deverá ser da forma:

https://maps.googleapis.com/maps/api/directions/json? *origin=x1,x2&destination=y1,y2* **&mode=walking &alternatives=true** &key=YOUR_API_KEY

O sublinhado corresponde ao link que deverá ser acedido. O itálico (*origin*) representa a origem em que 'x1' e 'x2' indicam a latitude e longitude, respetivamente. O itálico (*destination*) representa o destino em que 'y1' e 'y2' indicam a latitude e longitude, respetivamente. O negrito corresponde ao modo de deslocação que pretende utilizar (neste exemplo está assinalado a viagem a pé) e se pretende alternativas para o percurso. O restante corresponde à chave que deve ser gerada no início da criação da app.

Através da função JSONArray [38], é possível extrair a informação necessária. Neste caso, foi extraída a informação da distância, do tempo que demora a percorrer e uma string constituída por

uma série de coordenadas codificadas, utilizando a codificação PolyLine. Esta codificação é constituída pela diferença entre um par de coordenadas que são convertidas para ASCII, diminuindo de forma significativa o tamanho dos dados.

Para esta codificação é preciso percorrer os seguintes passos [39]:

1. Multiplicar as coordenadas por 10^5 (exemplo: $-179,9832104 * 10^5 = -179998321$)
2. Converter o valor decimal para binário. Utilizando o exemplo, obtemos:

   00000001 00010010 10100001 11110001

   11111110 11101101 01011110 00001110

   11111110 11101101 01011110 00001111
3. Fazer uma deslocação para a esquerda um bit do valor binário (*left-shift*). Obtemos:

   11111101 11011010 10111100 00011110
4. No caso do valor decimal original for negativo devemos inverter, obtendo:

   00000010 00100101 01000011 11100001
5. Dividir o valor binário em pedaços de 5 bits (começando do lado direito). O valor obtido é:

   00001 00010 01010 10000 11111 00001
6. Fazer a operação matemática *OR* com 0x20 se outro pedaço de bit seguir, obtendo:

   100001 111111 110000 101010 100010 000001
7. Converter cada valor em decimal:

   33 63 48 42 34 1
8. Somar 63 a cada valor:

   96 126 111 105 97 64
9. Por fim, converter este valor para o ASCII equivalente:

   ‘ oia@

Quando recebida esta string, será necessário descodificá-la de novo, para ser possível depois fazer o desenho do caminho no mapa. Para isso foi necessário seguir os passos anteriores, mas pela ordem inversa.

- RouteActivity

A particularidade desta atividade centra-se no código desenvolvido. A ideia inicial desta atividade seria proporcionar ao utilizador a possibilidade de obter uma rota, desde uma estação de metro até uma outra, mesmo sendo de linhas diferentes. No entanto, verificou-se que a informação fornecida pela Base de dados não era suficiente para o desenvolvimento desta atividade.

Assim, para resolver este constrangimento foi criada uma função que disponibiliza todos os cruzamentos que existem entre linhas, indicando a(as) estação(ões) de metro resultante(es) desse(s) cruzamento(s) e a respetiva posição.

Quando recebida a informação da estação de Origem e de Destino, selecionadas pelo utilizador, é verificado na função se são da mesma linha. Se forem, a informação é disponibilizada de imediato, se não, a função verifica a que linhas pertencem as duas estações introduzidas e, devolve a informação da existência de cruzamentos entre elas. Considerou-se como profundidade igual a 0 a correspondente a uma troca de linhas de metro (um cruzamento).

Considerou-se como profundidade igual a 1 a correspondente a uma troca de duas linhas de metro (dois cruzamentos). Nesta situação considera-se a não existência de cruzamentos entre a Origem e o Destino. Assim, a função irá verificar todos os cruzamentos que existem entre as linhas possíveis de destino e as restantes em que a estação Destino não está presente. Depois, compara os resultados obtidos com as linhas possíveis de origem, onde está presente a estação Origem. Este código foi limitado a apenas dois cruzamentos, pois após uma pesquisa de algumas redes de metro, verificou-se que por norma este número não é ultrapassado, ou seja, durante uma viagem de metro raramente se verificam mais de dois cruzamentos.

Por fim, como poderão existir mais do que uma estação de metro em que haverá cruzamento entre duas linhas, é necessário calcular qual a que fica mais próxima da estação de Origem para evitar que o utente tenha de percorrer um maior percurso. Para isso é realizada uma subtração do valor absoluto entre a posição da estação Origem e a estação de cruzamento i e comparada com a estação de cruzamento i+1, devolvendo o resultado mais pequeno. Depois, a informação que é guardada é comparada com a seguinte e por aí adiante, até se ter obtido todos os resultados dos cruzamentos existentes.

- ConnectionDetect

Foi utilizada uma função do Android, ConnectionManager [40] que permite verificar se existe algum tipo de ligação à Internet, quer seja por Wi-Fi ou por dados móveis.

- Connection

Para realizar a ligação entre o *Slim* e o Android é utilizada uma função do Android chamada URL [41]. Esta função recebe o URL a que pretende aceder e, através da linha de código

"url.openConnection()", esta faz a ligação. Para verificar se a ligação foi bem sucedida, utiliza-se a linha de código "getResponseCode()". No caso de o valor obtido ser 200ok significa que não houve nenhuma ocorrência inesperada. Caso exista alguma ocorrência inesperada a função devolve outros valores diferentes do anteriormente indicado.

De seguida é utilizada a função BufferedReader [42] para ler a informação recebida do *Slim Framework*, tornando possível depois ler essa informação como JSON e obter os valores esperados.

- Mensagens de aviso

Para a verificação desta funcionalidade foi utilizada uma função bastante útil do Android, alertDialog [43]. Esta função permite uma interação entre a GUI e o utilizador, disponibilizando uma janela de aviso sobre algum problema encontrado. Quando se verifica a necessidade de ter o GPS ligado para obter a localização e este não está ativo, a aplicação questiona o utilizador, sugerindo a sua ligação. Também quando o utilizador pretende fechar a aplicação é disponibilizada uma janela questionando se é mesmo essa a ação que pretende efetuar.

4.2.3 Interligação entre Casos de uso e Diagrama de classes

Os casos de uso referenciados no capítulo anterior foram utilizados como base para a obtenção dos módulos Slim Framework e Android.

Relativamente ao módulo *Slim Framework* a perceção é que este será gerido pelo ator admin. Os casos de uso por ele praticado, "Gerir a Base de dados", "Atualizar a Base de dados" e "Gerir o Web Server" foram essenciais para a criação e manutenção deste módulo. Para o funcionamento do Web Server tornou-se necessário instalá-lo numa máquina e configurá-lo. Este Web Server é considerado um repositório onde foi colocado o *Slim Framework*. Durante a fase de implementação das *queries* necessárias, foram realizadas várias alterações e, por isso, gerir o Web Server foi fundamental. Como a Base de Dados não continha toda a informação necessária, foi necessário alterá-la para mais tarde ser possível fornecer essa informação ao utilizador. Daí o surgimento dos casos de uso "gerir e atualizar a Base de Dados".

Para tornar esta aplicação Android *user-friendly*[1], foram testadas várias interfaces. Nesta fase inicial tivemos premente a necessidade de construirmos uma aplicação simples, com excelente usabilidade e apresentando informação precisa. Durante o desenvolvimento houve necessidade de introduzir alterações com relativo grau de complexidade, mas estas, tiveram sempre subjacentes os princípios enumerados anteriormente.

A classe *BeginActivity* é a que surge mal a aplicação arranca, permitindo a possibilidade de ser escolhido o idioma (neste projeto limitada ao português ou inglês), através da ativação do caso

[1] fácil de usar, amigável para o usuário

de uso "Escolha Idioma", estabelecendo assim a linguagem de interação entre o utilizador e a interface gráfica. Tivemos presente que a aplicação poderá ser utilizada por um cidadão nacional ou por um cidadão estrangeiro, pelo que, a escolha do inglês com segunda língua, teve subjacente essa possível utilização.

A classe *MainActivity* é das que proporciona maior número de casos de uso, visto ser a classe que tem ligação com muitas outras. A necessidade de utilização do mapa correspondente à área de implantação do metro, levou à necessidade de criação de um botão nesta mesma classe que abrirá uma outra classe (*MapsActivity*). O caso de uso presente é o "Abrir Mapa". Foram inseridos botões para criar rotas (*RouteActivity*), para abrir os favoritos (*FavActivity*), para abrir informação sobre a aplicação (*AboutActivity*), para abrir os contactos (*ContactsActivity*), para abrir o manual (*ManualActivity*) e para abrir as linhas de metro (*LinesActivity*) sendo os casos de uso respetivamente "Criar Rota", "Abrir Favoritos", "Abrir Sobre", "Abrir Contactos", "Abrir Manual" e "Abrir Linhas".

Quando aberta a classe *MapsActivity*, são apresentadas ao utilizador algumas funcionalidades como procurar por estações, abrir uma estação e até mesmo calcular um percurso. Para esta função, existem três casos de uso distintos, respetivamente "Procurar estações", "Abrir estação" e "Calcular distância". A introdução destes casos de uso deveu-se ao facto de considerarmos importante dar ao utilizador a possibilidade deste se situar e encontrar soluções de mobilidade dentro das disponibilizadas pela aplicação, através de um simples clique no ecrã do dispositivo móvel. O clique por exemplo numa estação existente no mapa abrirá a classe *DialogActivity* apresentando toda a informação de circulação dessa estação, em tempo real. Ao utilizador será dado a possibilidade de guardar essa estação para utilizações futuras, através de um botão apresentado. Com isso, o caso de uso "Guardar estação" é invocado.

Existem alguns casos de uso externos que são atuados pela interface gráfica que surge com base em algumas ações realizadas pelo utilizador. Quando o utilizador clica no botão "Mapa", a aplicação verifica de imediato se o GPS está ou não ligado (caso de uso "Verificar GPS") e, no caso de este estar desligado, gera uma mensagem a perguntar se o pretende ligar, só sendo possível prosseguir se o fizer. Esta mensagem é apresentada pela GUI, sendo o caso de uso "Pergunta se quer ligar GPS". Se o utilizador aceitar, aparecerá o mapa e a sua localização GPS.

A GUI, para obter este resultado, utiliza API's externas (Google e Android) para oferecer essa informação ao utilizador, sendo os casos de uso utilizados "Obtém mapa" e "Obtém GPS", respetivamente. Após o aparecimento do mapa, é disponibilizada uma funcionalidade que permite calcular distâncias, tal como referido anteriormente. Para que seja obtida esta informação, a GUI utiliza um sistema externo, a Google API. Para isso, existe o caso de uso "Buscar direções".

Uma outra funcionalidade, quiçá a razão de existência desta aplicação, é a possibilidade de se poder abrir estações para visualizar a informação de circulação em tempo real e, é nesse momento que atua o Web Server. O Web Server liga-se à Base de dados, extrai a informação pedida pelo Sistema e devolve-lhe essa mesma informação. Daí a importância do caso de uso "Buscar Base de dados".

Por fim, temos a classe *FavActivity* que apresenta todas as estações guardadas pelo utilizador.

Para obtermos esta informação, é gerada uma lista de preferências que guarda todas as estações que o utilizador indicar. Esta lista de preferências é criada por um dos sistemas externos da GUI, Android API's. O caso de uso ligado a este acontecimento é "Atualiza lista de preferências".

Capítulo 5

Produto Final - Testes do Sistema e Resultados

Neste capítulo serão apresentados os resultados obtidos após a implementação do código, demonstrando as funcionalidades e a forma como é apresentada a interface gráfica. Para haver uma verificação dos resultados, foram realizados testes durante a utilização da aplicação e que serão também explicitados.

5.1 Pré-Requisitos

Para dar início aos testes, deve-se garantir à partida algumas condições, entre elas destacam-se as seguintes:

1. Antes de iniciar qualquer tipo de teste, deve verificar-se se a ligação à Base de dados está em funcionamento. Para isso, deve ser verificado se a máquina onde foi instalado o Apache http Server está ativa e se o Servidor Apache se encontra 'arrancado'. Se sim, verificar a posteriori se o SLIM Framework está a fazer a ligação à Base de Dados corretamente. Caso alguma destas situações não ocorrer, deve-se verificar o problema, sendo o mais provável:
    - A máquina em causa estar desligada
    - O Servidor Apache estar desligado
    - A Base de Dados estar Off
2. O dispositivo móvel deverá estar ligado a uma internet WIFI e ter o GPS a funcionar corretamente no telemóvel em que será feito os testes.
3. Por fim, ter a aplicação instalada no telemóvel para a poder utilizar.

5.2 Descrição dos Casos de Teste

Em cada teste é(são) definido(s) o(s) requisito(s), procedimento e o respetivo critério de aceitação.

5.2.1 Menu Inicial

Descrição: Esta funcionalidade permite ao utilizador escolher o idioma pretendido e, assim, arrancar com a aplicação.

Passo	Procedimento	Critério de aceitação
1	Selecionar um dos idiomas disponíveis. Clicar na seta que se encontra à direita da combo-box de idiomas.	A aplicação tem como predefinição o idioma Português (pt.PT), sendo suficiente clicar no botão para a aplicação 'arrancar'. O idioma selecionado deverá ser o apresentado em toda a aplicação.
2	Clicar no botão "back" do telemóvel.	Deverá aparecer uma mensagem de aviso a perguntar se pretende sair da aplicação. Poderá clicar "cancelar" e apenas é fechada a janela de aviso ou "ok" e será fechada a aplicação.
3	Clicar no botão "i" no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.

5.2.1.1 Testes em situação de falha

Passo	Procedimento	Critério de aceitação
1	Clicar na seta que se encontra à direita da combo-box de idiomas com o acesso à Internet desligada no telemóvel.	Será visualizada uma mensagem de aviso a solicitar que seja ligada a internet. O utilizador será obrigado a fechar a aplicação, clicando no botão 'Fechar' não sendo possível a utilização da aplicação.

Tabela 5.1: Caso de Teste: Menu Inicial

Na figura 5.1 é possível observar os resultados obtidos neste Menu. Esta é a interface gráfica apresentada no arranque na aplicação. O tempo de arranque estimado desta interface gráfica é de aproximadamente 3 a 4 segundos, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

A funcionalidade oferecida por este Menu é a possibilidade de escolher o idioma que pretende utilizar ao longo da aplicação.

Figura 5.1: Menu apresentado no arranque da aplicação

5.2.2 Menu Principal

Descrição: Este menu permite ao utilizador escolher qual a funcionalidade que pretende utilizar, disponível na aplicação.

Passo	Procedimento	Critério de aceitação
1	Clicar na imagem de “Manual”.	Deverá abrir o menu “Manual”.
2	Clicar na imagem de “Criar Rota”.	Deverá abrir o menu “Criar Rota”.
3	Clicar na imagem de “Favoritos”.	Deverá abrir o menu “Favoritos”.
4	Clicar na imagem de “Mapa”.	Deverá abrir o menu “Mapa”.
5	Clicar na imagem de “Linha”.	Deverá abrir o menu “Linha”.
6	Clicar na imagem de “Sobre”.	Deverá abrir o menu “Sobre”.
7	Clicar no botão “back” já incorporado no próprio telemóvel.	Irá aparecer uma mensagem de aviso a perguntar se pretende sair da aplicação. Poderá clicar “cancelar” e apenas é fechada a janela de aviso ou “ok” e será fechada a aplicação.
8	Clicar no botão “i” no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.

5.2.2.1 Testes em situação de falha

Passo	Procedimento	Critério de aceitação
1	Clicar no botão “Criar Rota” com a Internet desligada.	Será visualizada uma mensagem de erro a informar que tem de ligar a Internet, não sendo possível aceder ao menu enquanto esta se mantiver desligada.
2	Clicar no botão “Mapa” com o GPS desligado.	Será visualizada uma mensagem de erro a informar que tem de ligar o GPS, sendo possível aceder diretamente às definições através do botão “Permitir”.

Tabela 5.2: Caso de Teste: Menu Principal

Na figura 5.2 é possível observar todos os Sub-Menus existentes. Esta é a interface gráfica apresentada é do tipo TabLayout, anteriormente referido. Cada imagem apresentada funciona como um botão que abrirá o Sub-Menu correspondente. O tempo de arranque estimado para esta interface gráfica é de aproximadamente 1 a 2 segundos, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

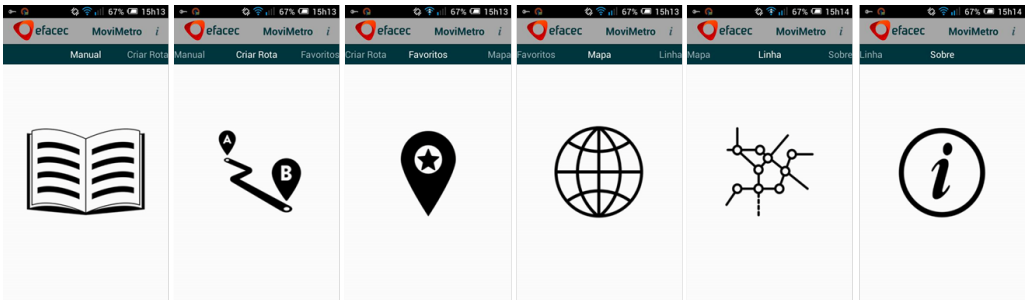


Figura 5.2: Menu apresentado para a escolha de Sub-Menus

5.2.3 Sub-Menu Manual

Descrição: Esta funcionalidade permite ao operador visualizar informações importantes para a utilização da aplicação.

Passo	Procedimento	Critério de aceitação
1	Abrirá uma janela em que poderá ser visualizada informação sobre a aplicação, explicando todas as suas funcionalidades existentes.	-
2	Clicar no botão “i” no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.
3	Clicar na seta que se encontra à esquerda do nome do menu.	Deverá retroceder para o menu principal.

Tabela 5.3: Caso de Teste: Sub-Menu Manual

Na figura 5.3 é possível observar toda a informação necessária para uma fácil utilização da aplicação. O tempo de arranque estimado desta interface gráfica é considerado imediato, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

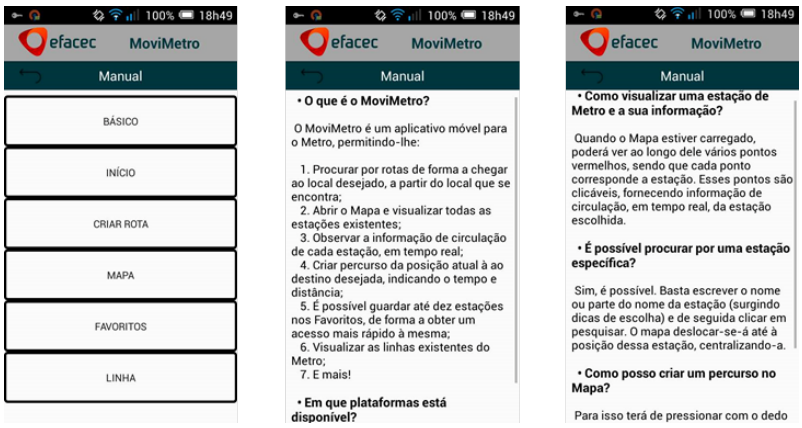


Figura 5.3: Sub-Menu com informação de utilização da aplicação

5.2.4 Sub-Menu Criar Rota

Descrição: Esta funcionalidade permite ao utilizador escolher a estação Origem e Destino, criando a rota mais apropriada que deverá ser percorrida.

Passo	Procedimento	Critério de aceitação
1	Clicar no botão “Criar Rota” do menu Principal.	Abrir o menu “Criar Rota”.
2	Escolher a estação Origem.	Poderá observar todas as estações de Metro existentes por ordem alfabética sendo que, após escolhida, ficará selecionada.
3	Escolher a estação Destino.	Poderá observar todas as estações de Metro existentes por ordem alfabética sendo que, após escolhida, ficará selecionada.
4	Selecionar o botão “Pesquisar”.	Por baixo do botão “Pesquisar” deverão ser visualizados os caminhos a percorrer, evidenciando as estações de transição (se necessário) até ao Destino escolhido.
5	Clicar em cada uma das estações de origem da viagem.	Será apresentada uma nova janela onde poderá ser observada, em tempo real, a informação de circulação dos veículos que poderão ser úteis para chegar ao Destino selecionado.
6	Clicar no botão “Atualizar” da janela de informação.	Deverá ser atualizada a informação de circulação apresentada.
7	Clicar no botão “Fechar” da janela de informação.	A aplicação deverá voltar ao Menu Criar Rota.
8	Clicar no botão “i” no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações sobre as funcionalidades existentes neste menu.
9	Clicar no botão “back” do telemóvel.	Voltará ao Menu Principal.
10	Clicar na seta à esquerda do nome do menu.	Deverá retroceder para o menu principal.

5.2.4.1 Testes em situação de falha

Passo	Procedimento	Critério de aceitação
1	Clicar no botão “Pesquisar” tendo sido selecionada a mesma estação nos campos Origem e Destino.	Deverá ser visualizada uma mensagem de erro a informar que as estações de Origem e Destino são iguais.

Tabela 5.4: Caso de Teste: Sub-Menu Criar Rota

Na figura 5.4 é possível observar os passos a seguir para a criação de uma Rota, escolhendo a estação Origem e a estação Destino. São apresentados os caminhos que se podem percorrer, com estações clicáveis para obter informação de circulação, em tempo real. O tempo de arranque estimado desta interface gráfica é considerado imediato, sendo uma estimativa subjetiva, pois depende da ligação à Internet. Quando clicado no botão "Pesquisar", o tempo de arranque estimado é de aproximadamente 1 segundo e a abertura de uma estação entre 1 a 2 segundos.

A funcionalidade disponível neste Sub-Menu dá a oportunidade de o utilizador criar a sua própria rota, desde a estação de metro em que se encontra até ao destino para onde pretende deslocar-se. Para além do percurso que o utilizador deverá percorrer, é também demonstrado, quando aberta uma estação de origem de um dos percursos, apenas os Metros que são possíveis utilizar para chegar ao destino.

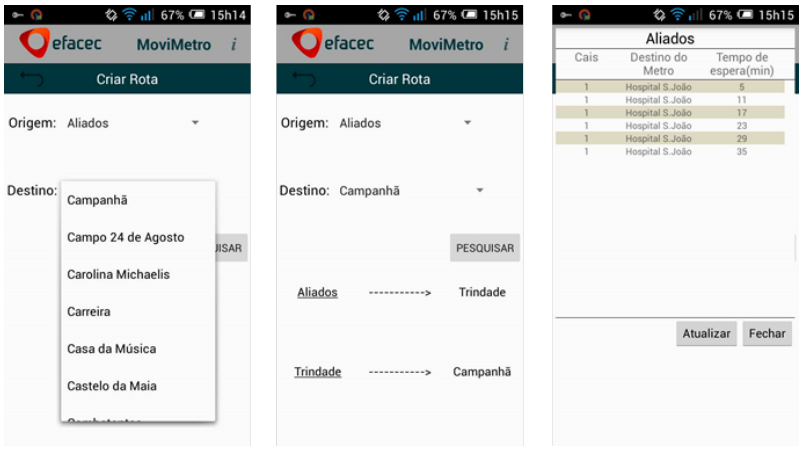


Figura 5.4: Sub-Menu apresentado para a criação de uma Rota

5.2.5 Sub-Menu Mapa

Descrição: Esta funcionalidade permite ao utilizador visualizar no mapa a sua localização, assim como visualizar todas as estações existentes, pesquisar por estações, calcular distâncias e tempos (de carro ou a pé) entre a sua posição atual e um local previamente escolhido.

Passo	Procedimento	Critério de aceitação
1	Clicar no botão "Mapa" do menu principal.	Abrir o menu "Mapa".
2	Clicar no botão de localização.	O mapa deverá mover-se de forma a centrar a posição atual do utilizador.
3	Clicar durante 1/2segundos num local e, de seguida, clicar no botão "carro" ou "a pé".	Após uns segundos deverá ser visualizado no mapa uma linha indicando o caminho a percorrer desde a localização atual até ao local pressionado, indicando também o a distância (em quilómetros) e a duração do percurso (em horas e minutos).
4	Escrever o nome de uma estação e clicar no botão "Pesquisar".	Enquanto escreve, deverão ser visualizadas sugestões de nomes de estações. Após o clique no botão, o mapa deverá mover-se de forma a centrar a posição na estação escolhida.
5	Clicar numa das estações apresentadas.	Será visualizada uma nova janela onde poderá observar, em tempo real, toda a informação de circulação para a estação escolhida.
6	Clicar no botão "Atualizar" da janela de informação.	A informação apresentada deverá ser atualizada.
7	Clicar no botão "Guardar" da janela de informação.	A estação deverá ser apresentada na área de "Favoritos" (teste 5.2.6). Deverá ser visualizada uma imagem informativa de que a estação foi guardada com sucesso. No caso de a área de "Favoritos" atingir o limite, deverá ser apresentada uma mensagem a informar que não é possível guardar mais nenhuma estação.
8	Clicar no botão "Fechar" da janela de informação.	No botão "Fechar" deverá voltar ao Menu Mapa.
9	Clicar no botão "i" no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.
10	Clicar no botão "back" do telemóvel.	Voltará ao Menu Principal.
11	Clicar na seta à esquerda do nome do menu.	Deverá retroceder para o menu principal.

5.2.5.1 Testes em situação de falha

Passo	Procedimento	Critério de aceitação
1	Clicar na estação com a internet desligada.	Deverá aparecer uma mensagem de erro em que, ao clicar em "Fechar", a aplicação deverá ser mantida no menu Mapa.

Tabela 5.5: Caso de Teste: Sub-Menu Mapa

Na figura 5.5 é possível observar o mapa, sendo o ponto a azul a localização GPS atual do utilizador e os markers a vermelho correspondentes às estações de metro. Pode abrir uma estação, sendo o tempo de arranque estimado de 1 a 2 segundos. Poderá também calcular a distância entre a sua localização e um ponto à escolha (correspondendo ao marker azul temporário), sendo que o tempo de arranque estimado é de 2 a 5 segundos. O tempo de arranque estimado desta interface gráfica é de aproximadamente 2 a 3 segundos, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

São apresentadas algumas funcionalidades tais como a possibilidade de disponibilizar ao utilizador um caminho possível a percorrer entre a sua localização atual até ao destino que pretenda ir. A informação recebida será a distância a que se encontra em quilómetros e o tempo que deverá demorar a percorrer, podendo escolher se se desloca a pé ou de carro. Para além disso, irá surgir uma linha vermelha sobre o mapa indicando o caminho que deverá seguir. Uma outra funcionalidade é o facto de poder procurar por uma estação de metro, centralizando no mapa essa mesma estação quando procurada pelo utilizador. As estações disponibilizadas podem ser abertas oferecendo ao utilizador informação de circulação, em tempo real. Por fim o utilizador poderá fazer *zoom in* e *zoom out* e escolher entre visão por satélite ou por via terrestre.

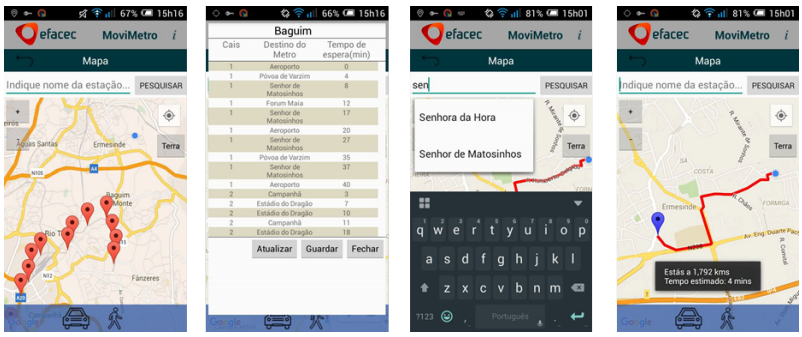


Figura 5.5: Sub-Menu apresentando o mapa e as funcionalidades disponíveis

5.2.6 Sub-Menu Favoritos

Descrição: Esta funcionalidade permite ao utilizador visualizar as estações previamente escolhidas como favoritas (máximo de 10), tendo acesso à informação em tempo real dessa estação.

Passo	Procedimento	Critério de aceitação
1	Clicar no botão "Favoritos" do menu principal.	Abrir o menu "Favoritos".
2	Clicar numa das estações apresentadas.	Será apresentada uma nova janela onde poderá ser observada, em tempo real, toda a informação de circulação da estação escolhida.
3	Clicar no botão "Atualizar" da janela de informação.	A informação apresentada deverá ser atualizada.
4	Clicar no botão "Apagar" da janela de informação.	Este botão permite apagar uma estação dos "Favoritos". Será apresentada uma janela de confirmação. Caso o utilizador selecione OK, a estação deverá desaparecer dos "Favoritos". O utilizador pode cancelar a operação, mantendo a estação nos "Favoritos".
5	Clicar no botão "Fechar" da janela de informação.	No botão "Fechar" deverá voltar ao Menu Favoritos.
6	Clicar no botão "i" no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.
7	Clicar no botão "back" do telemóvel.	Voltará ao Menu Principal.
8	Clicar na seta à esquerda do nome do menu.	Deverá retroceder para o menu principal.

5.2.6.1 Testes em situação de falha

Passo	Procedimento	Critério de aceitação
1	Clicar na estação com a internet desligada.	Deverá ser visualizada uma mensagem de erro em que, ao clicar em "Fechar", a aplicação volta ao Menu Principal.

Tabela 5.6: Caso de Teste: Sub-Menu Favoritos

Na figura 5.6 é possível observar todas as estações que foram guardadas pelo utilizador. Cada estação pode ser aberta, apresentando a informação de circulação, em tempo real. O tempo de arranque estimado para abrir uma estação é de 1 a 2 segundos. Relativamente ao tempo de arranque estimado desta interface gráfica é considerado imediato, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

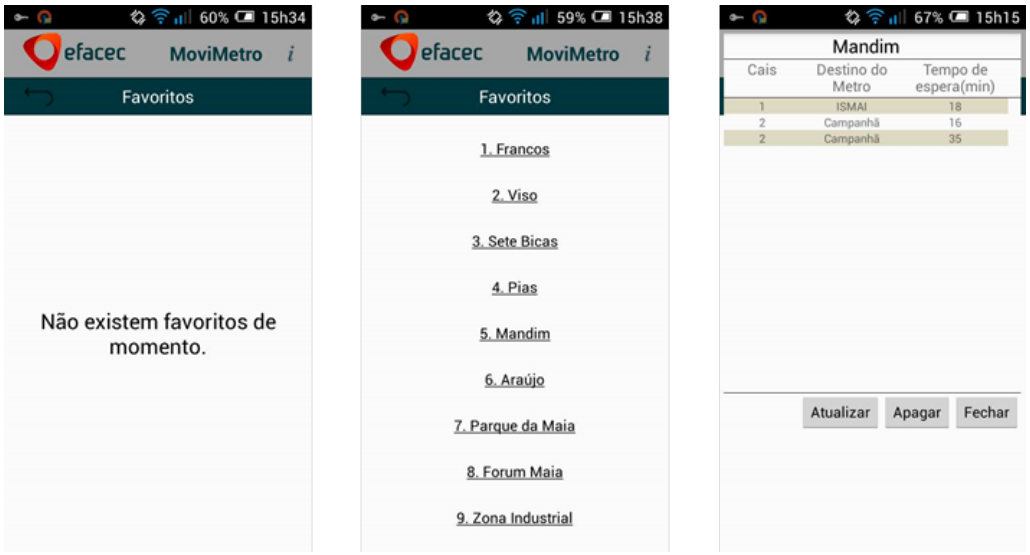


Figura 5.6: Sub-Menu apresentando todas as estações guardadas nos Favoritos

5.2.7 Sub-Menu Linha

Descrição: Esta funcionalidade permite ao operador visualizar as linhas que estão disponíveis no Metro do Porto.

Passo	Procedimento	Critério de aceitação
1	Clicar na Linha pretendida.	Abrirá uma nova janela podendo ser visualizada a linha completa escolhida, desde a estação de início até à estação de fim.
2	Clicar no botão "back" do telemóvel.	Voltará ao Menu Principal.
3	Clicar no botão "i" no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.
4	Clicar na seta à esquerda do nome do menu.	Deverá retroceder para o menu principal.

Tabela 5.7: Caso de Teste: Sub-Menu Linha

Na figura 5.7 é possível observar todas as linhas existentes no Metro do Porto. Cada linha corresponde a uma imagem que pode ser ampliada. O tempo de arranque estimado desta interface gráfica é considerado imediato, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

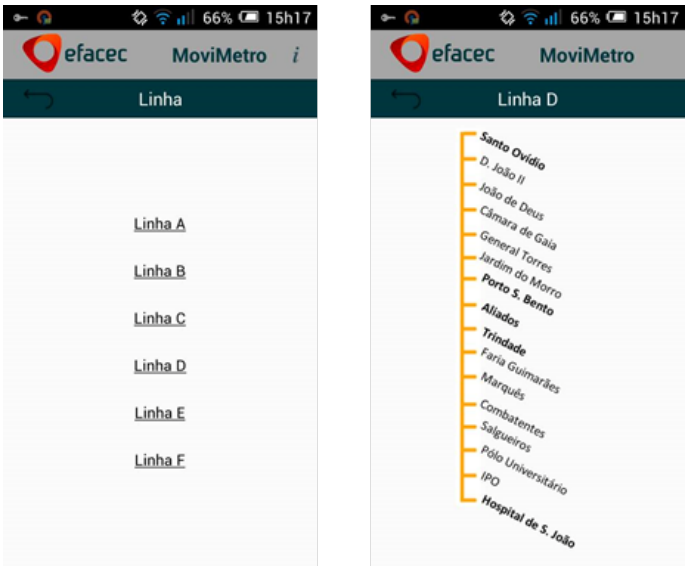


Figura 5.7: Sub-Menu apresentando as linhas de metro existentes

5.2.8 Sub-Menu Sobre

Descrição: Esta funcionalidade permite ao operador visualizar informações sobre a Empresa e implementações feitas.

Passo	Procedimento	Critério de aceitação
1	Abrirá uma janela em que poderá ser visualizada informação sobre a EFACEC, parcerias e o que motivou ao desenvolvimento deste projeto.	-
2	Clicar no botão "back" do telemóvel.	Voltará ao Menu Principal.
3	Clicar no botão "i" no canto superior direito.	Deverá ser visualizada uma mensagem de ajuda com informações das funcionalidades existentes neste menu.
4	Clicar na seta à esquerda do nome do menu.	Deverá retroceder para o menu principal.

Tabela 5.8: Caso de Teste: Sub-Menu Sobre

Na figura 5.8 é possível observar a informação sobre a empresa onde foi desenvolvido o projeto, quais as implementações utilizadas e o motivo do seu desenvolvimento. O tempo de arranque estimado desta interface gráfica é considerado imediato, sendo uma estimativa subjetiva, pois depende da ligação à Internet.

Figura 5.8: Sub-Menu apresentando informação sobre a empresa e implementações feitas

Capítulo 6

Conclusão e Trabalho Futuro

Neste capítulo são apresentadas as conclusões globais desta dissertação, tendo por base os objetivos que a motivaram. Tendo em consideração os resultados obtidos neste projeto, serão também apresentados possíveis trabalhos a desenvolver no futuro como complemento a este estudo.

6.1 Reflexão

Quando solicitados a fazer uma reflexão sobre o trabalho que ocupou um semestre letivo e que implicou conhecimentos e aprendizagens desenvolvidas ao longo dos quatro anos e meio de curso, muitos aspetos poderão ser objeto da particular atenção e merecedores de destaque especial.

Obviamente não podendo nem querendo ser exaustivo neste momento, pelo risco que esta reflexão comporta, uma vez que é convicção que um trabalho deste tipo não pode nunca ser fechado e também por omissões que certamente apareceriam nesta avaliação e que colocaria alguma situação de fragilidade, o que é de evitar.

Assim, como reflexão final, afirma-se apenas que este trabalho foi frutuoso, importante e motivador. Gerou dinâmicas e permitiu estabelecer lados profissionais e de amizade, que certamente irão acompanhar o percurso dos próximos anos.

Esta reflexão possibilitou essencialmente tomar consciência de:

- A importância por uma opção inicial na utilização do UML pela simplicidade que este encerra, pela fácil compreensão do método e pelo mesmo permitir uma acessível organização de ideias;

- A utilização do Web Server Apache como repositório para o Slim demonstrou ser uma opção bastante fiável pela segurança oferecida e simplicidade de configuração e programação. No entanto, este sistema demonstra alguma limitação visto não ser possível haver troca de partilha de informação, implicando, assim, que seja de baixo e médio uso ao nível do utilizador;

- Pelo constrangimento de tempo na realização deste trabalho, assumiu-se que esta aplicação seria apenas executada para dispositivos Android uma vez que este é o sistema operativo mais difundido no mercado;
- Foi colocada uma ênfase prévia e acentuada no desenvolvimento da interface, consciente e tendo como premissa, que esta teria de oferecer simplicidade e um aspeto visual atrativo para o utilizador.

6.2 Conclusão Final

A finalidade de implementação deste sistema é a apresentação de informação de circulação do metro do Porto, em tempo real, criação de favoritos e a possibilidade de calcular distância entre a posição atual do utilizador e a estação de metro mais próxima, tendo estas sido bem-sucedidas.

Com isto, pode-se dizer que os objetivos iniciais propostos foram todos cumpridos. Foram ainda tidos em consideração alguns objetivos complementares propostos pela orientadora da empresa, ao longo do desenvolvimento do projeto, resultando estes em ótimas funcionalidades para aplicação, referindo-se a este propósito a possibilidade de criar uma Rota.

A execução deste trabalho possibilitou ainda a aprendizagem e o aumento de conhecimentos e experiências no domínio dos dispositivos móveis, neste caso a linguagem Android, e no domínio dos Web Server, nomeadamente o Apache.

Regista-se algumas dificuldades sentidas ao longo do desenvolvimento deste projeto, nomeadamente na configuração e instalação do Apache na máquina utilizada e na extração de informação do Slim para o Android, uma vez que a informação recebida pelo Slim tem de ser previamente tratada antes de ser disponibilizada ao utilizador.

6.3 Trabalho Futuro

Como desenvolvimentos futuros propõe-se o seguinte:

- Utilização de uma outra ferramenta que permita que a aplicação seja utilizada em grande escala ao nível dos utilizadores. Uma solução proposta será a utilização de um webservice, visto permitir ao programador um maior controlo do código desenvolvido;
- No caso de se expandir esta aplicação para uso em redes de metros de outros países, uma utilidade necessária seria a extensão dos idiomas disponíveis;
- Devido à necessidade do desenvolvimento da aplicação nativamente, como trabalho futuro será importante criar uma versão da aplicação para IOS. Com isto permitiria abranger a grande maioria dos dispositivos móveis no mercado.

Referências

[1] Matthewadavid. Do you build an app or a mobile website?, Janeiro 2014. disponível em http://visualizetheweb.com/do-you-build-an-app-or-a-mobile-website/.

[2] Sarah Perez. Worldwide mobile app trends, Outubro 2011. disponível em https://techcrunch.com/2011/10/11/xyologic-releases-hundreds-of-reports-detailing-worldwide-mobile-app-trends/

[3] FEDERICO VITICCI. Apple announces 40 billion app store downloads, Janeiro 2013. disponível em https://www.macstories.net/tag/appstore/.

[4] Inc. IDC Research. Smartphone os market share, Agosto 2015. disponível em http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[5] W3Techs. Usage statistics and market share of microsoft-iis for websites, Janeiro 2016. disponível em https://w3techs.com/technologies/details/ws-microsoftiis/all/all.

[6] Netcraft Ltd. Web server survey, Abril 2014. disponível em http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html.

[7] SRINIVAS TAMADA. Create a restful services using slim php framework, Dezembro 2014. disponível em http://www.9lessons.info/2014/12/create-restful-services-using-slim-php.html.

[8] John Coggeshall. Improving performance through persistent connections, Fevereiro 2006. disponível em http://www.oracle.com/technetwork/articles/coggeshall-persist-084844.html.

[9] Jerry Hildenbrand. What is android, Maio 2015. disponível em http://www.androidcentral.com/what-android.

[10] Google Inc. Introduction to android. disponível em http://developer.android.com/guide/index.html.

[11] Apple Inc. What is ios. disponível em http://www.apple.com/ios/what-is/.

[12] Apple Inc. Apple developer program. disponível em https://developer.apple.com/programs/how-it-works/.

[13] Richard Hay. The exodus from the windows phone ecosystem, Abril 2015. disponível em https://http://winsupersite.com/windows-phone/exodus-windows-phone-ecosystem.

[14] James B. Lingan. Web server, Julho 2015. disponível em http://whatis.techtarget.com/definition/Web-server.

[15] Bob Lehto Keith Dodge e Mike Weiner. Internet information server. disponível em http://searchwindowsserver.techtarget.com/definition/IIS.

[16] Apache Software Foundation. Apache http server project. disponível em http://httpd.apache.org/ABOUT_APACHE.html.

[17] Slim Framework Team. Slim. disponível em http://www.slimframework.com/.

[18] Roy T Fielding e Richard N Taylor. Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[19] David Linthicum Moriah Sargent e Duraisamy Sivaram. Soap (simple object access protocol). disponível em http://searchsoa.techtarget.com/definition/SOAP.

[20] LoveToKnow Corp. Xml database - computer definition. disponível em http://www.yourdictionary.com/xml-database.

[21] Damon Feldman. Moving from relational modeling to xml and marklogic data models, Abril 2013. disponível em http://www.marklogic.com/resources/slides-moving-from-relational-modeling-to-xml-and-marklogic-data-models/resource_download/presentations/.

[22] Tutorialspoint. Xml - databases. disponível em http://www.tutorialspoint.com/xml/xml_databases.htm.

[23] Kimbro Staken. Introduction to native xml databases, Outubro 2001. disponível em http://www.xml.com/pub/a/2001/10/31/nativexmldb.html.

[24] Margaret Rouse. Relational database. disponível em http://searchsqlserver.techtarget.com/definition/relational-database.

[25] Oracle Technology Network. A relational database overview. disponível em https://docs.oracle.com/javase/tutorial/jdbc/overview/database.html.

[26] Techopedia Inc. Relational database. disponível em https://www.techopedia.com/definition/1234/relational-database-rdb.

[27] Android Delevoper. Saving key-value sets. disponível em https://developer.android.com/training/basics/data-storage/shared-preferences.html.

[28] Android Delevoper. Fragmentstatepageradapter. disponível em https://developer.android.com/reference/android/support/v4/app/FragmentStatePagerAdapter.html.

[29] Android Delevoper. Imagebutton. disponível em https://developer.android.com/reference/android/widget/ImageButton.html.

[30] Android Delevoper. Imageview. disponível em https://developer.android.com/reference/android/widget/ImageView.html.

[31] Android Delevoper. Dialogfragment. disponível em https://developer.android.com/reference/android/app/DialogFragment.html.

[32] Android Delevoper. Tablelayout. disponível em https://developer.android.com/reference/android/widget/TableLayout.html.

[33] Android Delevoper. Tablerow. disponível em https://developer.android.com/reference/android/widget/TableRow.html.

[34] Android Delevoper. Autocompletetextview. disponível em https://developer.android.com/reference/android/widget/AutoCompleteTextView.html.

[35] Android Delevoper. Marker. disponível em https://developers.google.com/android/reference/com/google/android/gms/maps/model/Marker.

[36] Android Delevoper. Location. disponível em https://developer.android.com/reference/android/location/Location.html.

[37] Android Delevoper. Help your users find their way. disponível em https://developers.google.com/maps/documentation/directions/?csw=1.

[38] Android Delevoper. Jsonarray. disponível em https://developer.android.com/reference/org/json/JSONArray.html.

[39] Android Delevoper. Encoded polyline algorithm format. disponível em https://developers.google.com/maps/documentation/utilities/polylinealgorithm?csw=1.

[40] Android Delevoper. Connectivitymanager. disponível em https://developer.android.com/reference/android/net/ConnectivityManager.html.

[41] Android Delevoper. Url. disponível em https://developer.android.com/reference/java/net/URL.html.

[42] Android Delevoper. Bufferedreader. disponível em https://developer.android.com/reference/java/io/BufferedReader.html.

[43] Android Delevoper. Alertdialog. disponível em https://developer.android.com/reference/android/app/AlertDialog.html.