Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2014

**Tiago Daniel Serra Queirós Pinto**

# Desenvolvimento de Aplicações Móveis em iOS

# Mobile Apps Development with iOS

**Tiago Daniel Serra Queirós Pinto**

# Desenvolvimento de Aplicações Móveis em iOS
# Mobile Apps Development with iOS

Relatório de Estágio apresentado à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Sistemas da Informação, realizado sob a orientação científica do Doutor José Maria Amaral Fernandes, Professor Assistente do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Herlander Jorge Madaleno dos Santos, Engenheiro de Software, Gestor Técnico e Líder de Equipa na Portugal Telecom Inovação e Sistemas.

Dedico este trabalho à minha namorada Sara pelo tremendo apoio e dedicação.

**o júri**

| | |
|---|---|
| **presidente** | Professor Doutor António Joaquim da Silva Teixeira<br>Professor Associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro |
| **vogal - arguente principal** | Professor Doutor António Miguel Pontes Pimenta Monteiro<br>Professor Auxiliar do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto |
| **vogal - orientador** | Professor Doutor José Maria Amaral Fernandes<br>Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro |

**agradecimentos**

Quero agradecer à minha namorada Sara por toda a ajuda e apoio incansável ao longo da fase final do meu percurso académico e por estar incondicionalmente ao meu lado.

Quero agradecer também de forma especial à minha mãe, que apesar de todas as dificuldades, conseguiu educar-me e contribuir em grande parte para a pessoa que sou hoje.

Agradeço aos meus irmãos fantásticos, Pedro e João por serem quem são.

Quero deixar um agradecimento à minha avó Agostinha pela infinita disponibilidade e preocupação.

Agradeço também ao meu Pai.

Quero também agradecer à PT Inovação e Sistemas pela oportunidade de integração num ambiente de trabalho empresarial, bem como aos meus colegas estagiários por toda a entreajuda e apoio.

Agradeço também ao Professor José Maria Amaral Fernandes pela orientação e disponibilidade durante todo o trabalho.

**palavras-chave**

Dispositivos móveis; Aplicações Móveis; iOS; *Objective-C*; *Swift*; Aplicações multiplataforma; *Google Analytics*; Testes automatizados; *iDoor*; PT.

**resumo**

Atualmente, os dispositivos móveis possuem um papel importante no quotidiano da sociedade em geral, pelo que existe uma incessante busca pelo aumento das funcionalidades e potencialidades destes dispositivos.

A *Apple*, com a criação do *iPhone*, teve um grande impacto no Mercado dos dispositivos móveis especialmente no dos "*Smartphones*". Com o *iPhone*, a *Apple* introduziu um novo ecossistema de desenvolvimento de aplicações móveis no sistema operativo iOS e no IDE *XCode* para suportar desenvolvimento de aplicações usando as linguagens *Objective-C* e mais recentemente em *Swift*.

O presente relatório descreve o estágio realizado no seio da PT Inovação e Sistemas focado no desenvolvimento de aplicações móveis especialmente em iOS em projetos em curso, nomeadamente no MEO Go e outros projetos centrados na distribuição de conteúdos multimédia, e na utilização de soluções multiplataforma *PhoneGap* e *AppAccelerator*. Estas soluções permitem o desenvolvimento de aplicações que podem ser executadas em iOS e em *Android* e que já suportam a manipulação de *streams* multimédia.

Durante este estágio, foi explorada utilização de *Google Analytics* e testes automatizados nos quais foram utilizados *UIAutomation* e *AppThwack* no processo de desenvolvimento e avaliação de aplicações móveis em iOS. Também foram implementadas duas aplicações para explorar a nova linguagem *Swift* e a utilização do *iBeacon* na aplicação *iDoor* concebida para permitir a abertura automática de portas através da tecnologia *Bluetooth* 4.0 LE usando um sistema de autenticação remoto.

**keywords**

Mobile devices; Mobile Apps; iOS; Objective-C; Swift; Multiplatform Applications; Google Analytics; Automated tests; iDoor; PT.

**abstract**

Nowadays, mobile devices play an important role in the everyday life of society in general, so there is a ceaseless quest for increased functionalities and capabilities of these devices.

Apple, with the creation of the iPhone, had a great impact on the mobile devices market especially in the "Smartphones". With the iPhone, Apple introduced a new mobile application development ecosystem in the iOS operating system and the XCode IDE to support development of applications using languages like Objective-C and, more recently, Swift.

This report describes the internship held within the PT Inovação e Sistemas which was focused on the development of mobile applications, especially in iOS in ongoing projects, particularly in MEO Go and other projects focused on the distribution of multimedia content and the use of cross-platform solutions like PhoneGap and AppAccelerator. These solutions enable the development of applications that can run on iOS and Android and support the handling of multimedia streams.

During this internship automated tests were explored using Google Analytics in which were used UIAutomation and AppThwack in the development process and evaluation of mobile applications for iOS. There were also implemented two applications to explore the new Swift language and the use of iBeacon in iDoor application designed to allow the automatic opening doors through Bluetooth 4.0 LE technology using a remote authentication system.

## Abreviaturas e Siglas

APIs - Interfaces de Programação de Aplicações
ATM - *Asynchronous Transfer Mode*
CET – Centro de Estudos de Telecomunicações
CPU – *Central Processing Unit*
CSS - *Cascading Style Sheets*
CTT - Correios e Telecomunicações de Portugal
DETI – Departamento de Eletrónica, Telecomunicações e Informática
DRM - *Digital Rights Management*
GB – Gigabyte
GECA – Grupo de Estudos de Comutação Automática
GPS - *Global Positioning System*
GSM – Sistema Global para Comunicações Móveis
HLS - *HTTP Live Streaming*
HTML - *HyperText Markup Language*
ID – *Identification*
IDE - Ambiente Integrado de Desenvolvimento de *Software*
IIS - *Internet Information Server*
Inc. - *Incorporated*
JSON - *JavaScript* Object Notation
Kb – Quilobyte
LE – *Low Energy*
LLVM – *Low Level Virtual Machine*
Mac – *Macintosh*
MPEG - *Moving Picture Experts Group*
MTV – Multimédia e Televisão Interativa e Convergente
PDA – *Personal Digital Assistant*
PT – Portugal Telecom
PTIS – Portugal Telecom Inovação e Sistemas
RAM – *Random-access Memory*
RTSP - *Real Time Streaming Protocol*

## Abreviaturas e Siglas

S.A. – Sociedade Anónima

SAPO - Serviço de Apontadores Portugueses

SDK – *Software Development Kit*

SGPS - Sociedade Gestora de Participações Sociais

SMS – *Short Message Service*

S.O. – Sistema Operativo

TMN - Telecomunicações Móveis Nacionais

UD – Unidade de Desenvolvimento

UI – *User Interface*

UUID - *Universally Unique Identifier*

Wi-fi – *Wirelles Fidelity*

XML - *eXtensible Markup Language*

## ÍNDICE

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABELAS

# 1. INTRODUÇÃO

A tecnologia faz, cada vez mais, parte do quotidiano do ser humano. Atualmente, a forma como a humanidade comunica estende-se à escala global graças ao desenvolvimento galopante da tecnologia de dispositivos móveis, bem como das tecnologias de transmissão de dados.

A convergência entre a comunicação e a informática originou uma notória evolução dos dispositivos móveis que se tornaram, não só um meio de comunicação, mas também uma via de acesso a uma incomensurável gama de funcionalidades e possibilidades. Desde o lançamento do *iPhone* pela *Apple* e do sistema operativo *Android* pela *Google*, as potencialidades dos *smartphones* aumentaram drasticamente, pelo que se tornaram as ferramentas de comunicação, produtividade e entretenimento de eleição da atualidade (Charles, 2012).

Apesar de recente, o mercado de tecnologias e aplicações para dispositivos móveis encontra-se bastante desenvolvido, pelo que foi gerada uma indústria bastante competitiva neste ramo que, sendo segmentado, não aposta unicamente em marcas de dispositivos móveis, como também em sistemas operativos, nomeadamente iOS, *Android*, *Windows*, *BlackBerry* e outros (Rivera & Meulen, 2014). Esta diversidade de sistemas operativos e, consequentemente, dos ambientes de desenvolvimento de aplicações para os mesmos obriga a que as empresas invistam na especialização das suas equipas de programadores de aplicações *mobile*.

Os estágios para alunos universitários em contexto empresarial são, então, vantajosos no sentido em que proporcionam ao estudante uma oportunidade de desenvolvimento de competências e aplicação e aquisição de conhecimentos teórico-práticos. Além disto, permitem que as empresas contactem com novas ideias e metodologias de trabalho introduzidas por jovens programadores especializados em diversas áreas, tal como programação em iOS.

## 1.2. CONTEXTUALIZAÇÃO

Este relatório foi elaborado no âmbito do estágio realizado na Portugal Telecom Inovação e Sistemas (PTIS) de forma a atingir os objetivos necessários para terminar o Mestrado em Sistemas de Informação do Departamento de Eletrónica, Telecomunicações e Informática (DETI), da Universidade de Aveiro.

O referido estágio teve a duração de um ano, entre Outubro de 2013 e Outubro de 2014, e foi orientado por José Maria Amaral Fernandes, Professor Auxiliar no DETI, e supervisionado por Herlander Jorge Madaleno dos Santos, Engenheiro de Software, Gestor Técnico e Líder de Equipa na PTIS.

Os diversos projetos, ou unidades de desenvolvimento (UD), desenvolvidos no período supracitado foram fomentados na área de Multimédia e Televisão Interativa e Convergente (MTV), um dos departamentos da PTIS, e tiveram como alvo o desenvolvimento de soluções de *software* para o sistema operativo de dispositivos móveis da *Apple*, iOS.

## 1.3. PERFIL DO DOCUMENTO

Os objetivos deste relatório prendem-se com diversos aspetos ligados ao local de estágio, à programação de aplicações móveis em iOS para dispositivos *Apple* e, consequentemente, com as unidades de desenvolvimento realizadas neste âmbito. Assim, este documento encontra-se dividido em três partes principais, nomeadamente: o Grupo Portugal Telecom (PT), o Estado da Arte e os Projetos Desenvolvidos.

Em primeiro lugar, é apresentado o Grupo PT e é descrita de forma breve a história do grupo e de duas das suas marcas de referência que impulsionaram o mercado das aplicações móveis em Portugal, o SAPO e o MEO. Além disto, é explorada a história da Portugal Telecom Inovação e Sistemas, de forma a apresentar a empresa na qual o estágio foi concretizado.

Na segunda parte deste trabalho é construído o Estado da Arte direcionado para a temática do desenvolvimento *mobile*, mais especificamente as aplicações criadas para o sistema operativo utilizado pela *Apple* nos seus dispositivos móveis, o iOS. Assim, primariamente são definidos e caracterizados os dispositivos e aplicações móveis, acerca das quais é distinguida a diferença entre aplicações nativas, *web* e híbridas.

Seguidamente, é examinada a evolução das tecnologias produzidas pela *Apple* ao longo da sua história, focando o *iPhone* e o marco que impôs na evolução tecnológica aquando da sua criação. De igual forma, é apresentado o iOS, bem como as características do ambiente de desenvolvimento do mesmo, o *XCode*. Para finalizar este capítulo serão analisadas as linguagens de programação utilizadas pela *Apple*: o *Objective-C*, a linguagem de eleição da empresa para a criação de aplicações que utiliza o compilador *Low Level Virtual Machine* (LLVM); e o *Swift*, uma nova linguagem que utiliza o mesmo compilador e que foi lançada com o objetivo de substituir, a longo prazo, o *Objective-C*.

Para finalizar, na terceira parte deste relatório serão descritos, explicados e explorados cinco projetos desenvolvidos ao longo do estágio, particularmente: as aplicações multiplataformas móveis, que existem como uma possibilidade de desenvolver um único projeto e exportá-lo para vários sistemas operativos móveis; a angariação de dados estatísticos acerca da utilização de uma aplicação através do *Google Analytics*, aplicado ao MEO GO; as plataformas de testes automatizados de *software*, como o UI *Automation* e o *AppThwack,* e a formulação de *scripts* específicos de análise e depuração de erros de *software* já existente; a demonstração das funcionalidades da nova linguagem de programação lançada pela *Apple,* o *Swift*, e a aplicação-exemplo criada; e, por fim, o *iDoor*, uma aplicação que permite a abertura simples de portas automáticas recorrendo a tecnologias *Bluetooth* 4.0 LE, *iBeacons* e a um sistema de autenticação remoto.

# 2. GRUPO PORTUGAL TELECOM

A Portugal Telecom, Sociedade Gestora de Participações Sociais (SGPS), Sociedade Anónima, S.A., mais conhecida por Grupo PT, é uma organização sediada em Portugal e direcionada para a oferta de serviços na área das telecomunicações e multimédia, quer a nível residencial, quer empresarial. A atividade da empresa prende-se com as comunicações fixas, móveis, de multimédia e de dados, bem como a criação de soluções empresariais (Rego, 2012).

O Grupo PT possui, atualmente, mais de cem milhões de clientes, e está presente em países como Portugal, Brasil, Cabo Verde, Moçambique, Timor, Angola, Quénia, China, São Tomé e Príncipe e Namíbia (Portugal Telecom, 2014a). A Portugal Telecom SGPS, S.A. possui, atualmente, marcas bastante competitivas no mercado das telecomunicações e reconhecidas internacionalmente tais como a MEO e a SAPO.

## 2.1. HISTÓRIA DO GRUPO

A Portugal Telecom (PT) foi fundada em 1992 após uma segmentação dos CTT (Correios e Telecomunicações de Portugal) que resultou na separação da secção de telecomunicações, pelo que se tornou uma empresa independente. No entanto, a PT possui origens mais remotas já que no final do século XIX deu-se início à sua constante progressão na implementação das tecnologias da comunicação em Portugal sendo, desde então a maior força impulsionadora da inovação das telecomunicações a nível nacional (Portugal Telecom, 2014b).

Apesar de ter sido uma empresa pública durante vários anos, em 2011 a PT sofreu um processo de privatização após cinco ofertas públicas de ações ao longo de dezasseis anos, pelo que o Estado Português prescindiu das suas 500 *Golden Share*, modificando a gestão da empresa (Diário de Notícias, 2011). Desde a sua criação, a Portugal Telecom manteve-se na vanguarda da tecnologia que nacional quer internacionalmente, o que se verificou com o lançamento do primeiro cartão pré-pago a nível mundial em 1995: o MIMO. Foi este serviço que contribuiu em grande parte para a adoção universal do telemóvel graças ao surgimento dos conceitos “pronto a falar” e “controlo de custos” (Portugal Telecom, 2005).

## 2.2. ESTRUTURA DA ORGANIZAÇÃO

A Portugal Telecom encontra-se subdividida em diversas empresas com áreas de atuação e abrangência distintas. O grupo é constituído pelas seguintes empresas, cujo organigrama se encontra na Figura 1:

- PT Comunicações, que se foca sobretudo nas áreas das comunicações, do fornecimento de Internet e da manutenção de portais *web*;
- Oi, operadora de telecomunicações integradas (fixo, móvel, dados, e multimédia), dirigida ao mercado brasileiro;
- MEO, Serviços de Comunicações e Multimédia, S.A., que se foca nas comunicações móveis, e provém do *rebranding* da antiga marca de Telecomunicações Móveis Nacionais (TMN);
- PT II, focada em investimentos internacionais;
- PT PRO, com responsabilidades de assessoria, administração e gestão empresarial;
- PT *Pay*, que presta serviços de pagamentos;
- PT *Contact*, focada em *telemarketing*;
- PT Inovação e Sistemas, responsável pela investigação e desenvolvimento de soluções tecnológicas e sistemas da Informação (Portugal Telecom, 2014c).

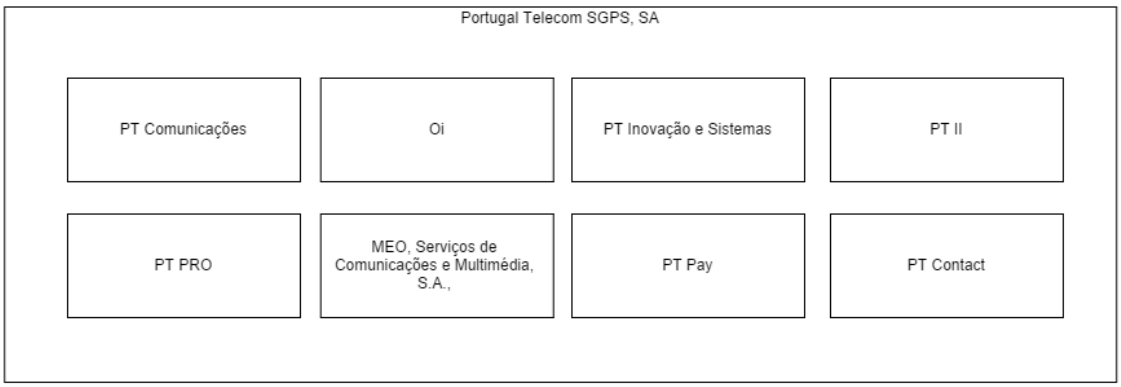


*Figura 1 - Organigrama da Portugal Telecom SGPS, S.A.*

## 2.3. SAPO

O SAPO[1] é uma das marcas de referência do Grupo PT em Portugal e dedica-se à prestação de serviços e fornecimento de produtos na Internet. Esta empresa é responsável pelo portal mais visitado em Portugal que fornece um motor de busca de referência a nível nacional e internacional já que é utilizado, atualmente, em Angola, Cabo Verde, Moçambique e Timor-Leste (Netscope, 2013)(SAPO, 2014a).

O SAPO foi criado a 4 de Setembro de 1995 pelo Centro de Informática da Universidade de Aveiro e o nome da marca teve origem numa adaptação da expressão "Serviço de Apontadores Portugueses". Em 1997 foi adquirido pela empresa Navegante, criada por seis membros do Centro de Informática da Universidade de Aveiro que, um ano depois, o vendeu à Saber & Lazer – Informática e Comunicação S.A. A partir desta data, a marca SAPO expandiu-se, lançando um serviço de correio eletrónico gratuito, o SAPO *Mail*, e posteriormente, em parceria com a Telepac, tornou-se também num fornecedor de Internet comercial e doméstico (SAPO, 2014b).

No ano de 1999, o SAPO foi comprado pela PT Multimédia – Serviços de Telecomunicações e Multimédia SGPS, S.A. e, em 2007, com a mudança da organização estrutural da empresa, passou a fazer parte da PT Comunicações, estrutura onde ainda se encontra atualmente (Dias & Leandro, 1999).

O portal SAPO possui uma vasta gama de plataformas, aplicações e serviços que englobam a generalidade de temáticas e assuntos de interesse, com vista a adquirir um maior número e diversidade de utilizadores. Existem inúmeras aplicações de dispositivos móveis desenvolvidas sob a alçada do SAPO que são frequentemente utilizadas como o MEO *Drive*, o MEO *Music*, entre outras (SAPO, 2014c).

[1] SAPO: http://www.sapo.pt/

## 2.4. MEO

O MEO – Serviços de Telecomunicações e Multimédia de Telecomunicações Fixas e Móveis, é uma marca comercial pertencente ao Grupo PT. Este serviço foi criado em 2008 com o intuito de vender e fornecer produtos *triple-play*, que incluem Televisão, Telefone e Internet (Jornal de Notícias, 2011). O MEO possui, atualmente, mais de um milhão de subscritores e, neste ano, foi anunciada a sua fusão com a marca de comunicações móveis TMN, mantendo-se o nome do primeiro (Público, 2014).

A aposta da empresa recai, atualmente, no M4O, o produto de eleição que o MEO que se prende com serviços de *quadruple-play*, englobando serviços de multimédia, dados, comunicações fixas e comunicações móveis. O M4O é vendido em diferentes possibilidades de velocidade de Internet ou pacotes de canais predefinidos, conforme os objetivos que o consumidor pretende atingir com este serviço (Simões, 2013).

Neste momento, o MEO concentra esforços para a implementação de uma rede nacional de fibra ótica, já reconhecida como o "melhor serviço de *quadruple-play* em Portugal" atribuído pelo Centro de Avaliação de Satisfação do Consumidor (MEO, 2014a).

### 2.4.1. Aplicações MEO

A marca MEO investe continuamente no acompanhamento da rápida evolução tecnológica da atualidade, pelo que promove um desenvolvimento eficiente de aplicações para dispositivos móveis por parte dos seus programadores. Existem várias aplicações desenvolvidas por empresas do Grupo PT disponíveis nos mercados dos diferentes sistemas operativos móveis, nomeadamente o iOS, o *Android* e o *Windows Phone.*

Uma das principais aplicações oferecidas é o MEO Wi-fi que permite detetar redes sem fios disponíveis num dado local e alertar o utilizador sempre que este se encontra perto desta rede. Desta forma, todos os clientes da PT que possuam as credenciais de autenticação necessárias têm garantido o acesso gratuito à Internet sem fios. Esta aplicação permite, igualmente, gerir a ligação à Internet num *hotspot* MEO Wi-fi. O MEO *Cloud*, por sua vez, é a aplicação que possibilita o alojamento remoto de todos os tipos de ficheiros e o seu acesso em qualquer local ou dispositivo, facultando 16GB de espaço de armazenamento para o utilizador guardar os seus dados (MEO, 2014b).

O MEO *Stories* é outra aplicação desenvolvida para a criação de um vídeo a partir de fotografias ou outros vídeos que existam no telemóvel do utilizador, de forma a construir uma história e partilhá-la, tanto na televisão como na Internet. Já o MEO *Music*

permite o acesso a inúmeras faixas de música a partir de qualquer local ou dispositivo, permitindo uma sincronização personalizada por utilizador, pelas suas preferências musicais e pelas suas listas de reprodução (MEO, 2014b).

Outra aplicação móvel desenvolvida pelo MEO, gratuita e disponibilizada de forma vitalícia, é o MEO *Drive*, que, orientada com apoio de sinal GPS, permite que o utilizador realize uma navegação orientada, facultando o acesso a milhares de pontos de interesse (MEO, 2014b).

Por fim, o MEO GO[2] é a aplicação disponível para os três principais sistemas operativos móveis, bem como para *Windows 8*, pelo que abrange um grande número de plataformas e, consequentemente, de utilizadores. O MEO GO permite a visualização de canais de televisão em qualquer lugar utilizando um dispositivo compatível, bem como a consulta da programação de cada canal disponibilizado. Além disto, é possível realizar a gravação automática de programas pretendidos pelo utilizador e recuar até programas ou instantes específicos na reprodução de qualquer canal graças à existência de uma *timeline*. Esta aplicação possui, ainda, um Videoclube no qual é possível alugar, por um período pré-determinado, qualquer filme existente nos catálogos disponibilizados, por um preço variável e especificado individualmente (MEO, 2014b).

[2] MEO GO: http://meogo.meo.pt/

## 2.5. PT INOVAÇÃO E SISTEMAS

A PT Inovação e Sistemas é uma empresa do Grupo Portugal Telecom direcionada para o avanço tecnológico, cujo objetivo se prende com o desenvolvimento e entrega de produtos e serviços que se caracterizem pela inovação no mercado das telecomunicações e das tecnologias da informação. A PTIS está sediada em Aveiro e conta com uma rede de subsidiárias que operam em vários países do mundo, nomeadamente o Brasil, Marrocos, Angola, Moçambique, Namíbia, África do Sul e Espanha. Atualmente, mais de 130 milhões de pessoas comunicam através de tecnologia desenvolvida ou implementada nesta empresa (PT Inovação e Sistemas, 2014).

A PT Inovação e Sistemas foi fundada em 1950 com o nome de GECA – Grupo de Estudos de Comutação Automática. Em 1972 o seu nome e estrutura sofreram alterações, pelo que foi criado o Centro de Estudos de Telecomunicações – CET. A fixação da empresa na cidade de Aveiro impulsionou a aposta na formação da área da tecnologia por parte da Universidade de Aveiro e a criação de uma parceria entre as duas entidades, o que resultou em vários protocolos e projetos conjuntos que ainda se encontram ativos e são periodicamente reiniciados (Portugal Telecom, 2009) (Universidade de Aveiro, 2013).

A PT Inovação e Sistemas orgulha-se de, ao longo da sua história, ter conquistado várias metas notáveis como o MIMO - o primeiro serviço móvel pré-pago no mundo, a primeira ligação de banda larga ATM (*Asynchronous Transfer Mode*) e o MEO- – Serviço de Televisão Interativa, entre outras. A empresa possui, de momento, um número de colaboradores próximo dos dois milhares, apostando em técnicos qualificados e com elevados graus de habilitações académicas. Nesse sentido, foram criados programas de recrutamento de novos licenciados nas áreas técnicas de tecnologia, mais concretamente o programa Academia PT, que consiste em estágios orientados com o objetivo de captar ativos.

A PT Inovação e Sistemas tem na sua estrutura várias subdivisões ou departamentos, sendo um deles o departamento de MTV, Multimédia e Televisão Interativa e Convergente, responsável por desenvolver e suportar a aplicação MEO GO, bem como desenvolver recursos e otimizar toda a plataforma de *software* das *boxes* que permitem a interação dos utilizadores/clientes com a televisão.

# 3. ESTADO DA ARTE

A segunda parte deste relatório prende-se com a exposição e análise das informações atualmente existentes acerca das aplicações e dispositivos móveis, bem como na programação em iOS para a criação de aplicações *mobile* para dispositivos *Apple*. Desta forma, pretende-se conhecer o estado da arte acerca destes tópicos de forma a alcançar um conhecimento mais aprofundado acerca das temáticas que envolvem os projetos realizados em estágio.

## 3.1. DISPOSITIVOS E APLICAÇÕES MÓVEIS

Para o Ser Humano a comunicação é uma necessidade inata pelo que, durante toda a sua existência, o Homem criou formas cada vez mais eficientes de comunicar entre si. Assim, com o evoluir dos tempos foram criadas tecnologias direcionadas para a comunicação para permitir o encurtamento de distâncias entre as pessoas. Foi em 1973 que surgiu o primeiro telemóvel e, desde então, a aposta na sua evolução não cessou. Foi na década de 80 do século XX que surgiram os primeiros telemóveis equipados com GSM, ou Sistema Global para Comunicações Móveis, que permitiu a existência do serviço de SMS, *Short Message Service*, o *roaming* e o *voice mail* (Moedas, 2011). No entanto, foi apenas dez anos mais tarde que se deu início à globalização dos telemóveis como meio de comunicação já que passaram a fazer parte do quotidiano das massas populacionais em todo o mundo (Marques, 2013).

Ao longo da evolução tecnológica das comunicações móveis surgiu o conceito de PDA, o *Personal Digital Assistant* (Figura 2), que consistia num pequeno computador que permitia ao utilizador realizar tarefas que um simples telemóvel não conseguia executar (Hewlett-Packard Development Company L.P., 2012). A Figura 2 ilustra os diversos tipos de PDA's que existiram no mercado das tecnologias da informação e, pela sua análise, é possível compreender que foram o primeiro passo para a criação dos *smartphones* no sentido em que possuíam um aspeto semelhante aos dispositivos móveis atuais, como o ecrã táctil, e permitiam aceder a diversas funcionalidades que um telemóvel de então não possuía.

*Figura 2 - Tipos de PDA's (Fonte: The Gadgeteer, 2014)*

O progresso tecnológico dos dispositivos de comunicação impôs a criação de uma tecnologia que reunisse todas as vantagens inerentes ao telemóvel, ao PDA, à Internet (até então apenas utilizada em computadores), às câmaras fotográficas e de filmar e às aparelhagens, suprimindo as limitações de cada uma e reunindo tudo num único dispositivo móvel: o *smartphone* (Marques, 2013). Os *smartphones* revolucionaram a comunicação do século XXI durante qual o número de subscrições realizadas a serviços e dispositivos móveis cresceu exponencialmente encontrando-se, atualmente, em 5,4 biliões, uma parte considerável da massa populacional mundial de 7 biliões de pessoas (United Nations, 2013).

Um dispositivo móvel caracteriza-se pela sua capacidade de ser utilizado de uma forma não fixa e/ou em movimento, pelo que foi largamente proliferado pela população mundial pois, além de ter a capacidade de acompanhar a pessoa no seu dia-a-dia, permite o acesso à informação em tempo real. Mas existem outras vantagens nestes dispositivos, nomeadamente: uma pequena infraestrutura física, que permite que a pessoa esteja sempre acompanhada pelo dispositivo já que este é fácil de transportar; a possibilidade de utilização em áreas remotas, permitindo o acesso a canais de comunicação que não existiam anteriormente; e o facto de apenas exigir uma literacia básica, pelo que diminui as barreiras de acesso à tecnologia. São estas características que permitem que os dispositivos móveis tenham um grande impacto no desenvolvimento humano, já que promovem o desenvolvimento de todas as áreas ligadas ao Homem e às suas atividades, como: a governação, a saúde, a educação, a agricultura, o emprego, a prevenção de crises e o ambiente (Moedas, 2011)(United Nations, 2013).

Uma aplicação móvel, mais conhecida por *app* (do inglês *Application*) define-se como um *software* instalado num *smartphone* ou noutro dispositivo móvel, como um *tablet* ou um reprodutor de música, que permite que o utilizador tenha acesso a serviços que utiliza num computador, mas em regime de mobilidade (Janssen, 2014).

O mercado de aplicações móveis está em constante mudança e crescimento já que surgem, cada vez mais, novas aplicações para suprimir as necessidades dos utilizadores, tanto particulares como empresariais. Assim sendo, em 2013 foram ultrapassados os setenta mil milhões de *downloads* de aplicações móveis em Portugal, a sua grande maioria realizada através de *smartphones* (Casa dos Bits, 2013). Neste momento, o mercado encontra-se dividido graças à existência de diversos sistemas operativos, sendo que existem três deles com uma forte presença: o *Android*[3], o iOS[4] e o *Windows Phone*[5]. Esta fragmentação suscita limitações aos programadores aquando do desenvolvimento de uma aplicação, já que dificulta a criação de funcionalidades únicas que operem eficazmente em todas as plataformas de aplicações. Além disto, impõe-se o desafio de criar uma *app* que opere em todos os dispositivos móveis existentes no mercado, devido às características singulares que cada marca estabelece para os seus equipamentos (Moedas, 2011). No ano passado, foi o sistema operativo *Android* que dominou o mercado português, seguido do iOS e do *Windows Phone,* cuja presença se revelou mínima face à concorrência, com apenas 2% de *downloads* de aplicações (Casa dos Bits, 2013).

São diversos os aspetos que um programador deve ter em atenção quando planeia desenvolver uma aplicação. Em primeiro lugar, deve compreender as limitações associadas aos dispositivos móveis, já que cada um possui características próprias de navegação e físicas, como o tamanho e a sensibilidade do ecrã (Pereira, 2013). Além disto, o programador deve conhecer os diversos ambientes de desenvolvimento de aplicações bem como as diversas interfaces que necessitam de ser criadas de forma a garantir o funcionamento da aplicação (Moedas, 2011). Para além destes aspetos, uma decisão que terá de ser tomada pelo programador neste processo, já que determinará o processo de criação de uma aplicação: criar uma aplicação nativa, uma aplicação *web* ou uma aplicação híbrida (Lemos, 2013).

Uma aplicação nativa é criada e implementada especificamente para um sistema operativo (S.O.) no qual se integram, enquanto uma aplicação *web* funciona com base num navegador na Internet que interpreta a linguagem de programação (Pereira, 2013). A Tabela 1 demonstra as diferenças entre as aplicações nativas e as aplicações *web*, fundamentais para compreender em que sentido são vantajosas para a aplicação que se quer implementar, bem como para reconhecer as limitações de ambas:

[3] *Android*: http://www.android.com/

[4] iOS: https://www.apple.com/pt/ios/

[5] *Windows Phone:* http://www.windowsphone.com/

| | Aplicações nativas | Aplicações *web* |
|---|---|---|
| **Linguagem de programação** | Específica para o S.O.:<br>– *Android: Java*<br>– iOS: *Objective C*[6]*/Swift*[7]<br>– Windows *Phone*: C#[8] | Universal:<br>– HTML[9]<br>– CSS[10]<br>– *JavaScript*[11] |
| **Interação com o sistema operativo** | Direta | Indireta:<br>Camada intermédia entre a aplicação e o sistema operativo |
| **Interação com o *hardware*** | Pode aceder a todo o *hardware* do dispositivo | Define o *hardware* a que a aplicação acede |

*Tabela 1 - Aplicações Nativas vs. Aplicações web (Fonte: Pereira, 2013)*

Com o desenvolvimento tecnológico, surgiu a combinação de ambos os tipos de aplicações móveis de forma a suprimir as desvantagens de cada uma e tirar o máximo proveito das vantagens das mesmas: as aplicações híbridas. As últimas utilizam uma linguagem de programação única, tal como HTML5, CSS3 ou *JavaScript*. No entanto, correm num "recipiente" nativo, pelo que podem ser executadas em modo *offline*, bem como aceder às funcionalidades do *smartphone* através do *PhoneGap*. Tudo isto ocorre simultaneamente com a utilização da tecnologia *web* que permite que a aplicação seja executada em múltiplas plataformas, o que origina um processamento mais lento quando comparada com uma aplicação nativa derivado da possibilidade de diferentes interpretações da linguagem utilizada, nomeadamente o *JavaScript* (Lemos, 2013).

[6]*Objective-C:* https://developer.apple.com/library/mac/documentation/cocoa/conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html

[7]*Swift:* https://developer.apple.com/swift/

[8] C#: http://msdn.microsoft.com/pt-BR/library/z1zx9t92.aspx

[9] HTML: https://developer.mozilla.org/en-US/docs/Web/HTML

[10] CSS: https://developer.mozilla.org/en-US/docs/Web/CSS

[11] *Javascript*: https://developer.mozilla.org/en-US/docs/Web/JavaScript

## 3.2. APPLE

No primeiro dia de abril de 1976, Steve Jobs, Steve Wozniack e Ronald Wayne fundaram a empresa que deu origem a uma das atuais empresas de tecnologia líderes no mercado de dispositivos e aplicações móveis: a *Apple Inc*.. Três meses após a sua criação, Wayne decidiu vender a sua parte das ações e deixar os dois colegas, antigos colaboradores da *Atari* e da *Hewlett-Packard*, à frente da *Apple Computer*, uma empresa em constante crescimento e na linha da frente da inovação desde a sua fundação (TIME Inc., 2014). Desta forma, foram várias as apostas tecnológicas da *Apple* ao longo dos anos, nomeadamente:

1. *Apple* I (1976): modelo primitivo do *Personal Computer;*
2. *Macintosh/Mac* (1984): primeiro computador com um preço acessível aos utilizadores privados e com características únicas: interface gráfica inovadora (ícones, *desktop*); utilização do rato; duplo-clique e clicar-e-arrastar para ações com o rato; *"what you see is what you get"* (edição de texto e gráficos); nomes de arquivos longos; *design* estético e ergonómico (Pereira *et al.*, 2006)
3. *iMac* (1998): computador produzido para utilizadores sem experiência na área da informática e com um aspeto visual apelativo, motivo que impulsionou as vendas deste produto;
4. *iPod* (2001): *player* de música portátil com interface inovadora e capacidade de *download* de músicas com uma velocidade elevada;
5. *MacOS X* (2001): melhorias na interface e aspeto da máquina, maior velocidade de processamento e facilidade de utilização;
6. *iTunes Store* (2003): venda de música *online* com um ambiente *user friendly*, uma vasta biblioteca de multimédia e com possibilidade de ser utilizada no *iPod*;
7. *iPhone* (2007): resultado do sucesso do *iPod*, que permitiu à *Apple* a entrada no mercado dos *smartphones* e considerada a "Invenção do Ano" pela revista Time.
8. *iPad* (2010): entrada da *Apple* no mercado dos *tablets* (TIME Inc., 2014).

Apesar de se ter destacado em diversos campos da tecnologia, a *Apple* ultrapassou a concorrência quando criou o *iPhone*, que revolucionou a indústria das comunicações móveis em todo o mundo, dominando cerca de 53% do mercado e sendo

o telemóvel mais vendido em alguns países do mundo, como os Estados Unidos e o Japão (Neal, 2014). Mas quais são os motivos para o sucesso deste *smartphone*?

Um dos principais motivos para o sucesso do *iPhone* é o seu *design*, um aspeto em que a *Apple* sempre investiu para impulsionar a comercialização dos seus produtos e que sempre foi descurado pelas outras empresas de alta-tecnologia (Frommer, 2009). Assim, o aspeto físico e a interface centrados no ecrã foram um trunfo fundamental para a angariação de clientes, principalmente os que nunca tinham adquirido produtos desta empresa (Grossman, 2007).

Outro aspeto foi o *touchscreen* que, apesar de não ser uma invenção da *Apple*, foi utilizado pela primeira vez num telemóvel, o que revolucionou o mercado dos *smartphones* e criou uma interface inovadora: táctil, com controlo dos dados através do toque, o multitoque (*zoom*), o acelerómetro, ou rotação do *display,* e a rotação de ecrãs (Frommer, 2009). Além disto, a *Apple* conseguiu inserir num pequeno aparelho o seu sistema operativo conseguindo que este assumisse um carácter de portabilidade, permitindo o acesso às suas aplicações básicas, como a visualização de vídeos e a *App Store,* e a aplicações *web*, como os seus *browsers* (Grossman, 2007)(Frommer, 2009).

Por fim, foi o preço do *iPhone* que o tornou bastante competitivo face a outras opções no mercado, já que foi considerado razoável face à qualidade dos equipamentos e da tecnologia neles embutida. Para os programadores de aplicações móveis, também se tornou a opção mais viável para o lançamento de aplicações pois o mercado da *Apple* permite que eles obtenham 70% dos lucros da venda das suas criações. Estes dois fatores foram fundamentais para a mobilização de clientes e programadores de outras empresas de tecnologias móveis por parte da *Apple* e para a promoção do seu crescimento no seio das tecnologias móveis (Frommer, 2009).

### 3.2.1. iOS

Com o lançamento do primeiro *iPhone* foi criado um *Software Development Kit* (SDK) para que os programadores externos à *Apple* obtivessem as ferramentas e as interfaces necessárias para criar, instalar e correr aplicações nativas que funcionassem nos dispositivos da empresa (*Apple* Inc., 2014a) (Fonseca *et al.*, 2013). Foi apenas em 2010 que surgiu o iOS, um sistema operativo para dispositivos móveis, nomeadamente o *iPhone, iPad e iPod Touch*, cujo ambiente de desenvolvimento é o *XCode*. O último possui diversas *frameworks* que fornecem as interfaces necessárias para criar código (Fonseca *et al.*, 2013).

Este sistema operativo, que já se encontra na sua oitava versão, é o responsável pela gestão do *hardware* e é a fonte dos instrumentos necessários para a implementação de aplicações nativas que se encontram sempre disponíveis para utilização, mesmo em "Modo Avião". Assim, o iOS é um intermediário entre o *hardware* e a aplicação criada, já que define interfaces de sistema que permitem a comunicação entre ambo. Assim, permite escrever aplicações nativas que funcionam de igual forma em dipositivos com diferentes características de *hardware*, desde que operem com iOS. Para a implementação das tecnologias, este sistema operativo foi desenhado em camadas, sendo que as inferiores, o *Core OS* e o *Core Services*, representam os serviços básicos e as superiores, *Media* e *Cocoa Touch*, figuram serviços mais especializados e sofisticados, ilustrado pela Figura 3 (Apple Inc., 2014a).

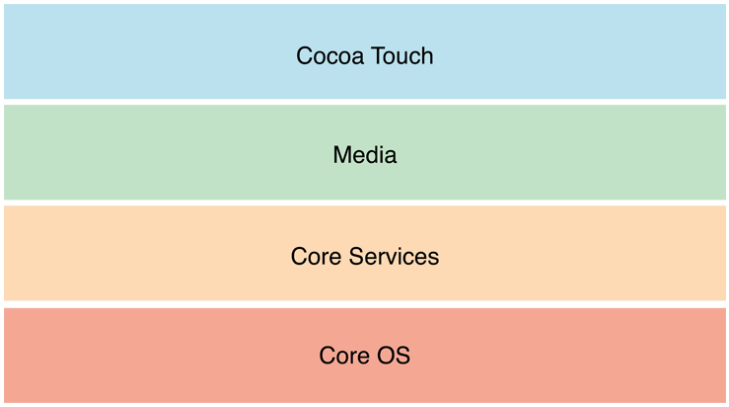


*Figura 3 - Camadas do iOS (Fonte: Apple Inc., 2014)*

O *Cocoa Touch* é a *framework* que a maioria dos programadores utiliza para desenvolver aplicações móveis e que, pela sua vez, se divide em duas partes: *Framework Foundation* e *Framework UIKit.* A primeira providencia classes de suporte à criação de *apps*, nomeadamente *strings*, números, coleções, etc., enquanto a segunda suporta a construção de interfaces gráficas, como janelas, controlos, vistas, entre outros (Fonseca *et al.*, 2013).

O desenvolvimento de *software* em iOS apenas fica completo com a fase de testes da aplicação e a sua submissão para avaliação para posterior colocação no mercado através da *App Store*, onde é disponibilizada aos utilizadores dos dispositivos (Fonseca *et al.*, 2013).

### 3.2.2. XCode

O *XCode* é o ambiente de desenvolvimento integrado para as plataformas Apple (MacOs e iOS) que proporciona um desenvolvimento aplicacional rápido e inclui um *Interface Builder*, que possibilita a criação de interfaces gráficas através de uma *storyboard.* A última surgiu com o iOS 5 e define-se como um ficheiro que reúne toda a informação da interface gráfica de uma aplicação agrupando os ecrãs, as vistas, e os controladores da mesma (Fonseca *et al.*, 2013).

O *XCode* caracteriza-se, igualmente, por apresentar uma interface única onde é possível editar código, construir a interface do utilizador, gerir dados e testar a aplicação. Todas estas funcionalidades encontram-se numa janela única, que se modifica de acordo com a tarefa que está a ser executada, o que permite ao programador que se concentre numa única tarefa (Apple Inc., 2014b). A Figura 4 ilustra a interface do *XCode* e as cinco áreas principais da mesma: a barra de ferramentas (*Toolbar*); a área de edição de código (*Editor area*), a área de serviços (*Utilities area*); a área de *debug* e a área de navegação (*Navigator area*).

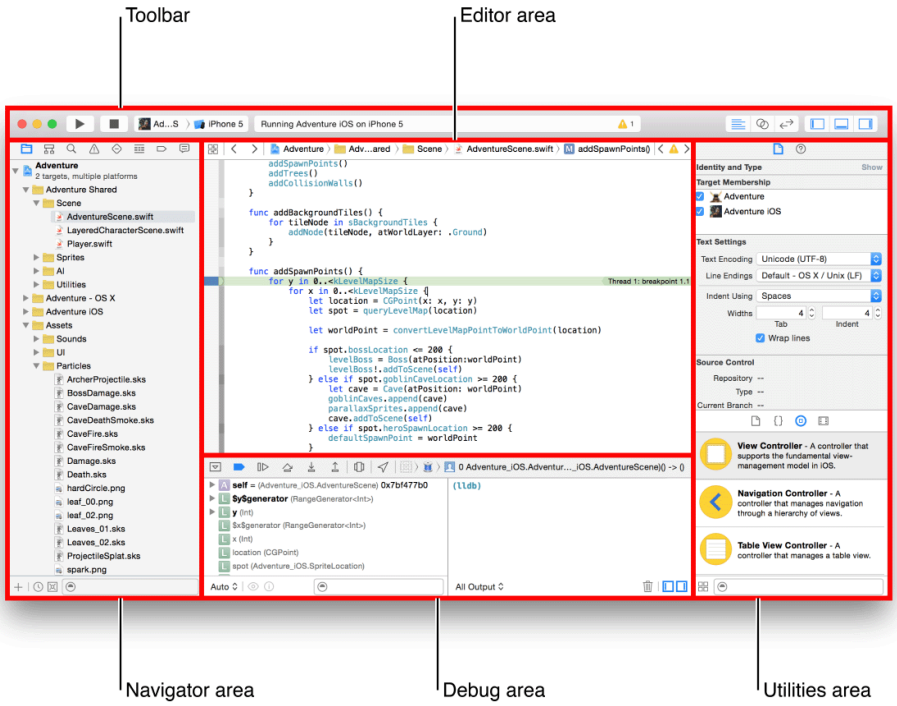


*Figura 4 – Interface do XCode (Fonte: Apple Inc., 2014b)*

São diversas as possibilidades que o *XCode* oferece aos programadores para que estes desenvolvam uma aplicação. Em primeiro lugar, permite a criação e edição de código assistida, já que deteta erros no código-fonte e emite alertas para que estes sejam corrigidos. Ademais, permite construir uma *User Interface* (UI), pelo que é possível selecionar objetos de uma biblioteca que são configuráveis e construir uma *storyboard*, de forma a definir a sequencialidade da aplicação. Permite, também, realizar *debugging* integrado, na qual o *XCode* lança a aplicação e começa a sessão de depuração no iOS *Simulator* ou no próprio *Mac*, sendo possível também realizar esta opção no código-fonte (Apple Inc., 2014b).

Por fim, o *XCode* fornece apoio ao lançamento de uma aplicação no sentido em que executa testes e integrações contínuas da funcionalidade e da performance através de uma *framework*, gravando automaticamente todo o trabalho que é realizado. Quando é verificado que a aplicação não possui erros e se encontra com a adequada eficiência, este ambiente de desenvolvimento providencia ferramentas para a preparação da admissão da aplicação à *App Store* e permite a sua distribuição pelos *beta-testers* (Apple Inc., 2014b).

## 3.3. LINGUAGENS DE PROGRAMAÇÃO EM IOS

Desde a criação do iOS que a linguagem de programação utilizada é o *Objective-C*, uma extensão da linguagem C cuja programação é orientada a objetos (Fonseca *et al.*, 2013). No entanto, recentemente surgiu o *Swift*, uma linguagem que se caracteriza pela inovação e por permitir escrever código de forma interativa, com uma sintaxe mais sucinta e com uma rapidez de processamento superior face ao *Objective-C* (Apple Inc., 2014c). Seguidamente, serão exploradas as principais características destas duas linguagens de forma a compreender as diferenças entre elas.

### 3.3.1. Objective-C

O *Objective-C* é a linguagem de programação mais conhecida e utilizada do iOS que incorpora conceitos de programação orientada a objetos da linguagem *SmallTalk* e é uma extensão da Linguagem C, com tempo de execução da aplicação dinâmico (Fonseca *et al.*, 2013).

Esta linguagem é diferente das linguagens C, C++ e *Java* no sentido em que o modelo de programação é baseado no envio de mensagens entre objetos, enquanto nas outras linguagens referidas os métodos são aplicados diretamente num objeto. Apesar de facultar a sintaxe, todos os aspetos relacionados com a interação com o utilizador, o acesso à rede e aos dados é proporcionado pelo *Cocoa Touch* (Fairbairn, Fahkrenkrug & Ruffenach, 2012).

Através do *SmallTalk*, foram importadas diversas características desta linguagem de programação, que permitem implementar o comportamento da aplicação através da escrita de código. Assim, são diversos os aspetos a considerar em *Objective-C*: Objetos, Classes, Mensagens e Métodos, e Protocolos (Fonseca *et al.*, 2013).

#### Objetos

Os objetos definem-se como a concretização particular de uma classe, sendo as unidades básicas da mesma. A classe descreve o tipo de dados que um objeto deve armazenar, bem como o tipo de operações que ele pode realizar (Fonseca *et al.*, 2013)(Fairbairn *et al.*, 2012).

Os objetos, além de construírem as classes, agrupam os dados com comportamentos relacionados e comunicam com outros objetos para resolver problemas. Para criar um objeto e para permitir que este funcione é necessário criar uma instância de uma classe particular: primeiro realiza-se a alocação do bloco de memória que guardará a instância; e, em segundo, é realizada a iniciação das variáveis da instância (Fonseca *et al.*, 2013) (Apple Inc., 2013).

## Classes

Uma classe é o projeto da estrutura de um ou mais objetos num sistema que têm objetivos semelhantes e que contêm os dados e métodos para controlo dos dados. Além disso, é nas classes que os objetos realizam diversas operações e se define o comportamento e as propriedades de um tipo de objeto (Fonseca *et al.*, 2013)(Fairbairn *et al.*, 2012). Esta é a forma mais básica de encapsulamento em *Objective-C*, já que a classe reúne um pequeno número de dados com um conjunto de funções vitais que manipulam a informação, permitindo que apenas a interface pública dos objetos seja visível, para que os conteúdos dos objetos sejam apenas restritos aos mesmos (Fonseca *et al.*, 2013)(Fairbairn *et al.*, 2012) (Apple Inc., 2013).

Um aspeto fundamental nesta linguagem é, também, a herança que permite que uma nova classe seja uma especialização da anterior, definindo comportamentos e características adicionais à classe-mãe e prevenindo a duplicação de classes (Apple Inc., 2013).

## Mensagens e Métodos

Em *Objective-C* os objetos comunicam através de mensagens criando um método, ou função, que permite a manipulação dos dados. Para que um método possa ser declarado, é necessário indicar um de dois tipos: métodos de instância, que acedem às variáveis das instâncias das classes; ou métodos de classe, que se aplicam a toda a classe e não têm acesso às suas instâncias ou variáveis (Fonseca *et al.*, 2013).

Para que um programador consiga criar a interação entre objetos necessita de empregar uma sintaxe que utilize parênteses retos. Para tal tem que, em primeiro lugar, definir o destinatário, seguido da mensagem a ser enviada (Fairbairn *et al.*, 2012). Na

Figura 5 é possível compreender o mecanismo de envio de mensagens no qual em 1 é visível a execução da linha de código e, em 2, o envio da mensagem:

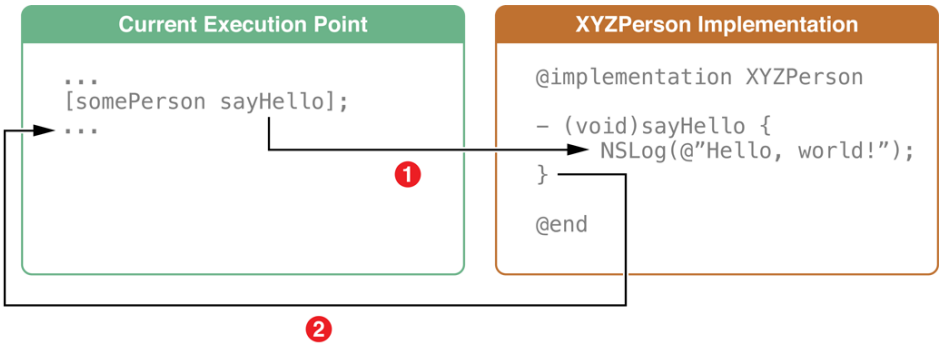


*Figura 5 - Mensagens entre Objetos (Fonte: Apple Inc., 2013)*

Por vezes, para que um objeto possa compreender a mensagem, é necessário adicionar informação, que é representada por um ou mais valores chamados argumentos. Nesta linguagem, as mensagens podem ser enviadas entre objetos e para classes, de forma a obter ou requisitar informações a uma instância específica da classe (Fairbairn *et al.*, 2012).

## Protocolos

Em *Objective-C*, os protocolos definem a interação entre os objetos estabelecendo um conjunto de comportamentos específicos para situações comuns. Assim, um protocolo assume-se como uma interface de programação que qualquer classe pode implementar, permitindo que comunique com outras que se encontrem relacionadas com ela à distância (Apple Inc., 2013). Na figura 6 é possível compreender de que forma as classes transmitem comportamentos entre si:

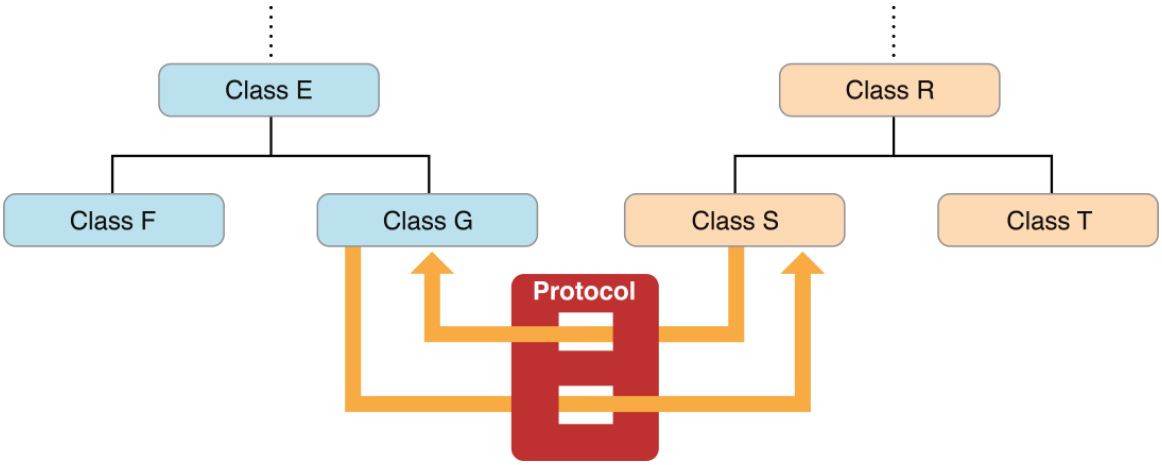


*Figura 6 – Transmissão de comportamentos entre classes através de protocolos (Fonte: Apple Inc., 2013)*

Em *Java* ou C# um protocolo é semelhante ao conceito de uma interface e, como ocorre em *Objective-C*, uma classe pode adotar diversos protocolos, bem como um protocolo pode adotar múltiplos protocolos, estabelecendo uma herança múltipla. Desta forma, é possível separar objetos e permitir que comuniquem de forma independente à classe a que pertencem. Este é um aspeto de particular importância em três ocasiões aquando da construção do código-fonte da aplicação: declaração de métodos que se supõe que serão implementados por outras classes; declaração da interface de um objeto, sem expor a sua classe; e, por fim, captação de comportamentos semelhantes entre classe não relacionadas hierarquicamente (Fonseca *et al.*, 2013) (Fairbairn *et al.*, 2012).

### 3.3.2. Swift

O *Swift* é a linguagem de programação recentemente apresentada pela *Apple* que reúne diversas características da Linguagem C e do *Objective-C*, que se prevê que será suplantado por esta nova linguagem. O *Swift* utiliza o modelo de programação de objetos dinâmicos, pelo que é uma linguagem orientada a objetos, bem como utiliza o *Cocoa Touch* como *framework* e o *XCode* como ambiente de desenvolvimento, tal como o *Objective-C* (Apple Inc., 2014c).

Apesar de se basear em *Objective-C* e ter sido concebido segundo a mesma filosofia de programação, o *Swift* apresenta várias diferenças. Em primeiro lugar, é considerado mais moderno, no sentido em que a sintaxe é mais sintética e se aproxima

de linguagens mais recentes como *Python* e *Ruby* (Apple Inc., 2014c). A análise da Figura 7 ilustra este aspeto já que é possível observar que o *Swift* emprega uma sintaxe mais simplista e concisa que o *Objective-C* para declarar um *Array*:

Objective-C

```
NSArray *myArray = [[NSArray alloc] initWithObjects:@"one", @"two", @"three", nil];
```

Swift

```
var myArray = ["one", "two", "three"]
```

*Figura 7 - Diferenças de sintaxe entre Objective-C e Swift na declaração de um Array (Fonte: Clifton, 2014)*

No *Objective-C* é necessário realizar a alocação de memória na chamada do método, enquanto no *Swift* este processo é realizado de forma automática. Tal facto demonstra que esta nova linguagem apresenta um maior grau de facilidade na construção do código-fonte, bem como a incapacidade do programador de interagir a um nível mais básico. É possível comparar o processo de chamada de um método pelas duas linguagens através da análise da Figura 8:

*Objective-C*

```
1 SpecificCar *myCar = [[SpecificCar alloc] init];
2 [myCar getOdometerReading];
```

*Swift*

```
1 var myCar = SpecificCar()
2 myCar.getOdometerReading()
```

*Figura 8 - Diferenças na chamada de métodos entre Objective-C e Swift (Fonte: Ching, 2014)*

Para além dos aspetos mencionados anteriormente, o *Swift* apresenta *playgrounds* interativos nos quais o resultado de uma linha de código aparece

imediatamente numa janela adjacente, tal como é observável na Figura 9 (Apple Inc., 2014c):

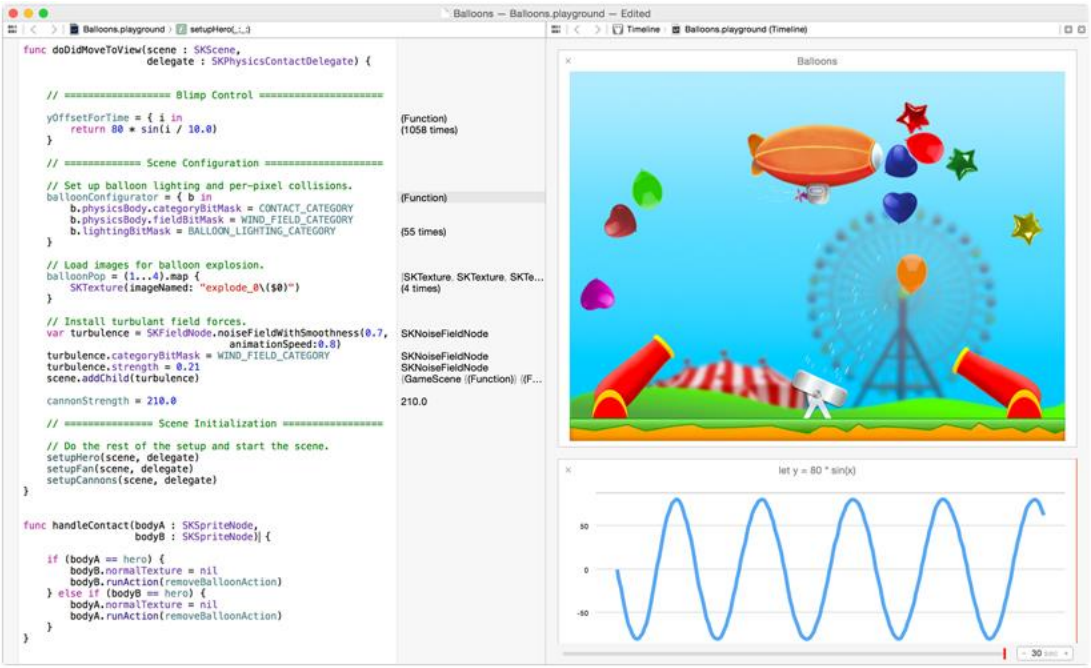


*Figura 9 – Playground em Swift (Fonte: Apple Inc., 2014c)*

Outra particularidade fundamental no *Swift* é a segurança. Esta linguagem é considerada mais segura que o *Objective-C*, no sentido em que assinala automaticamente todas as classes que contenham código considerado inseguro, bem como procede à alocação e inicialização das variáveis automaticamente antes da sua utilização. Ademais, o *Swift* é considerado mais rápido e eficiente já que o código criado para o desenvolvimento de aplicações nativas possibilita uma elevada *performance* das mesmas. Isto deve-se ao seu desempenho de encriptação otimizado face a *Objective-C*, no qual o *Swift* pode ser integrado, tal como em C e *Python* (Apple Inc., 2014c).

Por fim, destaca-se uma vantagem do *Swift* face à maioria das outras linguagens de programação: o facto de uma única linha de código poder gerar um programa. Ou seja, uma linha única de código tem a capacidade de gerar o mesmo resultado de múltiplas linhas de código escritas noutra linguagem. Assim, não é necessário importar

uma biblioteca para permitir a funcionalidade do mesmo, o *input/output* ou a gestão das *strings* o que diminui, desta forma, os obstáculos que possam existir ao desenvolvimento de uma aplicação (Apple Inc., 2014d).

# 4. PROJETOS DESENVOLVIDOS

No estágio realizado na PT Inovação e Sistemas foram desenvolvidos vários projetos, denominados em ambiente de estágio como unidades de desenvolvimento, que irão ser explorados e analisados seguidamente, o que permitirá expor o trabalho que foi desenvolvido no decorrer do referido estágio. Destacam-se os seguintes projetos desenvolvidos:

1. O estudo de desenvolvimento de aplicações móveis multiplataforma e as respetivas tecnologias associadas: o *PhoneGap* e o *Appcelerator Titanium*;
2. A implementação de testes automatizados específicos para aplicações iOS através do *Google Analytics* de forma a obter dados estatísticos sobre a utilização de uma aplicação
3. A introdução de variáveis de análise de comportamento e estatística em aplicações móveis iOS pelo *UI Automation* e os testes automatizados em dispositivos móveis com o *AppThwack*;
4. A demonstração das potencialidades da nova linguagem de programação lançada pela *Apple*, o *Swift*; e a exploração de uma aplicação criada nesse âmbito;
5. A criação uma aplicação iOS móvel capaz de detetar *iBeacons* e aceder a serviços consoante a localização espacial, o tempo, ou qualquer outra variável pertinente, denominada *iDoor*.

## 4.1. APLICAÇÕES MULTIPLATAFORMA

Com o aumento do número de utilizadores de dispositivos móveis, a diversidade de sistemas operativos e a necessidade, por parte das empresas, de incrementar o leque de oferta no mercado das aplicações móveis, tornou-se imprescindível a criação de *frameworks* que permitissem desenvolver aplicações para diferentes clientes, com sistemas operativos distintos. Assim, uma *framework* de desenvolvimento multiplataforma móvel é uma ferramenta de desenvolvimento que permite criar uma aplicação única que opere em múltiplos sistemas operativos móveis, tendo como referências o *Android* e o iOS (Lemos, 2013).

Atualmente são reconhecidas numerosas soluções que permitem desenvolver aplicações para multiplataformas móveis pelo que se destacam como exemplos os seguintes: *PhoneGap*, *Appcelerator Titanium*, *JQuery Mobile*[12],*Sencha Touch 2*[13], *Monosuite* (*Xamarin.Android* e *Xamarin.iOs*)[14]; *RhoMobile*[15] e *Corona*[16] (Dalmasso *et al.*, 2013). Todas estas ferramentas permitem a criação de aplicações para *Android* e iOs no entanto, o *Windows Phone* foi descartado por algumas delas.

Todas as plataformas referidas apresentam diferenças já que algumas possuem uma maior capacidade de personalização das interfaces, enquanto outras apresentam um nível superior de compatibilidade com as funcionalidades e o *hardware* dos *smartphone*s. Graças à variabilidade de *frameworks* e ao prazo definido para a execução de um relatório para a PT Inovação e Sistemas sobre esta temática, foi estipulado o seguinte conjunto de critérios para limitar e definir as ferramentas a analisar:

- A capacidade de desenvolver uma aplicação que opere em simultâneo em iOS e *Android* que esteja apta para a compatibilidade com *Windows Phone 8*, apesar deste não ser um aspeto prioritário;
- A existência de Interfaces de Programação de Aplicações (APIs) por omissão, similares às existentes em linguagens nativas e a compatibilidade da generalidade das plataformas com as APIs implementadas;
- A necessidade de compatibilidade com *stream* de vídeo (HLS, *Smooth Stream* IIS);
- A facilidade de licenciamento ou custo-zero de licenciamento;
- A compatibilidade e possibilidade de funcionamento com as principais potencialidades de *hardware*: acelerómetro, câmara fotográfica, interação com contactos, geolocalização, gestão e visualização multimédia, e armazenamento.

Respeitando os critérios definidos anteriormente, foram escolhidas as plataformas *PhoneGap* e *Appcelerator Titanium* como as melhores alternativas à programação *mobile* em linguagem nativa pelo que são apresentadas as diversas propriedades de cada uma delas.

---

[12] *JQuery Mobile:* http://jquerymobile.com/

[13] *SenchaTouch 2*: http://www.sencha.com/products/touch/

[14] *Monosuite* (*Xamarin.Android* e *Xamarin.iOs*): http://xamarin.com/platform

[15] *RhoMobile*: http://rhomobile.com/

[16] *Corona*: http://coronalabs.com/

### 4.1.1. PhoneGap

O *PhoneGap*[17] é uma das *frameworks* mais utilizadas no seio dos programadores de aplicações móveis e é uma plataforma de desenvolvimento *open source*, pelo que a criação de uma aplicação privada é gratuita. No entanto, é exigido um custo se a *app* for produzida com vista à sua venda e distribuição. São três as linguagens utilizadas nesta plataforma, nomeadamente HTML, CSS e *JavaScript* (Corral, Sillitti and Succi, 2012).

Através da análise da Figura 10, é possível constatar que o *PhoneGap* apresenta duas camadas distintas: uma camada de nível superior responsável pela apresentação de conteúdos e onde é realizada a interface com o utilizador em HTML e CSS; e uma segunda camada, de nível inferior, onde é criada a estrutura da aplicação construída através de *JavaScript* (Corral *et al.*, 2012):

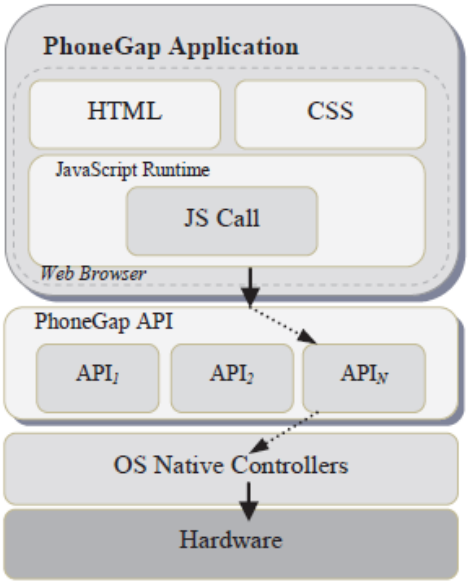


*Figura 10 – Camadas de uma aplicação em PhoneGap (Fonte: Corral et al., 2012)*

O *PhoneGap* apresenta diferentes compatibilidades com diversas funcionalidades de alguns *smartphones,* consoante o sistema operativo dos mesmos. O estudo desta compatibilidade é apresentado na Tabela 2:

[17] *PhoneGap:* http://phonegap.com/

| | ***iPhone* 3G** | ***iPhone* 3GS+** | ***Android*** | ***Blackberry* OS 6.0+** | ***WebOS*** | ***Windows Phone* 7+8** | ***Symbian*** |
|---|---|---|---|---|---|---|---|
| Acelerómetro | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Câmara | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Bússola | X | ✓ | ✓ | X | ✓ | ✓ | X |
| Contactos | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ |
| Gestão de Ficheiros | ✓ | ✓ | ✓ | ✓ | X | ✓ | X |
| Geolocalização | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| *Media* | ✓ | ✓ | ✓ | X | X | ✓ | X |
| Rede | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Notificações | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Armazenamento | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | X |
| DRM | X | X | X | X | X | X | X |

*Tabela 2 - Compatibilidade do PhoneGap em vários smartphones*

Todas as funcionalidades exibidas na tabela anterior são garantidas através de APIs específicas de cada sistema operativo móvel. No entanto, as interfaces que podem ser desenvolvidas através desta ferramenta são pouco dotadas no sentido em que a experiência de utilização fica comprometida graças à complexidade de replicação do *"look and feel"* de uma aplicação nativa. Quanto ao desempenho do *PhoneGap* é consensual entre vários autores que a utilização de *frameworks* de desenvolvimento multiplataforma influencia negativamente o desempenho das aplicações (Dalmasso *et al.*, 2013) (Corral *et al.*, 2012).

Não é clara a prova de veracidade das afirmações acima descritas pois não existem estudos suficientes que as sustentem, já que o *PhoneGap* é utilizado em parceria com outras ferramentas, como por exemplo o *JQuery Mobile* ou o *Sencha Touch 2*. Porém, existem estudos que defendem que o *PhoneGap* é a plataforma que menos recursos utiliza em termos de CPU, memória RAM e energia elétrica devido à débil componente gráfica que apresenta. No entanto é importante realçar que apresenta um desempenho inferior face às aplicações nativas (Corral *et al.*, 2012).

O *site* oficial do *PhoneGap* disponibiliza um conjunto de guias para os programadores que explicitam de que forma se processa a instalação e utilização da *framework*, bem como a utilização das APIs que foram desenvolvidas. Sendo uma das ferramentas mais populares no seu domínio, existem inúmeros tópicos *online* acerca do *PhoneGap*, pelo que é possível destacar como exemplos as 40 mil entradas no *Stackoverflow*[18] e os cerca de 3 milhões de resultados de uma pesquisa rápida no *Google*, o que comprova a vastidão de suporte técnico aliado a esta plataforma.

### 4.1.2. Appcelerator Titanium

O *Appcelerator Titanium*[19] possibilita a criação de uma aplicação *mobile* e todos os serviços adjacentes de forma simultânea e numa única operação, publicando-a posteriormente nos diferentes sistemas operativos. Esta *framework* possui o seu próprio ambiente integrado de desenvolvimento de *software* (IDE) baseado no *Eclipse* e utiliza a linguagem *JavaScript* para a criação de código.

Para além das características acima referidas, esta multiplataforma faculta APIs independentes que permitem o acesso nativo a componentes da *user interface* (UI), incluindo barras de navegação, menus, caixas de diálogo, alertas. Além disto, permite o acesso a funcionalidades nativas como o sistema de ficheiros, som, *networking*, bases de dados locais, entre outras e às APIs dessas funcionalidades como a geolocalização, acelerómetro e mapas. Ademais, possui um mercado próprio que permite ao programador adquirir licenciamento para APIs proprietárias, bem como suporte a *codecs* para reprodução multimédia, funcionalidades diferenciadas e *scripts* adicionais.

Quanto ao desempenho desta tecnologia, é necessário ter em consideração que todo o código-fonte da aplicação é interpretado utilizando um motor de *JavaScript* no dispositivo móvel: *Rhino*[20] da *Mozilla* para *Android* e *BlackBerry*; e *JavaScriptCore*[21] da *Apple* no caso do iOS. Em termos práticos, a aplicação apresentará menor desempenho do que aquelas que são desenvolvidas com os SDKs nativos, devido à necessidade de carregar todas as bibliotecas e o interpretador, antes deste compreender o código fonte, o que não ocorre nas aplicações nativas.

---

[18] *Stackoverflow*: http://stackoverflow.com/

[19] *Appcelerator Titanium*: http://www.appcelerator.com/titanium/

[20] *Rhino*: https://developer.mozilla.org/en-US/docs/Mozilla/Projects/Rhino

[21] *JavaScriptCore:* https://developer.apple.com/library/mac/documentation/Carbon/Reference/WebKit_JavaScriptCore_Ref/_index.html

Tem sido relatado por diversos programadores que existem questões em torno do comportamento das APIs multiplataforma que os incita a reescrever as suas aplicações anteriormente realizadas em código nativo, nomeadamente a estabilidade e a gestão da memória. No entanto, o *Appcelerator Titanium* demonstra resultados rápidos e ser adequado para prototipagem, pelo que a partir de 2013 houve mais de 55 mil aplicações construídas nesta *framework* e enviadas para as *Stores* (Appcelerator, 2012). Segundo a APPCELERATOR (2012: 6):

"*Appcelerator's customers include NBC, PayPal, eBay, Orange, and Cisco. These companies develop their mobile applications on Appcelerator products so they can decrease time-to-market and development costs, increase customer adoption and revenues, and enjoy greater flexibility and control*."

Em relação à documentação de apoio ao programador disponível, o *Appcelerator Titanium* faculta no seu *site* uma secção de tutoriais, apesar de possuir algumas limitações ao nível de suporte de problemas por parte de outros utilizadores. Devido à maior complexidade inerente ao desenvolvimento nesta *framework*, o número de programadores que a utiliza é menor comparativamente ao *PhoneGap* para desenvolvimento móvel. No entanto, uma pesquisa no *Google* consegue reproduzir mais de 850 mil resultados.

### 4.1.3. Testes das Plataformas

A necessidade de realizar uma comparação entre as diferentes tecnologias de desenvolvimento multiplataformas suscitou a criação de uma aplicação com uma temática ligada às atividades realizadas no MTV: o programa televisivo "*Secret Story*". O critério para o seu desenvolvimento foi a uniformidade face às diferentes tecnologias.

Os objetivos definidos para a criação da aplicação em causa foram os seguintes: manter a simplicidade; possibilitar ao utilizador a visualização de *streams* de vídeo em direto; permitir a consulta dos vídeos dos melhores momentos da "*Secret Story*"; proporcionar informações relativas a cada concorrente; e aceder aos dados das votações, mais concretamente os contactos telefónicos para proceder à votação num determinado concorrente. A aplicação móvel foi implementada em *PhoneGap* e *Appcelerator Titanium*, sendo realizada uma posterior comparação das vantagens e desvantagens de cada uma das ferramentas.

## Aplicação de Teste em *PhoneGap*

Para construir a aplicação de teste mencionada anteriormente foi desenvolvida, primariamente, uma aplicação simplificada que utilizou o *PhonegGap* e o *JQuery Mobile* para o sistema operativo *Android*, com o intuito de verificar o comportamento da plataforma perante a reprodução de um *live stream* de vídeo e de vídeos em MPEG-4. Numa fase posterior também foi testada a aplicação em iOS. Assim sendo, a aplicação foi construída com quatro menus principais:

1. O primeiro menu apresenta a informação relativa a cada um dos concorrentes, informação foi obtida através de fontes RSS disponibilizadas para este âmbito;
2. O segundo menu apresenta os vídeos mais recentes do "*Secret Story*";
3. O terceiro apresenta uma *live stream* de um canal televisivo como o AXN e a RTP 1;
4. No último menu é possível constatar se existe alguma votação ativa e quais os contactos telefónicos que os espetadores podem usar para expulsar um concorrente.

Na Figura 11, é possível visualizar a primeira categoria da aplicação e, simultaneamente, a *homepage* da mesma. A categoria referida consiste na reprodução de *stream* de vídeo e, organizado numa *scrollview* localizada abaixo dessa mesma reprodução de vídeo, a possibilidade de escolher outros conteúdos a reproduzir.

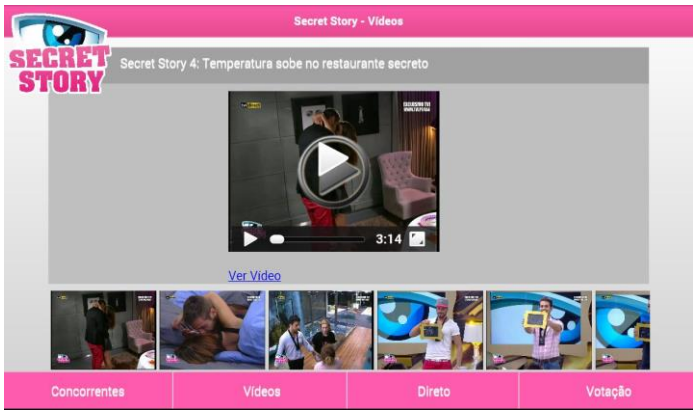


*Figura 11 –Interface da Homepage da Aplicação*

A Figura 12 corresponde à categoria de vídeos da aplicação, onde há a possibilidade de o utilizador escolher qualquer vídeo que pretenda reproduzir. Estes vídeos correspondem a ficheiros que estão no formato MPEG-4 e que se encontram alojados num servidor externo, cujo endereço é facultado pelo serviço providenciado para a aplicação.

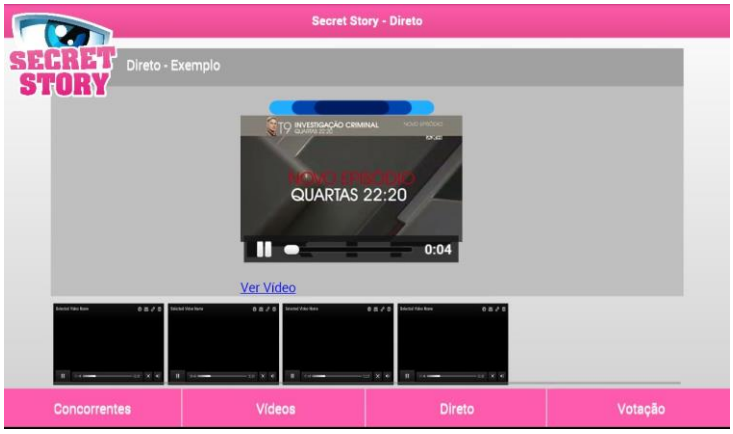


*Figura 12 – Interface da página de vídeos da aplicação*

A Figura 13 mostra a página da aplicação onde é mostrada a fotografia de cada concorrente, bem como uma listagem de todos os concorrentes em que, após a seleção de um deles, é possível aceder a informações a eles correspondente.

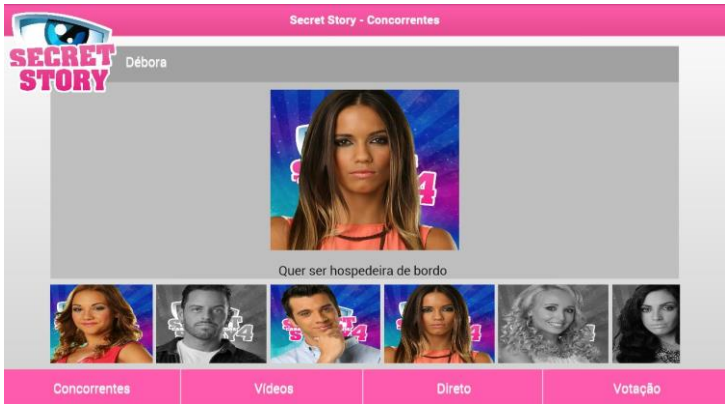


*Figura 13 – Interface da página Lista dos Concorrentes*

Sendo a reprodução de um *stream* de vídeo um dos maiores desafios do desenvolvimento da aplicação de teste, foram realizados ensaios em várias versões dos sistemas operativo *Android*. Desta forma foi possível verificar que existiam discrepâncias no comportamento da reprodução de *streams* de vídeo, consoante a versão do sistema operativo, como é possível compreender pela análise da Tabela 3:

<table>
<tr><th>Versão Android</th><th>Linguagem Nativa</th><th>PhoneGap</th></tr>
<tr><td>3.1</td><td>Apresentação normal dos vídeos MPEG-4.<br>O stream de vídeo só é reproduzido no formato RTSP.</td><td>Apresentação normal dos vídeos MPEG-4.<br>Impossibilidade de apresentação de streams de vídeo.</td></tr>
<tr><td>4.0.4</td><td rowspan="2">Apresentação normal dos vídeos MPEG-4.<br>O stream de vídeo é reproduzido em HLS e RTSP.</td><td>Apresentação normal dos vídeos MPEG-4.<br>O stream de vídeo é reproduzido em HLS e RTSP.</td></tr>
<tr><td>4.4</td><td>Apresentação normal dos vídeos MPEG-4.<br>O stream de vídeo é reproduzido em HLS e RTSP apesar da existência de alguns bugs.</td></tr>
</table>

*Tabela 3 –Linguagem Nativa vs. PhoneGap*

Como é possível concluir através da análise da tabela anterior, o *Android* ainda apresenta alguns problemas na apresentação de conteúdos em direto já que as versões mais antigas deste sistema operativo, como a *Honeycomb*, não conseguem reproduzir conteúdo em HLS. Foi encontrada, então, uma solução híbrida, através da utilização do *player* nativo para a reprodução de um *stream* em RTSP. Porém, esta poderá não ser uma solução satisfatória uma vez que, quando se procede à reprodução do vídeo, todo o menu fica inoperável. Além disto, a apresentação de conteúdos multimédia na versão 3.1 do *Android* apresenta algumas limitações, pelo que compromete a experiência de utilização. No caso do iOs não ocorreu nenhum erro na reprodução dos vídeos em MPEG-4 e da *stream* dos mesmos, tanto na versão iOs 6, quer na iOs 7.

O *PhoneGap* revelou-se, então, uma boa solução para aplicações simples que não necessitem de um grande poder de processamento. No entanto, a experiência de utilização fica comprometida devido à escassa complexidade das interfaces que esta ferramenta oferece. A aplicação de *JQuery Mobile* proporciona uma maior variedade de eventos e elementos gráficos, do que aqueles que seriam expectáveis para uma solução que utilizasse exclusivamente o *PhoneGap*.

## Aplicação de Teste em *Appcelerator Titanium*

Para testar as propriedades de desenvolvimento de aplicações móveis do *Appcelerator Titanium*, foi criada uma aplicação similar à apresentada anteriormente e cujo objetivo é a apresentação do conteúdo do programa televisivo "*Secret Story*" de forma interativa. A figura 14 ilustra o fluxograma dessa mesma aplicação:

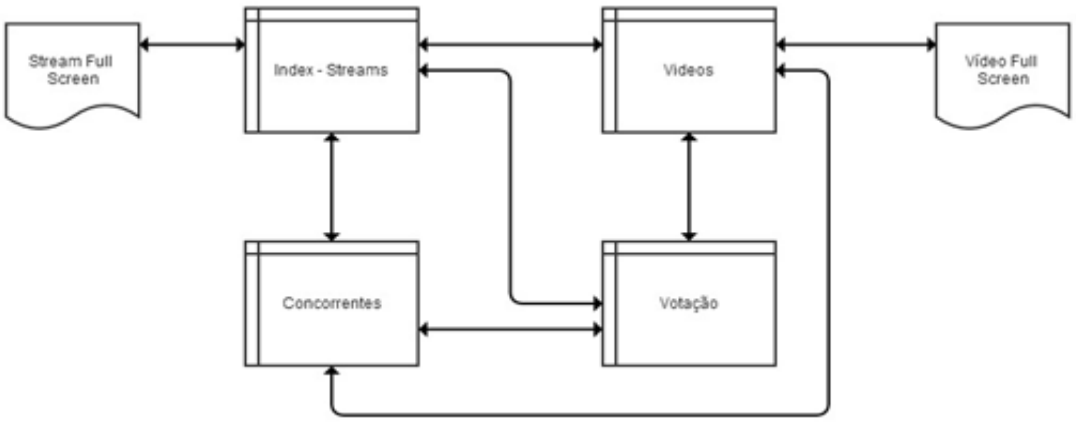


*Figura 14 - Fluxograma da Aplicação "Secret Story" em Appcelerator Titanium*

A aplicação apresenta quatro "*main Windows*" que se baseiam especificamente na apresentação de *streams* de vídeo com o objetivo de divulgar as diferentes câmaras da casa onde se realiza o *reality show.* Além disto, contém a apresentação de vídeos, a informação dos concorrentes e por último informação relativa às votações. É apresentada, na Figura 14, a organização simplista das quatro janelas da aplicação, bem como é explicitado de que forma é possível aceder a diversas janelas através da janela principal, exceto quando existe a exibição de um vídeo em *full screen*, obrigando a uma regressão à mesma.

A aplicação desenvolvida em *Appcelerator Titanium*, à semelhança da aplicação construída em *PhoneGap*, apresenta streams de vídeo na sua página inicial como é visível na Figura 15. Nesta área é possível escolher qual a stream de vídeo a visualizar.


*Figura 15 – Interface da página principal da aplicação*

Relativamente à organização das interfaces, optou-se por uma organização simplista através da sua divisão em duas regiões: a superior, onde é disponibilizada a informação, ou área de conteúdos; e a inferior onde é possível realizar a seleção dos conteúdos que se pretende visualizar através da execução de um *scroll* horizontal para aceder aos mesmos. Na figura 16 é exibida a interface da aplicação apresentada no simulador de *iPad* do *XCode*, relativa aos vídeos em MPEG-4, onde é permitido visualizá-los em *live stream.*

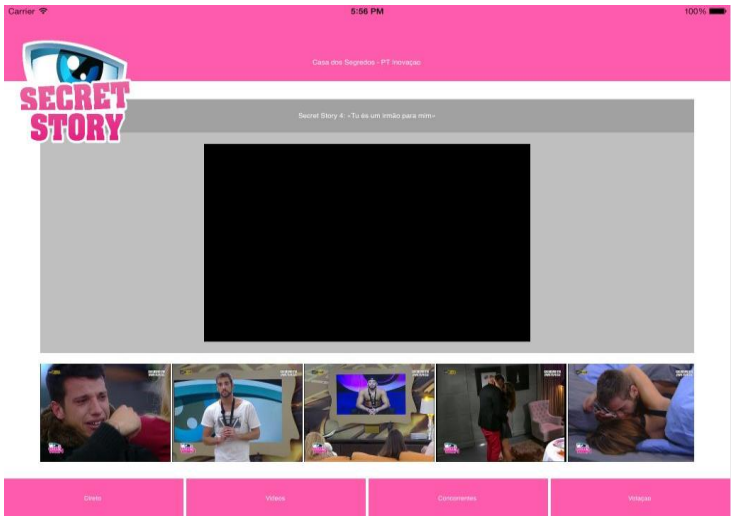


*Figura 16 – Interface de vídeos em live stream*

Como é possível visualizar na Figura 17, a aplicação exibe horizontalmente os candidatos, na região inferior, cujos perfis após seleção através de um clique, são acedidos e é possível consultar os seus respetivos dados.

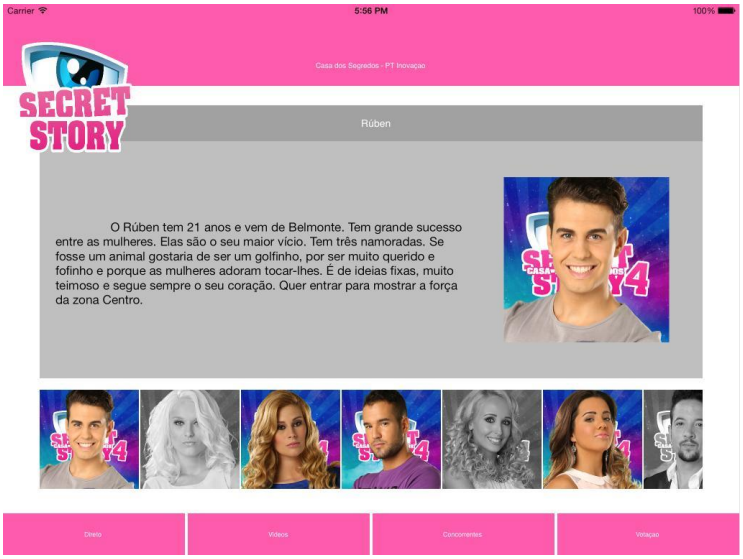


*Figura 17 – Perfis dos Concorrentes*

A aplicação foi desenvolvida inicialmente para a plataforma iOs porém, numa fase posterior, foi convertida e testada na plataforma *Android*, pelo que os resultados de ambos os testes foram diferentes em vários aspetos.

Em iOs, não ocorreram erros na apresentação do conteúdo multimédia pois o *player* da aplicação foi capaz de correr eficazmente vídeos em HLS e com o formato MPEG-4. No entanto, algumas APIs utilizadas em iOS não foram suportadas em *Android*, pelo que a conversão das aplicações de teste do primeiro para o segundo não foi direta. Assim, foram realizados diversos ajustes sem ter sido necessário recorrer a programação nativa para reproduzir os vídeos em direto. Após os testes, conclui-se que a aplicação desenvolvida adaptou-se de forma positiva à interface em iOS e a todos os requisitos do sistema, apresentando uma grande fluidez e qualidade de apresentação.

Quanto ao funcionamento em *Android*, tal como aconteceu com o *PhoneGap*, foi possível constatar que as versões mais antigas do *Android* não possuem suporte para HLS, pelo que foi necessário utilizar o protocolo RTSP para realizar a reprodução das *live streams*. Durante os testes, surgiu um obstáculo relacionado com a apresentação das imagens dos concorrentes e dos *thumbnails* dos vídeos, pois estas deixaram de aparecer da forma desejada devido ao seu tamanho elevado. Como o carregamento das imagens ocorre em simultâneo, o sistema *Android* apresentou algumas dificuldades em apresentá-las e, por isso, foi necessário criar um serviço externo para as redimensionar. No entanto, os resultados não foram os expectáveis, já que o *player* de vídeo não apresentou o controlador para colocar o conteúdo em *full screen*.

Após a utilização do *Appcelerator Titanium* conclui-se que o seu IDE é bastante complexo e de difícil adaptação, já que a sua instalação e configuração inicial apresentam um nível razoável de dificuldade. Porém, esta *framework* baseia-se na utilização de *JavaScript* como “primeira camada” que pode evitar atrasos na leitura e interpretação de HTML e aumentar o seu nível do desempenho. No entanto, o seu nível de complexidade é superior, bem como a necessidade de adaptação e o tempo de desenvolvimento em contexto inicial.

### 4.1.4. Conclusões da Unidade de Desenvolvimento

As *frameworks* de desenvolvimento multiplataforma móvel surgiram como resposta à necessidade das empresas de criar aplicações para mais do que um sistema operativo de forma a diminuir o tempo de desenvolvimento e, portanto, o seu custo.

Depois das aplicações-exemplo terem sido desenvolvidas, realizaram-se comparações em relação ao desempenho e às funcionalidades, o que permitiu uma melhor compreensão das vantagens ou desvantagens de cada uma das *frameworks*. A análise de desempenho baseou-se na interpretação da memória RAM consumida por cada aplicação. Foram tidos em conta dois critérios de análise: a memória partilhada, correspondente à utilizada pelos vários processos do sistema operativo; e a memória dedicada, memória usada especificamente pela própria aplicação. Os resultados desta análise são apresentados na seguinte tabela:

| **Framework** | **Memória Partilhada (Kb)** | **Memória Dedicada (Kb)** |
|---|---|---|
| *PhoneGap* | 12091 | 6036 |
| *PhoneGap* + JQuery *Mobile* | 14730 | 9424 |
| *Appcelerator Titanium* | 17500 | 8676 |

*Tabela 4 - Análise de desempenho através da interpretação da memória RAM*

Segundo a análise do desempenho, o *PhoneGap* é menos exigente em termos de recursos do sistema mas, segundo LEMOS (2013:29), não se aproxima do desempenho apresentado por aplicações nativas.

Em termos de reprodução de vídeo, o *Appcelerator Titanium* apresenta uma vantagem face ao *PhoneGap* uma vez que esta *framework* converte o código produzido para código nativo, promovendo a utilização do *player* de vídeos nativo. Outro aspeto neste âmbito é a impossibilidade de visualização de vídeos com DRM através de ambas as plataformas. Além disto, ambas apresentam um conjunto de APIs que permitem utilizar a maioria das funcionalidades nativas dos sistemas operativos móveis.

A utilização das multiplataformas de desenvolvimento pressupõe que os programadores se encontram dependentes de terceiros e que poderão não ter um acesso imediato às novas funcionalidades que, por exemplo, a *Google* e a *Apple* disponibilizam.

Concluindo, em aplicações que exigem um processamento mais elevado, que necessitem de APIs e/ou *plugins* específicos, deve-se ponderar a utilização de uma solução multiplataforma, pois esta poderá não ser solução ideal.

## 4.2. RECOLHA AUTOMATIZADA DE DADOS ESTATÍSTICOS

No ciclo de vida de uma aplicação móvel é importante conhecer a opinião e preferências dos utilizadores pelo que o *Google Analytics*[22], como ferramenta de angariação automatizada de dados neste âmbito, assume um papel fundamental.

O *Google Analytics* é uma ferramenta que permite monitorizar os hábitos de utilização de uma aplicação por parte dos utilizadores finais, providenciando informações sobre: visitantes, qual a sua nacionalidade, quantas vezes utilizam a aplicação, entre outros; origem do acesso à aplicação, ou seja, como é que as pessoas começaram a utilizá-la; conteúdo, de forma a conhecer quais as interfaces mais utilizadas; objetivos, que distingue quais as ações mais realizadas pelos utilizadores; e, por fim, *e-commerce*, que fornece dados sobre as possíveis compras e transações realizadas através da aplicação (Peters, 2011).

Todas as informações obtidas através desta ferramenta encontram-se sob a forma de gráficos nos quais é possível alterar os parâmetros de avaliação dos dados, como o intervalo de tempo, o tipo de dados que serão alvo de avaliação, comparação de diversos tipos de dados, entre outros (Peters, 2011).

A introdução desta tecnologia na aplicação MEO GO, desenvolvida na PTIS, foi realizada com o intuito de compreender os modelos de interação dos clientes para otimizar a aplicação e, assim, melhorar a experiência de utilização dos mesmos. Para tal, foram registadas as interações pretendidas, através da programação dos métodos necessários, e especificado o intervalo de tempo em que se pretendia enviar para a *Google* os dados reunidos. Assim, o *Google Analytics* associou todas as interações e eventos numa determinada conta individual através de um *Tracking-ID* facultado a cada utilizador na sua plataforma *web* e no qual é possível consultar todas as informações estatísticas relacionadas com a utilização da aplicação por parte dos utilizadores.

Na Figura 18 possível visualizar a interface *online* do *Google Analytics* que apresenta, no lado esquerdo, as opções de visualização onde é possível selecionar a funcionalidade pretendida.

[22] *Google Analytics*: http://www.google.com/analytics/

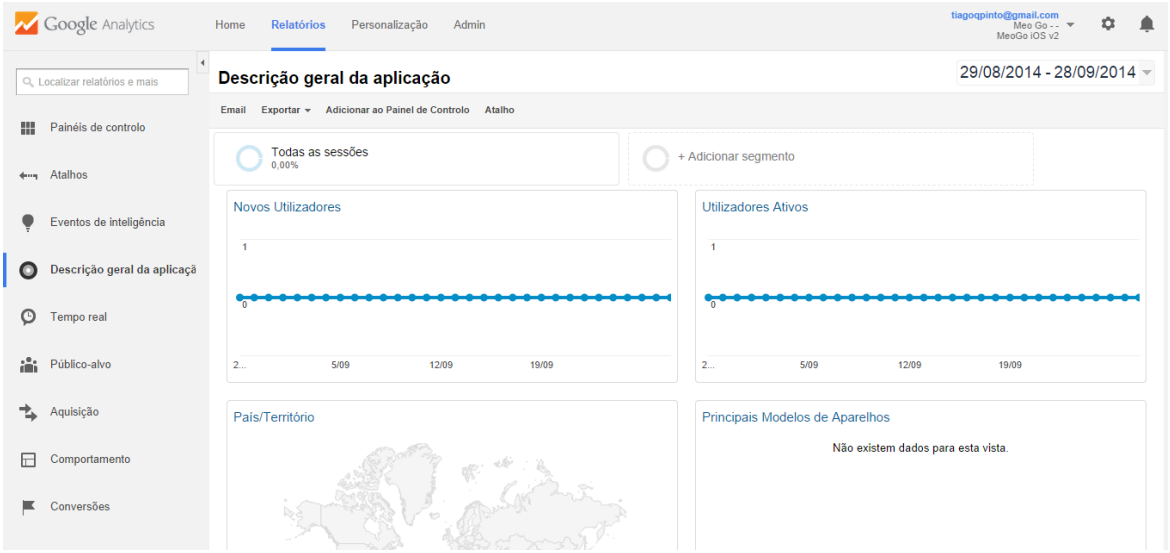


*Figura 18 - Interface online do Google Analytics (Fonte: Google, 2014)*

### 4.2.1. Análise dos Dados

No portal do *Google Analytics*, existem diversos tópicos ligados à análise dos dados estatísticos resultantes do uso de uma aplicação por parte dos seus utilizadores. Assim, o "Público-Alvo" permite conhecer a região onde a aplicação é mais utilizada a nível nacional e conhecer os países em que apresenta uma maior prevalência de utilização. No entanto, são os tópicos "Tempo Real" e "Comportamento" que foram alvo de exploração.

O "Tempo Real" deve ser a opção selecionada pelo programador se o seu objetivo for a observação do uso da aplicação em tempo real ou conhecer de que forma é realizada a navegação entre as interfaces da aplicação pelos utilizadores. Contudo, se a finalidade é a análise dos dados recolhidos durante um dado intervalo de tempo, deve ser selecionada a opção "Comportamento", que também permite examinar concretamente as funcionalidades da aplicação que o cliente utilizou.

#### "Tempo Real"

São diversas as opções que se podem selecionar dentro desta funcionalidade, como é possível observar na Figura 19, nomeadamente: Descrição Geral, Localizações, Ecrãs, Eventos e Conversões.

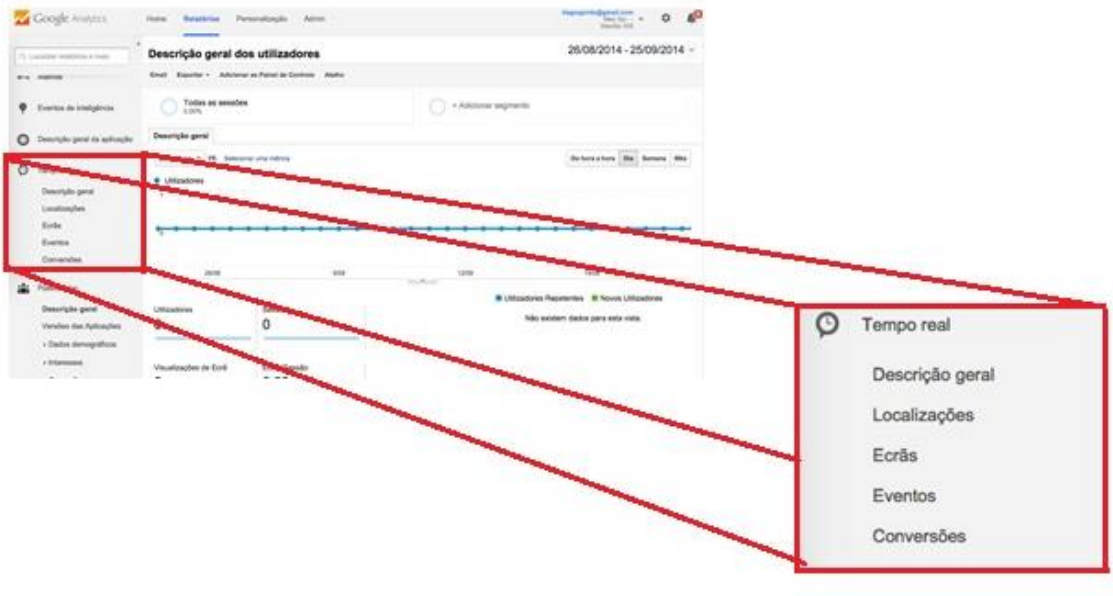


*Figura 19 – Menu "Tempo Real" (Fonte: Google, 2014)*

As opções descritas permitem compreender de que forma a aplicação é utilizada no momento exato em que está a ser analisada pelo *Google Analytics*, ou seja, pode ser conhecido o número de utilizadores em atividade, os eventos despoletados e quais os ecrãs da aplicação que foram selecionados. Ademais, é possível constatar quais as versões da aplicação que se encontram ativas e as localizações geográficas onde é acedida.

### "Comportamento"

Existem sete opções que o programador pode selecionar no âmbito do menu "Comportamento", mais concretamente: Descrição Geral, Ecrãs, Fluxo de Comportamentos, Falhas de Sistema e Exceções, Velocidade da Aplicação, Eventos e Experiências. Todos estes aspetos encontram-se ilustrados pela Figura 20:

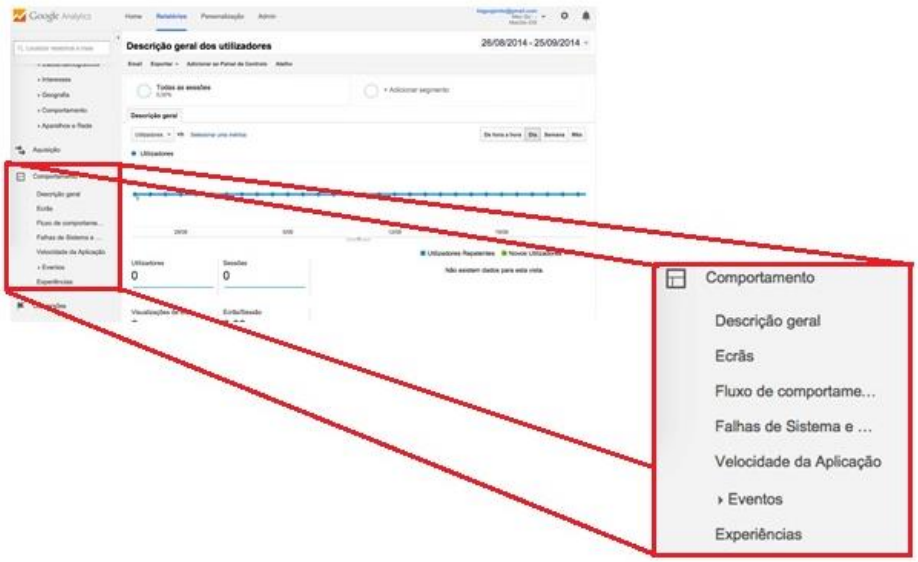


*Figura 20 - Menu "Comportamento" (Fonte: Google, 2014)*

O “Comportamento” agrega todos os eventos e ecrãs visualizados ao longo do tempo em que a aplicação se encontra operacional. Com os “Ecrãs” é possível compreender quais os menus mais utilizados e, eventualmente, se algum é dispensável para a aplicação. Para além disto, a recolha de ecrãs dá origem a outra funcionalidade deste tópico, o “Fluxo de Comportamento”. Neste elemento, é possível compreender qual o fluxo mais natural e comum que os clientes da aplicação percorrem caracterizando o tipo de navegação que exercem. Por exemplo, é possível analisar rapidamente se os atalhos do menu inferior do MEO GO são preteridos em relação ao regresso ao menu inicial.

Existe também a possibilidade observar os eventos recolhidos pela ferramenta que foram previamente definidos pelo programador, pelo que são cinco os aspetos que ele pode avaliar:

- Ecrã: é a interface, ou menu, da qual surgiu o evento;
- Categoria: o grupo ao qual o evento pertence;
- Ação: o nome atribuído ao evento;
- *Label*: valor ou conteúdo de um determinado evento;
- Variáveis de Segunda Dimensão (*Custom Dimensions*): toda a informação considerada importante que faz parte de um evento mas que não foi possível enviar na *Label*. Esta informação é opcional já que nem todos os eventos têm este tipo de variáveis. Cada uma destas variáveis apresenta um nome e tem de ser definida previamente na parte administrativa do *Google Analytics*.

### 4.2.2. Implementação no MEO GO

Esta tarefa foi desenvolvida com base no trabalho já desenvolvido anteriormente na versão do MEO GO *Android* e seguiu uma tabela descritiva. Assim, foram especificadas todas as interações possíveis e necessárias a atribuir a um evento específico e que obedecessem a necessidades reais de cálculo estatístico e análise do produto.

Na introdução da ferramenta *Google Analytics* na aplicação MEO GO iOS foram efetuados diversos envios de dados de processos de utilização em intervalos regulares através de agendamento. Para executar o envio dos dados que expunham as interações, foi implementado um método que recebe as *strings* parametrizadas e armazena todos os eventos acionados. Desta forma, foram salvaguardados os dados e o seu envio só se processa em intervalos delineados, como de 30 em 30 segundos. Na Figura 21 é mostrado um exemplo no qual o pressionar de um botão inicia a chamada de um método em que é adicionada uma ação do *GoogleAnalytics* à informação agendada para envio:

```
-(void)logButtonPress:(UIButton *)button{

     id<GAITracker> tracker = [[GAI sharedInstance] defaultTracker];

    [tracker set:kGAIScreenName value:@"Stopwatch"];
    [tracker send:[[GAIDictionaryBuilder createEventWithCategory:@"UX"
                                                          action:@"touch"
                                                           label:[button.titleLabel text]
                                                           value:nil] build]];
    [tracker set:kGAIScreenName value:nil];
}
```

*Figura 21 - Exemplo de envio de uma ação para o Google Analytics (Fonte: Sherwood, 2013)*

Na Figura 22 é possível observar parcialmente a classe de *strings* onde se encontram especificadas as ações e os respetivos identificativos para enviar para o *Google Analytics*. Cada identificativo de uma ação a enviar é composto por uma conjugação de elementos na aplicação, como “VOD_Variable_Analytics_TwitterShare”, em que se pretende mostrar que é realizado uma publicação no *Twitter,* por parte do utilizador, com o intuito de partilhar um filme (categoria VOD). Através da análise da Figura 22 é possível concluir que esta classe promove uma organização otimizada do código--fonte bem como evita a presença de *strings* definidas no código lógico:

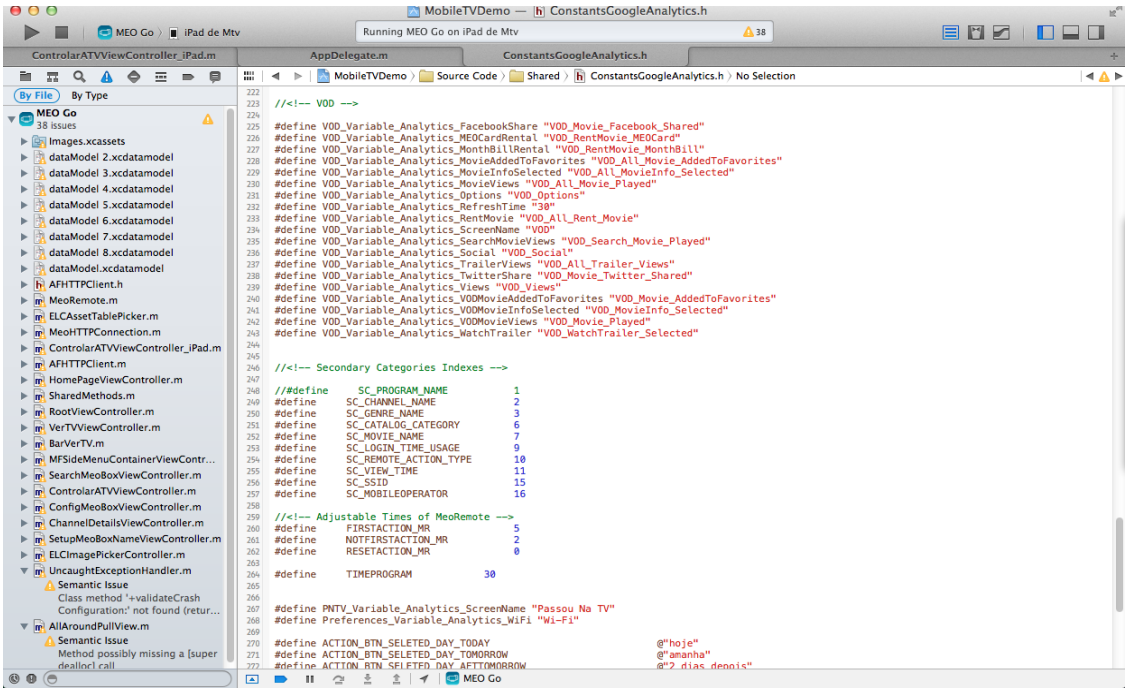


*Figura 22 - Classe de strings para envio para o Google Analytics*

O evento de envio de dados para o *Google Analytics* é chamado quando o utilizador realiza alguma ação específica. Estes eventos são especificados e implementados pelo programador no código da aplicação, exemplificado pela Figura 23, em que se observa a interface do MEO GO e modo de funcionamento da ferramenta em causa:


*Figura 23 - Envio de dados do MEO GO para o Google Analytics (Fonte: Apple Inc., 2014f)*

Quando um utilizador da aplicação clica num filme, como ilustrado pela seta na Figura 23 é acionado o método consequente e é chamada a função de envio do *Google Analytics* que indica que determinado utilizador clicou num determinado filme, de um determinado género (ação, comédia, suspense, etc…), dentro de uma localização geográfica específica, e num determinado momento. Estas ações são, ainda, explicitadas através do diagrama da Figura 24:


*Figura 24 - Diagrama de envio de eventos para o Google Analytics*

Através da análise do diagrama anterior, é possível concluir que a partir do premir de um botão, e depois de ser chamado o seu respetivo método, o Método Genérico na Classe Genérica, é invocada a função de envio dos dados para o *Google Analytics*. Por sua vez, esta ferramenta acede ao documento de constantes onde estão especificados o *Tracking-ID* do utilizador da conta, a periodicidade de envio, e as *strings* a enviar.

## 4.3. TESTES AUTOMATIZADOS

A necessidade de controlo da qualidade da aplicação MEO GO iOS e a importância de manter a satisfação dos clientes relativamente à mesma suscitou a criação de um projeto de testes automatizados orientados à aplicação em causa.

O constante acompanhamento das particularidades da utilização de uma aplicação permite, aos programadores e à equipa de especificações e *design* de aplicações móveis, conhecer os utilizadores no aumento do seu grau de satisfação. É importante para qualquer empresa garantir a qualidade das aplicações móveis que desenvolve. Os testes automatizados simulam a interação de um utilizador normal na realização de ações predefinidas na aplicação.

De forma a realizar os testes foi desenvolvido um *script* de autenticação através do *UI Automation*[23]*,* uma ferramenta do *XCode*, que especifica as interações entre elementos das aplicações iOS, simulando o uso da mesma por um utilizador real. Para além disto, esta ferramenta permite guardar *scripts*, realizar análises contínuas de desempenho, apresentar capturas de ecrã periódicas e "gravar" a interação com um utilizador real e convertê-la para *Javascript*, oferecendo a possibilidade de modificar o mesmo.

Após o seu desenvolvimento, o *script* de autenticação foi posteriormente carregado, com o código da aplicação, para a plataforma *AppThwack*, que permite testar uma aplicação em centenas de dispositivos móveis de forma rápida e automatizada. A utilização conjunta do *UIAutomation* e do *AppThwack* permitiu, assim, realizar testes automatizados e personalizados numa vasta gama de dispositivos evitando encargos financeiros para aquisição de equipamentos.

O *script* referido foi desenvolvido em *Javascript* para simular a interação de um utilizador com a interface da aplicação MEO GO. Para tal, foi necessário apenas aplicar "*accessibility labels*" a cada elemento interativo pretendido e especificar o tipo de interação que se pretendia executar com o referido elemento. Um processo básico de diagnóstico de elementos e obtenção de todos os elementos visíveis da aplicação é o "logElementTree()", que se encontra representado na Figura 25 :

[23] UI *Automation:* https://developer.apple.com/library/ios/documentation/DeveloperTools/Conceptual/InstrumentsUserGuide/UsingtheAutomationInstrument/UsingtheAutomationInstrument.html

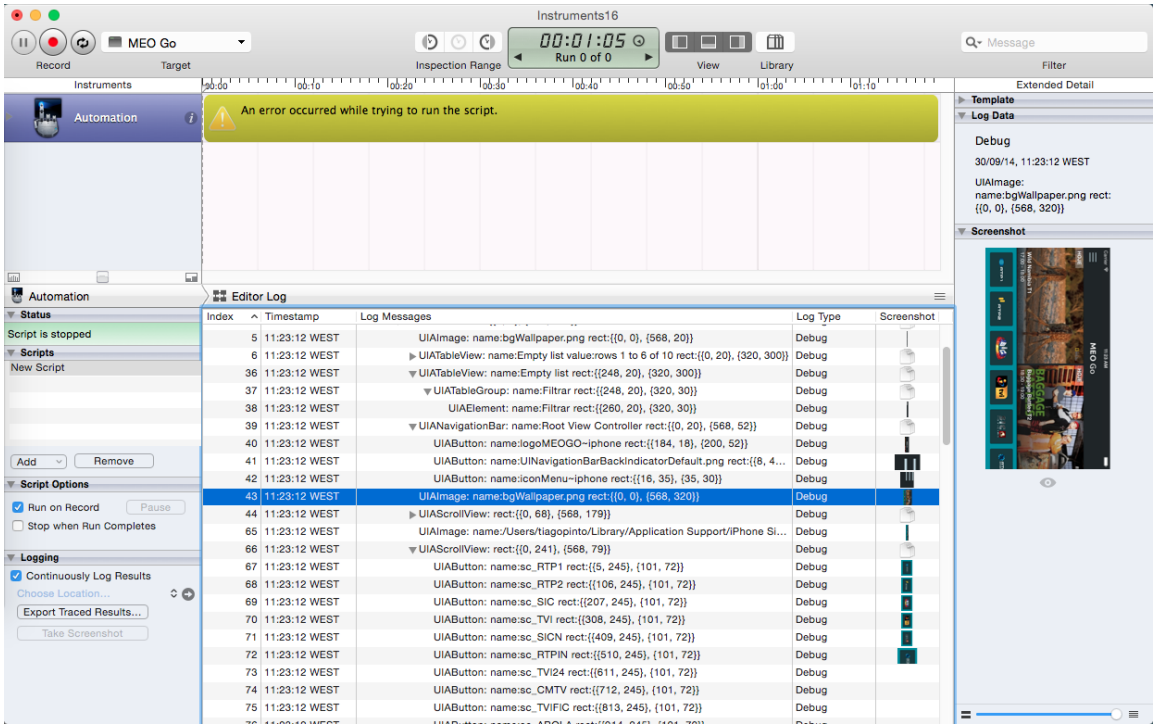


*Figura 25 – Visualização de todos os elementos visíveis no ecrã mostrado na aplicação utilizando o comando " logElementTree()".*

Nesta figura, é possível observar uma hierarquia (uma "UIAScrollView" com vários "UIAButton", por exemplo) de elementos gráficos da aplicação MEO G. Esses elementos encontram-se identificados individualmente com uma "*accessibility label*" no código de programação da aplicação, que se traduz campo "*name*:" distinguido pela ferramenta UI *Automation* do *XCode*.

### 4.3.1. Requisitos do *script*

Com a necessidade da implementação de testes automatizados foi necessário esquematizar sequencialmente um fluxo de interação da aplicação do MEO GO a ser testada, de forma a simular a interação entre a mesma e um utilizador real. Assim, foi criado um *script* que, organizado de forma sequencial, permitiu simular uma interação física com o dispositivo, possibilitando a realização de todos os movimentos naturais da aplicação.

O *script* foi implementado para *iPhone* e *iPad*, já que ambos realizam processos distintos de interação com o utilizador. Um aspeto vital para a avaliação da aplicação foi o registo dos processos através de *PrintScreens* pois, apesar de o UI *Automation* produzir relatórios contínuos, surgiu a necessidade de compreender quais os *PopUps* que apareceram em caso de um eventual erro.

Para o desenvolvimento deste *script* foram previamente estabelecidos requisitos que lhe atribuíram a especificidade necessária para cumprir os objetivos da implementação dos testes automatizados:

- Ultrapassar o primeiro ecrã de ajuda, realizando um clique;
- Selecionar um canal de televisão para assistir (por exemplo RTP1);
- Realizar o *Login*, introduzindo as credenciais e realizando o clique de submissão;
- Detetar o início da emissão do stream de vídeo do canal de televisão e captar *PrintScreens*;
- Seleção de outro canal enquanto se está a assistir ao atual;
- Retrocesso para o menu principal;
- Acesso à categoria “VideoClube”, clicando no devido botão;
- Escolha de um filme pré-definido;
- Clique no botão “Ver *Trailer*”;
- *PrintScreen* da reprodução vídeo do trailer;
- Em caso de erro em qualquer etapa do processo é retirado um *PrintScreen.*

### 4.3.2. AppThwack

Após o desenvolvimento do *script* e consequentes testes nos dispositivos móveis disponíveis no departamento de MTV na PT Inovação e Sistemas, surgiu a necessidade de testar o MEO GO iOS no maior número de dispositivos possível. Sendo assim, foi selecionada a ferramenta do *AppThwack*[24], uma plataforma *online* capaz de receber o projeto da aplicação e o seu *script* para realizar testes automatizados em centenas de dispositivos móveis distintos existentes numa *cloud*.

[24] *AppThwack*: https://appthwack.com/

Apesar de não existir uma grande variedade de dispositivos móveis que contenham iOS, o *AppThwack* conseguiu realizar os testes em todos os que efetivamente existiam na sua plataforma. A utilização de uma ferramenta como esta é uma mais-valia para as empresas, no sentido em que evita gastos extraordinários em terminais móveis para testar as suas aplicações. O AppThwack tem, no caso do iOS todos os dispositivos e versões de sistemas operativos disponíveis para teste.

No decorrer de um teste executado em *AppThwack* é possível analisar todo o *log* dos comandos executados, bem como o *timestamp* que define o momento da sua execução. Alem disto, é possível obter *PrintScreens* predefinidos no *script* que permitem a apresentação sequencial e a identificação temporal do comando executado. A Figura 26 retrata este aspeto, já que ilustra um exemplo de simulação da autenticação da aplicação MEO GO entre *logs* periódicos:

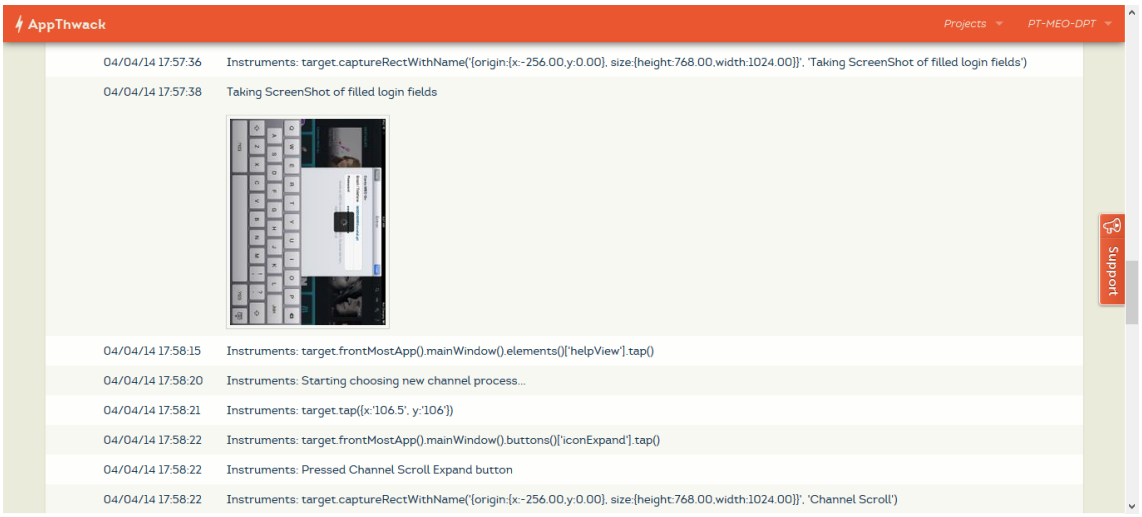


*Figura 26 - Teste da aplicação MEO GO com AppThwack*

Estes testes são realizados de forma individual em cada dispositivo disponível e é possível consultar o *Log* da execução de cada um. Os testes são executados em dispositivos idênticos mas com sistemas operativos distintos de forma a possibilitar um *debug* correto e específico.

Na Figura 27 é possível observar, de forma parcial, o relatório final de um teste executado com MEO GO iOs do *script* desenvolvido. Nessa mesma figura destaca-se a parte superior onde estão localizados os resultados gerais dos testes em que se observa quantos foram realizados correta ou incorretamente e quais os erros que surgiram durante os mesmos:

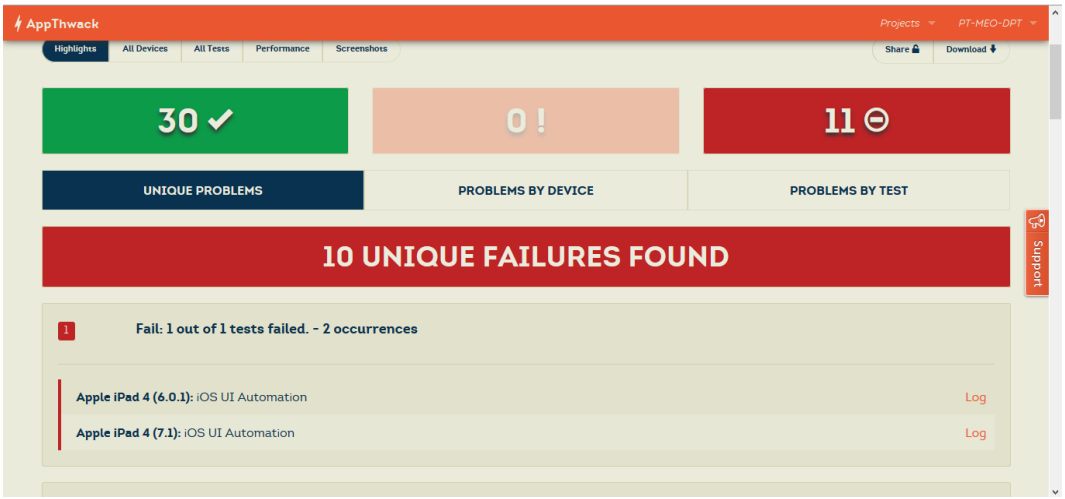


*Figura 27 - Relatório final do teste do MEO GO em AppThwack*

O *AppThwack* disponibiliza, ainda, outra funcionalidade: a análise da *performance* da aplicação. Os resultados dessa análise são organizados em diagramas no decorrer dos testes, como demonstra a Figura 28:

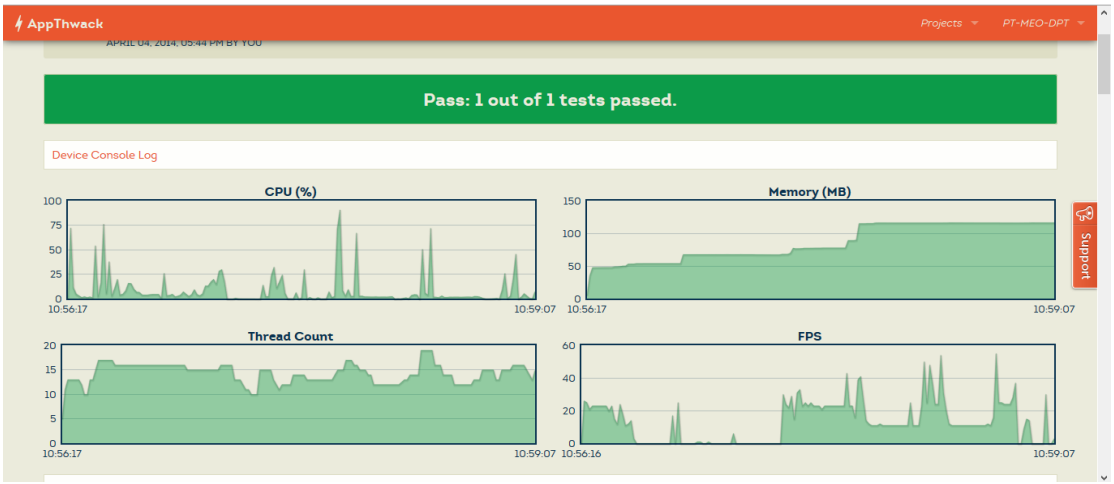


*Figura 28 - Análise da performance da aplicação MEO GO em AppThwack*

Como se pode constatar, a avaliação da *performance* consiste na análise do processamento do CPU, da memória utilizada, do número de *threads* e os *frames* por segundo da aplicação. Estas informações para compreender o desempenho atual da aplicação e, em caso de problemas de *performance*, reconhecer a origem do erro e atuar em conformidade.

## 4.4. DEMONSTRAÇÃO SOBRE SWIFT

Um dos projetos desenvolvidos em estágio remeteu-se ao estudo do *Swift*, a recente linguagem de programação lançada pela *Apple* lançada e planeada para substituir a atual e predominante linguagem em iOS, o *Objective-C.* Para demonstrar as potencialidades desta nova linguagem foi criada uma aplicação-exemplo que será alvo de análise posteriormente.

No presente ano, a *Apple* apresentou o *Swift*, cuja prioridade estabelecida para o seu desenvolvimento foi a melhoria do desempenho, pelo que esta linguagem é interpretada pelo compilador LLVM e transformada em código nativo otimizado de forma a obter a máxima *performance* dos dispositivos móveis da *Apple*. Para além disto, a segurança e o *design* foram outras prioridades na criação do *Swift*, que introduziu também o conceito de *playgrounds*. O lançamento oficial foi realizado através da versão "*Gold Master*" do sistema operativo da *Apple*, o *Yosemite*[25], após um longo período de testes e avaliação. Apesar das diferenças que apresenta em relação ao *Objective-C*, o *Swift* pode interagir com essa linguagem e o programador pode realizar uma coordenação entre classes de ambas as linguagens, tal como foi realizado no projeto desenvolvido em causa (Apple Inc., 2014c).

### 4.4.1. "FlappyPT"

Após a investigação realizada acerca do *Swift* foi desenvolvida uma aplicação exemplificativa, o "FlappyPT", para demonstrar quais as funcionalidades e limitações desta linguagem. O "*FlappyBird*", um jogo de sucesso da atualidade, foi a aplicação em que o "FlappyPT" foi baseado, cuja finalidade é o controlo de um pássaro que se desloca horizontalmente de forma constante e que o utilizador deve mobilizar verticalmente para evitar os obstáculos que surgem no seu percurso.

Esta aplicação foi inspirada no Campeonato do Mundo de Futebol no Brasil em 2014, em que ocorreu a derrota inédita da Seleção brasileira por 7-1 com a congénere alemã. Assim, em vez de controlar um pássaro, o utilizador faz deslocar um jogador da equipa brasileira que tentar ultrapassar os jogadores da Alemanha que surgem no seu caminho, aumentando o *score* à medida que consegue ultrapassar os obstáculos.

[25] *Yosemite:* https://www.apple.com/pt/osx/

A Figura 29 ilustra dois *screenshots* do “FlappyPT”, onde é possível observar que o jogador brasileiro, de camisola amarela, tenta percorrer o caminho livre entre os jogadores alemães, de camisola branca:

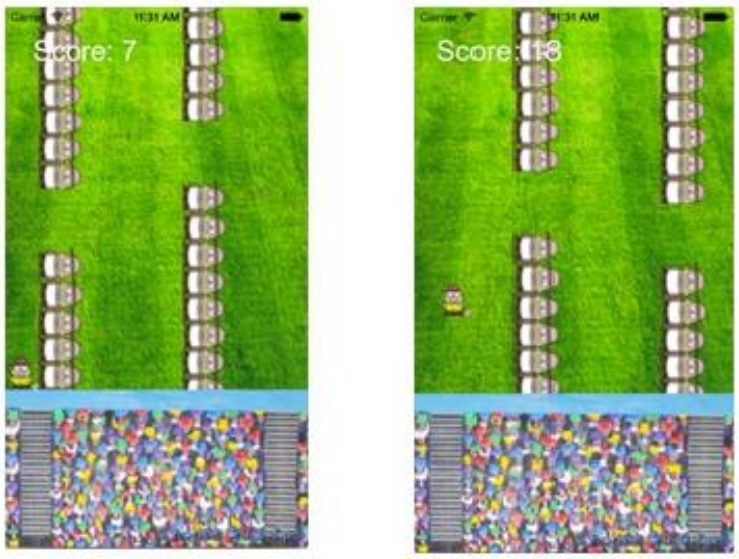


*Figura 29 - Screenshots do "FlappyPT"*

Este jogo foi desenvolvido em *SpriteKit*, uma biblioteca iOS compatível com *Objective-C* e *Swift* e que faculta um ambiente de gestão e apresentação de animações e formas visuais que poderão ser usadas em aplicações ou jogos. Esta biblioteca otimiza o desempenho do *hardware* dos dispositivos em relação ao processamento e desempenho gráfico, possibilitando o suporte de componentes gráficos como *Sprites*, ou retângulos com ou sem texturas, elementos textuais, vídeos e formas geométricas aleatórias especificadas por coordenadas (Apple Inc., 2014e).

Sendo um dos objetivos do desenvolvimento desta aplicação compreender a interação existente entre código de programação em *Objective-C* e *Swift*, retiraram-se as seguintes ilações:

- Para utilizar métodos públicos ou variáveis públicas numa classe *Objective-C* a partir de uma classe *Swift* é apenas necessário incluir nas primeiras o *header* do projeto. Tal ação pode ser exemplificada pelo seguinte exemplo de código: “#import “FlappyPT-Swift.h”;

- No sentido inverso, para aceder a métodos ou variáveis públicas de uma classe em *Swift* a partir de uma classe em *Objective-C*, é necessário criar um ficheiro *header* especifico com o nome de "*bridging-header*", onde se deverão importar os *headers* de todas as classes *Objective-C* a que se pretendam aceder. A criação deste *header* é apoiado pelo *XCode* que torna este processo mais intuitivo.

Na Figura 30 encontra-se o diagrama de classes do "FlappyPT" que demonstra a estrutura do jogo e qual o processo de desenvolvimento do mesmo em iOS:

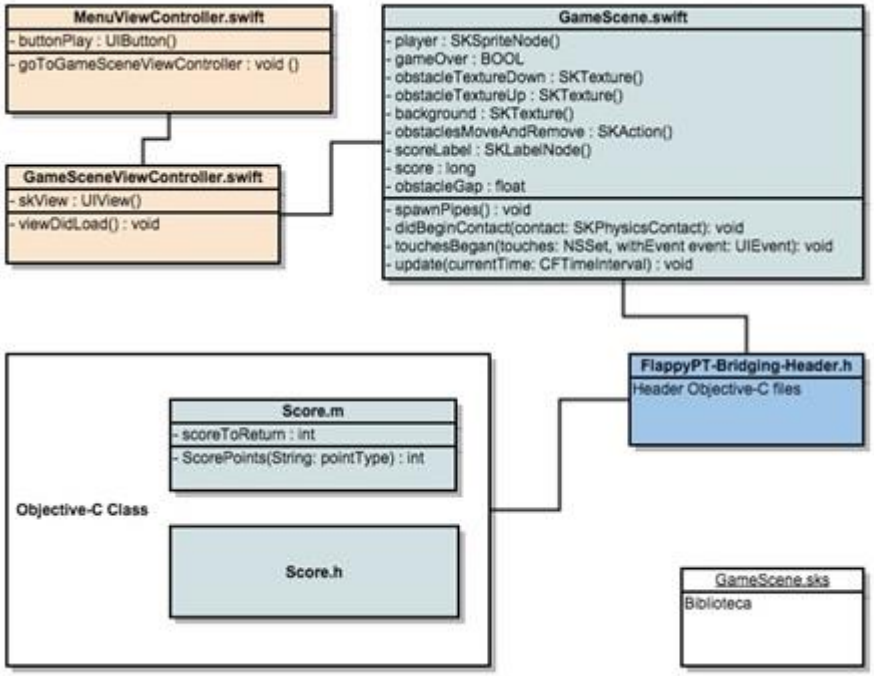


*Figura 30 - Diagrama de classes do "FlappyPT"*

Como é possível constatar através da análise do diagrama, a aplicação recorre apenas a dois "*ViewControllers*", sendo que um corresponde ao ecrã de jogo em si e o outro ao ecrã de menu que é mostrado no inicio da aplicação e sempre que o utilizador perde. O "*MenuViewController*" possui apenas um botão na sua interface e um método que redireciona o utilizador para o jogo. O "*GameSceneViewController*" inicializa a classe "*GameScene (UIView)*", que é onde o jogo é mostrado bem como onde são interpretadas todas as interações por parte do utilizador.

No diagrama é possível visualizar, igualmente, a interação entre *Objective-C* e *Swift*, pelo que em relação à primeira linguagem existe uma componente composta apenas pela classe "Score.m" e pelo respetivo *header*, que se destina unicamente a incrementar o valor do *score* do utilizador à medida que consegue ultrapassar os obstáculos. Esta classe possui ainda um método acedido pela classe "*GameScene.swift*" que atualiza o valor da pontuação conforme o pretendido.

Na Figura 31 é possível perceber como é realizado o acesso a um método *Objective-C* a partir de uma classe em *Swift*: a classe "File.m" é inicializada e associada a uma constante na linha 19 da referida figura, sendo possível através dessa mesma constante o acesso aos métodos da classe *File*.

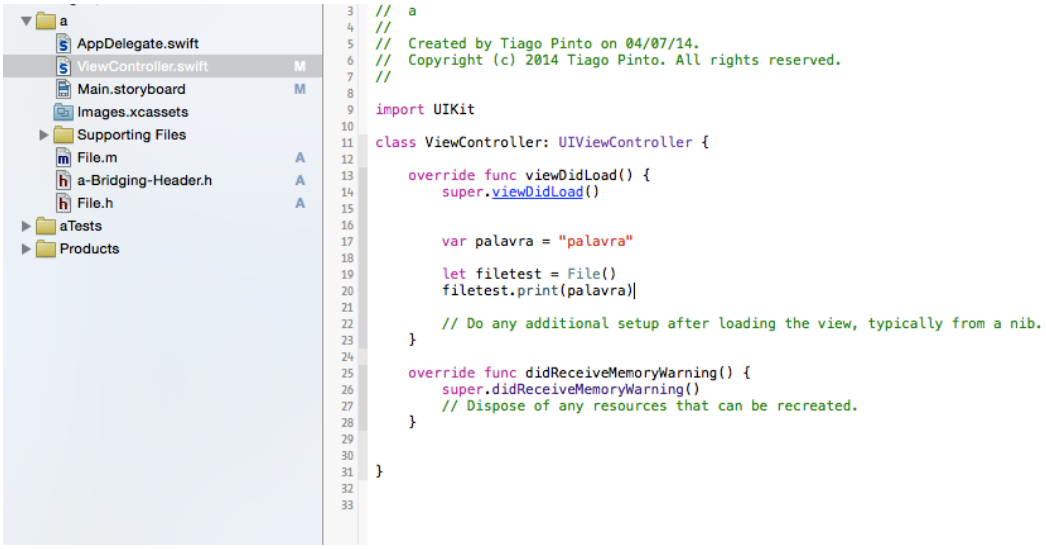


*Figura 31 - Chamada de um método Objective-C numa classe em Swift*

### 4.4.2. Conclusões

Apesar de ser possível construir uma aplicação utilizando apenas uma linguagem, *Objective-C* ou *Swift*, a demonstração das funcionalidades da última através da criação da aplicação "FlappyPT" permitiu concluir que pode existir uma interação entre as duas linguagens no mesmo projeto.

Aquando do desenvolvimento da aplicação foi possível corroborar que a sintaxe utilizada em *Swift* caracteriza-se pela sua facilidade de interpretação de código já existente. No entanto, possui um menor nível de personalização e de controlo em relação ao *Objective-C* devido, principalmente, à gestão automatizada das variáveis.

## 4.5. IDOOR

O *iDoor* é uma plataforma que permite abrir portas automáticas a partir da localização do utilizador, já que tira partido do funcionamento de pequenos dispositivos *Bluetooth*, os *iBeacons,* capazes de notificar os terminais móveis que se encontrem próximos geograficamente. Esta tecnologia permite calcular a distância entre o dispositivo móvel e o emissor facultando uma localização espacial pormenorizada.

A arquitetura do sistema *iDoor* inclui uma aplicação-cliente para instalação nos dispositivos móveis que está disponível nos sistemas operativos *Android* e iOS. Assim, esta aplicação baseia-se na deteção da distância relativa à porta e ao dispositivo móvel e o servidor que gere o acesso recebe o pedido de abertura da porta. Após a autenticação e validação do utilizador, a porta é aberta e é mostrada uma mensagem de boas-vindas num ecrã.

O *iDoor* é uma aplicação que pretender revolucionar o futuro já que permitirá, por exemplo, que um colaborador da PT Inovação e Sistemas consiga abrir todas as portas automáticas no seu local de trabalho mantendo o seu *smartphone* no bolso, desde que tenha a aplicação instalada. Desta forma, poderá ser substituído o atual método de validação do acesso através do contacto do cartão de identificação pessoal com o leitor.

### 4.5.1. iBeacons

Os *iBeacons*[26] são dispositivos que transmitem informação de forma constante através de *Bluetooth* e que se caracterizam por serem dispositivos de baixo custo monetário, com gastos energéticos eficientes e pela capacidade de notificar a sua presença a dispositivos móveis que se encontrem próximos.

Esta tecnologia foi desenvolvida pela *Apple* e destina-se, preferencialmente, à utilização nos seus dispositivos e sistemas operativos, apesar de já existirem soluções semelhantes para *Android*. Os *iBeacons* baseiam-se no novo protocolo de *Bluetooth* existente, o *Bluetooth* LE*,* que possui uma eficiência energética rigorosa. Cada dispositivo iBeacon possui um UUID (Universal Unique Identifier) específico, composto por um atributo alfanumérico de 16 bytes como, por exemplo: “D9B9EC1F-3925-43D0-80A9-1E39D4CEA95C”. Cada dispositivo possui, igualmente, um “Major” e um “Minor” representados por um valor numérico de 2 bytes.

[26] *iBeacons:* https://developer.apple.com/ibeacon/

Os *iBeacons* podem ser monitorizados regionalmente, o que possibilita que os terminais móveis consigam, até com as aplicações destinadas ao efeito em b*ackground*, detetar a entrada ou a saída das diversas regiões. Uma região é definida por uma área que rodeia o *iBeacon* com um raio limitado ao alcance do mesmo, podendo sofrer alterações consoante a existência de obstáculos físicos no ambiente.

### 4.5.2. Arquitetura do Sistema

Existem seis componentes básicos que definem o funcionamento do sistema *iDoor*. Um componente é definido pela base de dados onde foram implementadas as tabelas que especificam os utilizadores autorizados, bem como cada abertura de porta realizada. Quando um utilizador é autenticado, o serviço verifica na base de dados se o utilizador está registado e, caso não esteja, o serviço *iDoor* recorre ao serviço de "*Windows Domain*" presente na PT Inovação e Sistemas de forma a ter acesso aos dados do utilizador em causa, como é demonstrado pela Figura 32:

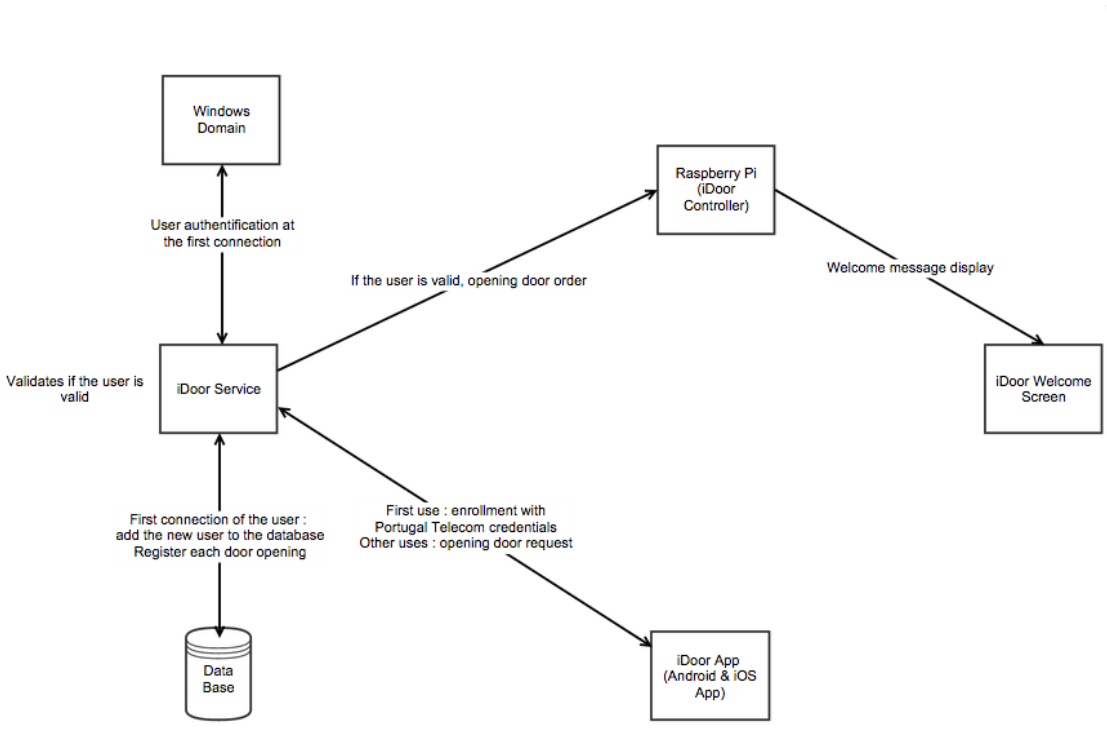


*Figura 32 - Arquitetura do iDoor*

Existem quatro interações principais com o Serviço *iDoor*, nomeadamente: o "*Enroll*", o "*Disenroll*", o "*OpenDoor*" e o "*BeaconsList*". O primeiro serviço remete-se à autenticação do utilizador que, se se encontrar válido no "*Domain*" existente na PT Inovação, será adicionado à base de dados de permissões de acessos do iDoor. Este processo de autenticação vai devolver um identificador de segurança, o "*Security Hash*", que será requisitado na execução dos restantes serviços.

O "*Disenroll*" é o reverso do "*Enroll*", pelo que remove a permissão dada ao utilizador pelo sistema. Já o serviço "*OpenDoor*", a partir da identificação da porta no URL e da identificação do utilizador e da *Hash* de segurança, envia o pedido de abertura da porta para o *Raspberry Pi* destinado ao controlo da mesma, como é demonstrado na Figura 33:

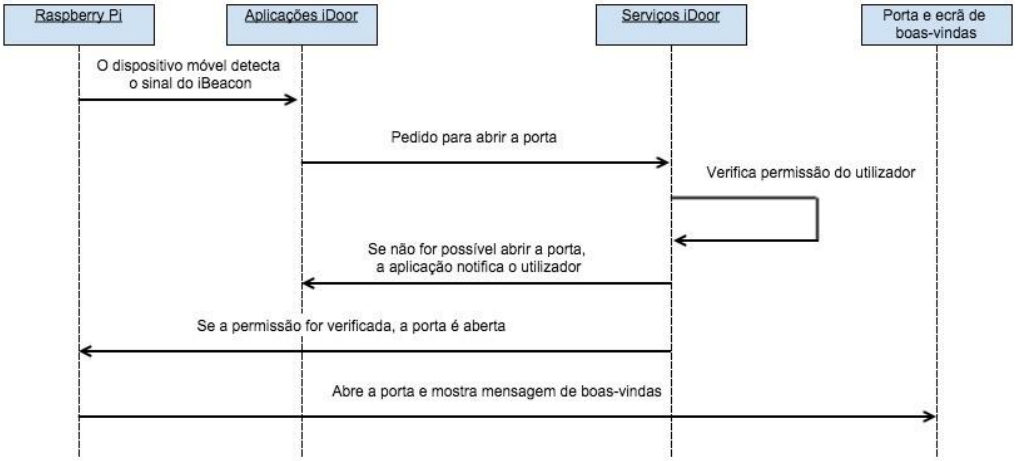


*Figura 33 - Diagrama de sequência do iDoor*

No diagrama anterior é possível constatar que a sequência do sistema é iniciada na deteção do *iBeacon* à semelhança do mecanismo de abertura da porta por parte da aplicação móvel.

Após o reconhecimento e validação dos requisitos de proximidade, ou a distância mínima, em relação à porta/*iBeacon* é feito o pedido de abertura de porta que será validado pelo serviço *iDoor*, que autentica o utilizador. Sempre que ocorre a execução do serviço é sempre remetido um XML ou JSON que informa a aplicação do sucesso ou não da operação. No entanto, o utilizador apenas é notificado quando o acesso à abertura de porta não é permitido ou quando existe algum problema nos processos de autenticação.

Após a validação pelo serviço *iDoor* e ser dada a permissão para a sua abertura, a porta é aberta e é apresentado um ecrã de boas-vindas como é visível na figura seguinte:


*Figura 34 - Ecrã de boas-vindas do iDoor*

### 4.5.3. Aplicação móvel em iOS

Como explicado anteriormente, o sistema *iDoor* conta com a interpretação da localização de *iBeacons* a partir de terminais móveis, pelo que é necessário uma aplicação que permita realizar esta operação, o objetivo principal deste projeto. Assim, a aplicação móvel para o *iDoor* foi desenvolvida em *Swift* de forma a tirar partido das vantagens que este apresenta face ao *Objective-C*.

O *iDoor* em iOS tem como finalidade a interação com o utilizador, pelo que lhe permite iniciar/terminar sessão e enviar o pedido de abertura da porta conforme a distância em relação à mesma. Na página principal da aplicação são apresentados os resultados em tempo real da busca por *iBeacons* através do terminal móvel. Isto permite ao utilizador o acompanhamento constante da distância relativa a cada *iBeacon*, em metros, bem como a consulta de informação detalhada sobre o mesmo, quando selecionado na tabela de *iBeacons*.

A aplicação foi desenvolvida para funcionar tanto em *foreground,* ou escondida, como em primeiro plano. Além disto, foi construída de forma a emitir notificações *push*, também denominadas notificações do sistema, que alertam o utilizador quando ocorre uma tentativa de abrir uma porta e se a operação foi bem-sucedida ou não. Tanto o serviço de procura constante por *iBeacons* bem como o serviço de notificações são passíveis de serem desligadas no ecrã de Definições da aplicação. Todas as interações de uso relatadas encontram-se representadas no diagrama de *Use Cases* da Figura 35:

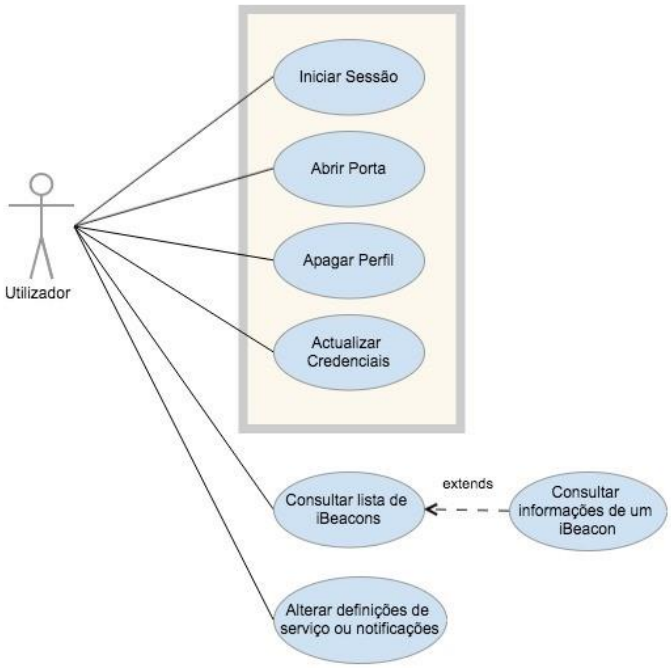


*Figura 35 - Diagrama de Use Cases do iDoor*

Nesta aplicação existem oito “ViewControllers”, classes inerentes às interfaces e respetivos elementos visuais, bem como à lógica consequente que se pretende implementar, possuindo um relacionamento direto com as páginas existentes. Existe ainda um “ViewController” que não possui aspeto gráfico e destina-se apenas à gestão do fluxo da aplicação. Assim, possui quatro ecrãs principais, ilustrados no fluxo de ecrãs da Figura 36:

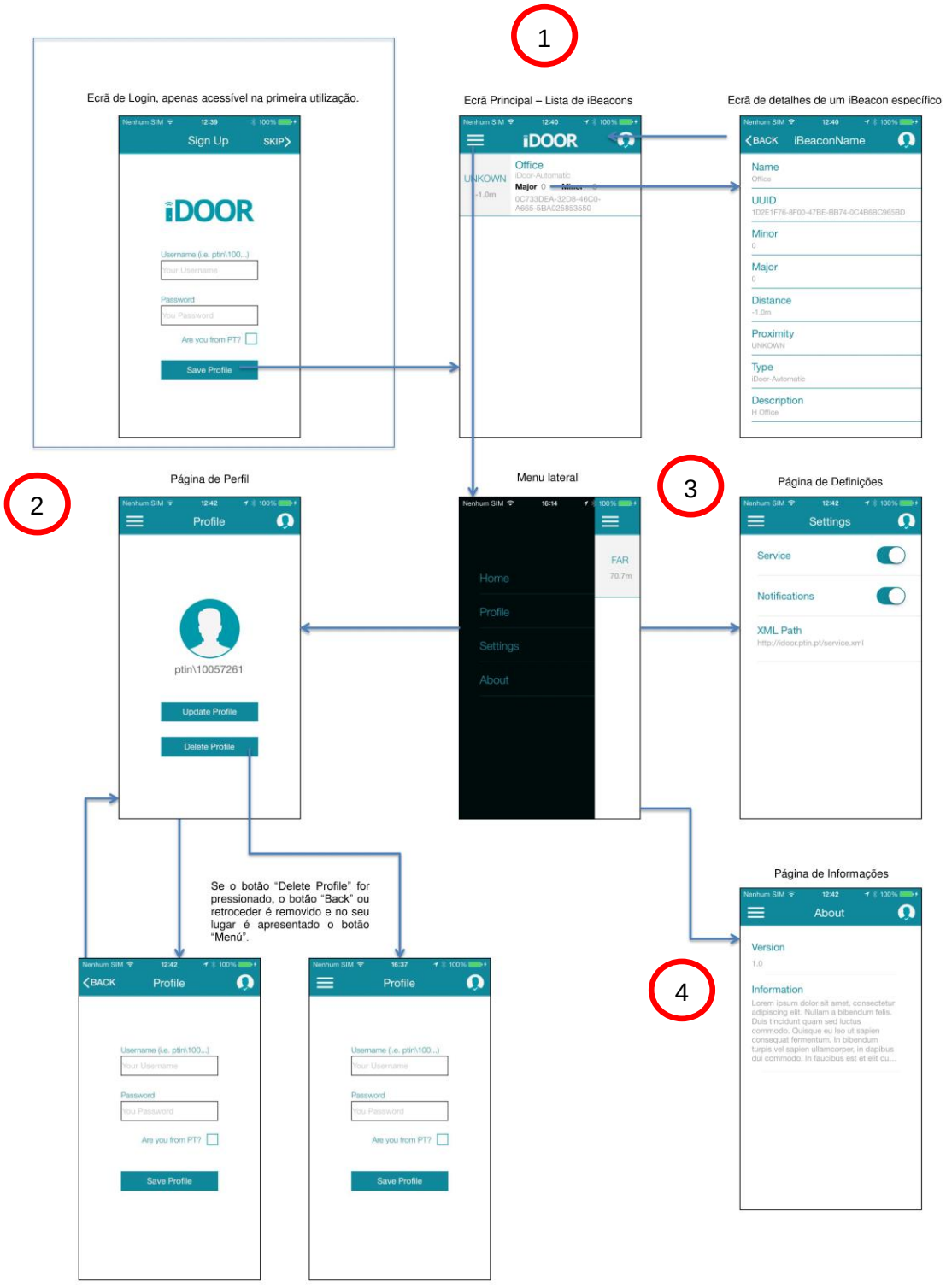


*Figura 36 - Fluxo de ecrãs do iDoor*

Como é possível compreender pela análise da figura anterior, existem quatro ecrãs principais na aplicação *iDoor*:

- Página "*Home*" (1): apresenta todos os *iBeacons* encontrados e as informações a eles associadas, nomeadamente: o UUID, o *major*, o *minor*, a distância relativa, o nome e a sua descrição. Parte desta informação é recebida no XML da listagem de *beacons* a que a aplicação acede sempre que é iniciada. Existe, igualmente, a possibilidade de o utilizador clicar num *iBeacon* presente na lista e aceder à página de informações acerca dele;
- Página "*Profile*" (2): permite que o utilizador inicie/termine sessão, caso não o tenha feito no primeiro ecrã após a instalação da aplicação. Ademais, esta página é mostrada consoante o estado atual do utilizador, pelo que se ele se encontrar autenticado, aparecerão as informações relativas ao mesmo, podendo ainda atualizar os seus dados pessoais e a eliminar o seu perfil;
- Página "Definições" (3): Nesta página o utilizador pode especificar se pretende que o serviço esteja ligado ou desligado e optar qual o estado que quer submeter às notificações *push*;
- Página "Acerca" (4): Página de informações relativas à aplicação.

Ao iniciar a aplicação, o utilizador depara-se com o ecrã de autenticação, existindo a possibilidade de iniciar sessão ou de ignorar esta etapa. Consequentemente, dado o estado do perfil do utilizador (se autenticado ou não), a aplicação adapta-se, facultando diferentes ecrãs em situações distintas, como é o caso da página "*Profile*". De qualquer forma, o utilizador da aplicação iDoor necessitará de proceder à autenticação para conseguir efetivamente abrir uma porta. Este facto é explicado pela necessidade de manutenção dos padrões de segurança necessários ao regular funcionamento do departamento de MTV da PT Inovação e Sistemas. A figura 37 ilustra estes aspetos para além de esclarecer as interações entre a aplicação móvel e o restante sistema que compõe o iDoor:

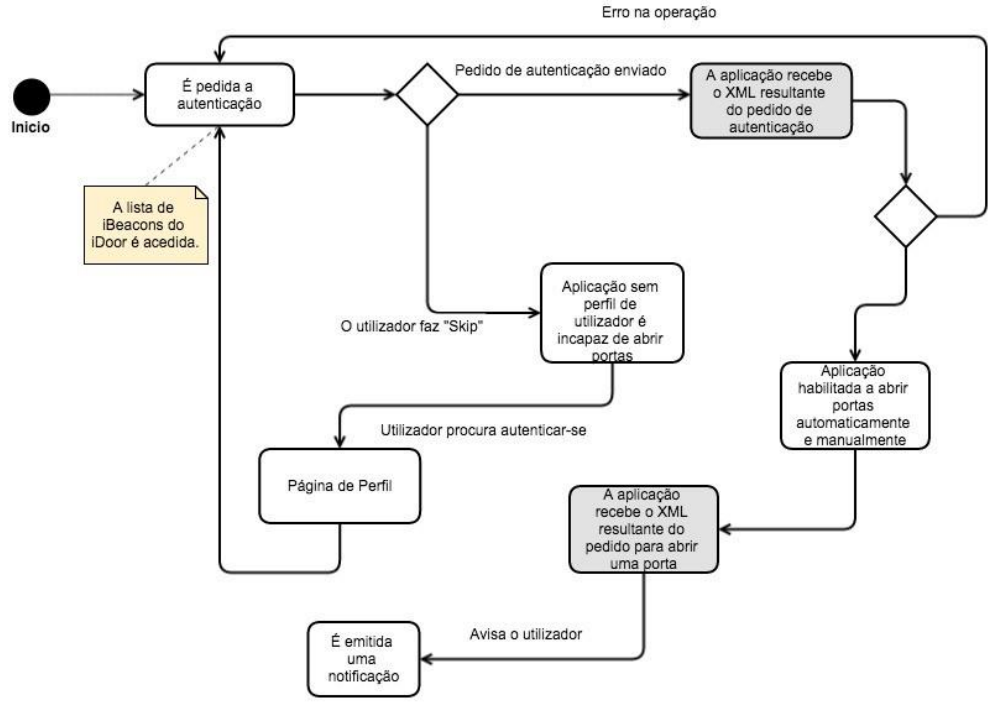


*Figura 37 – Diagrama de interações entre o serviço iDoor e a aplicação móvel*

Por fim, torna-se necessário conhecer a estrutura programática da aplicação móvel, nomeadamente as suas classes, métodos, variáveis e objetos utilizados. Para tal é apresentado, na página seguinte, um diagrama de classes (Figura 38) segmentado por cores de forma a explicitar a lógica envolvente ao código criado para a aplicação:

- A cor azul define as classes “ViewController” usadas como lógica e interface na aplicação. A classe “MainViewController” tem como objetivo coordenar e gerir o fluxo entre diferentes páginas, alterando os “ViewControlllers” invocados, consoante a interação com o utilizador;
- A cor vermelha demonstra as classes implementadas com a finalidade processar as conexões HHTP bem como do *Parse* (interpretação) consequente da informação retribuída pelo XML.
- A cor verde caracteriza as classes que formam os objetos estruturais para uso noutras classes.

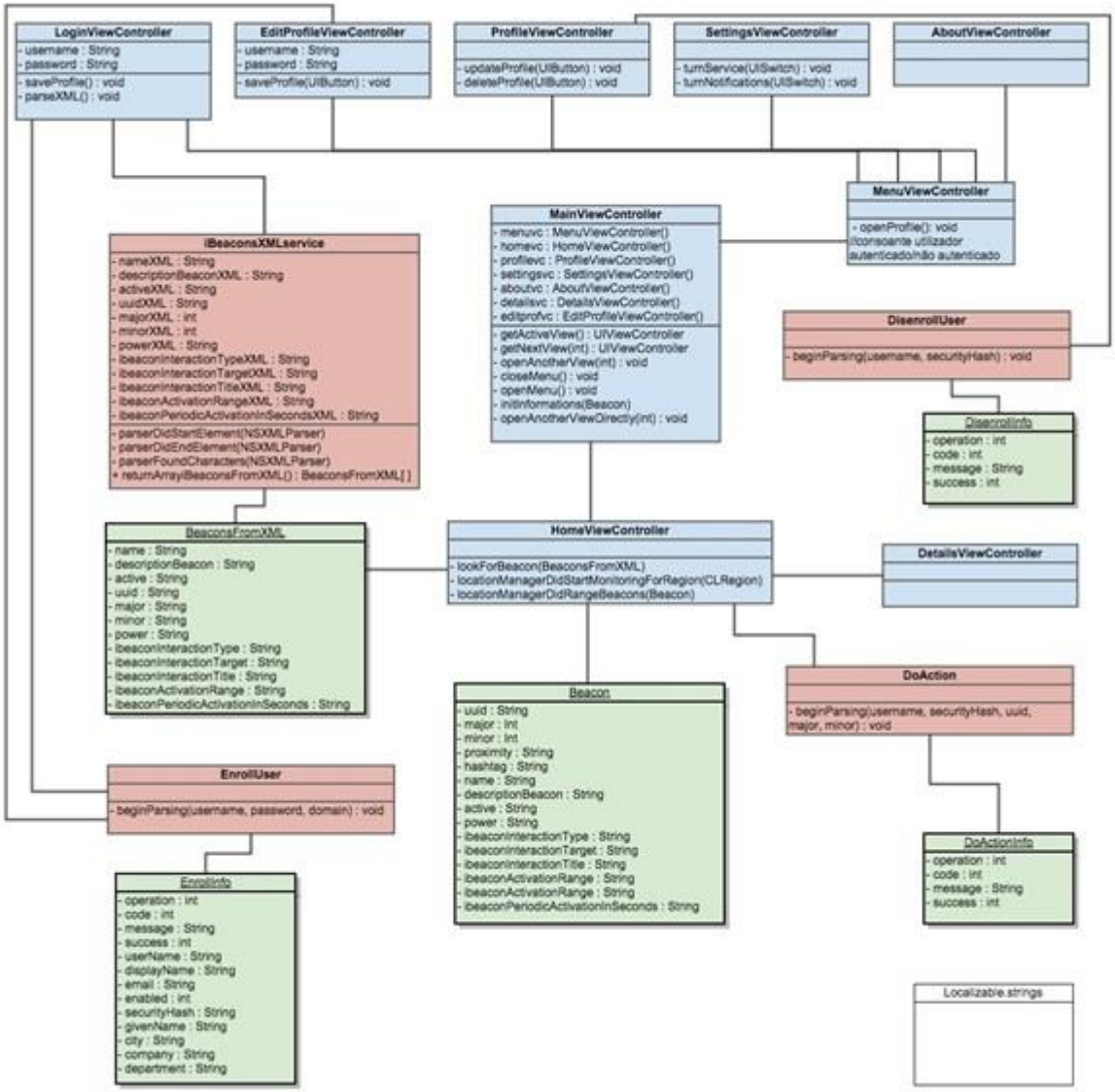


*Figura 38 – Diagrama de classes da aplicação móvel do iDoor*

# 5. CONCLUSÃO

O estágio realizado na PT Inovação e Sistemas permitiu o desenvolvimento de conhecimentos e competências técnicas nunca antes adquiridas em contexto curricular, nomeadamente em relação à programação em iOS. Através deste relatório foi possível, igualmente, conhecer as instituições que permitiram esse desenvolvimento, a PT e a PT Inovação e Sistemas, e de que forma impulsionaram o desenvolvimento tecnológico na área das comunicações em Portugal. Além disto, foi fundamental conhecer os projetos mais inovadores da PT, nomeadamente o SAPO e o MEO.

Para a construção deste documento foi necessário explorar os conhecimentos inerentes aos projetos desenvolvidos em estágio, nomeadamente os dispositivos e aplicações móveis e o iOS. Assim, conhecer a *Apple* e reconhecer a importância da inovação tecnológica que trouxe ao mercado das comunicações móveis através do *iPhone* foi fundamental para compreender o surgimento do iOS, do seu ambiente de desenvolvimento, o *XCode*, e das linguagens de programação a ele inerentes: o *Objective-C* e o *Swift*.

Após a construção do estado da arte, foram descritos cinco projetos desenvolvidos ao longo do estágio realizados no âmbito da temática do iOS: aplicações multiplataforma, a recolha automatizada de dados estatísticos, os testes automatizados, o *Swift* e o iDoor.

O primeiro projeto remeteu-se à investigação com aplicações móveis associadas, com o intuito único de demonstrar a utilidade das ferramentas multiplataformas no desenvolvimento de aplicações específicas que, no exemplo relatado, continha *streams* de vídeo. No caso estudado, concluiu-se que as ferramentas multiplataforma não seriam a opção a implementar, apesar de possuírem vantagens relativamente às tecnologias nativas.

O *Google Analytics* é uma ferramenta de angariação de dados estatísticos que foi aplicada no MEOGO de forma a estudar, neste âmbito, uma aplicação criada na PTIS. Para realizar este estudo foi necessário implementar e adaptar métodos existentes no projeto bem como interpretar o local ideal para a chamada dos devidos métodos de envio para o *Google Analytics*. Tudo isto foi implementado em *Objective-C*, linguagem usada na aplicação em causa.

Quanto ao projeto que envolveu os testes automatizados de *UIAutomation* no MEOGO iOS, foi desenvolvido cumprindo requisitos previamente especificados. O *script* gerado no âmbito dos testes automatizados em causa é capaz de sintonizar um canal, autenticar um utilizador especificado e escolher um *trailer* para posterior reprodução. Todas estas etapas foram controladas e ilustradas através de capturas de ecrã regulares.

A investigação relativa ao *Swift* revelou diversas diferenças face ao *Objective-C*, nomeadamente vantagens para o programador aquando da construção do código-fonte, pelo que a principal novidade prende-se com a integração do conceito de *playgrounds*. Para demonstrar todos os aspetos positivos desta nova linguagem, foi criada uma aplicação denominada "FlappyPT", apresentada aos colaboradores do MTV na PT Inovação e Sistemas.

Por fim, é explorado o desenvolvimento da aplicação iDoor na nova linguagem da *Apple* e cujo objetivo é a deteção de iBeacons para a abertura automática de portas, um projeto que promete revolucionar o acesso dos trabalhadores aos seus locais de trabalho.

## 5.2. PARECER E TRABALHO FUTURO

A variedade de projetos realizados em estágio possibilitou o desenvolvimento de conhecimentos e aptidões no âmbito da programação em iOS. Assim, foi possível contactar com diversas tecnologias e linguagens de programação em ambiente empresarial, que trouxe vantagens para a minha formação e futuro profissional no sentido em que aprimorei capacidades técnicas, teóricas e interpessoais.

Após a exploração das diversas tecnologias utilizadas nos diversos projetos desenvolvidos em estágio, sugiro que se invista no seu aprimoramento no futuro já que, por exemplo, o *Google Analytics* e o *UIAutomation* carecem de algumas funcionalidades que estimulem os programadores a utilizá-las mais frequentemente nas aplicações móveis que criam. Em relação ao *iDoor* é necessário aperfeiçoar a tecnologia *Bluetooth* em relação à calibração das distâncias entre os *iBeacons* e o utilizador. Isto acontece porque, atualmente, a distância atribuída pelo dispositivo móvel relativamente aos *iBeacons* não apresenta um nível de exatidão perfeito.

Para terminar, gostaria de realçar a importância da realização de estágios profissionais para os alunos do Mestrado em Sistemas de Informação bem como a criação e manutenção das parcerias entre a Universidade de Aveiro e as empresas. Tudo isto permite ao aluno desenvolver competências-chave para o ambiente laboral que não consegue obter na sua formação curricular.

# 6. REFERÊNCIAS BIBLIOGRÁFICAS


APPCELERATOR - NBC.com iPad app reaches # 1 in the App Store and 2 million users within 9 months: Rapid mobile innovation powered by Appcelerator. California. 2012).

APPLE INC. - **Start Developing iOS Apps Today: App Development Process** [Em linha], atual. 2013. [Consult. 2 out. 2014]. Disponível em WWW:<URL:https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/AppDevelopmentProcess.html#//apple_ref/doc/uid/TP40011343-CH4-SW1>.

APPLE INC. - **iOS Technology Overview: About the iOS Technologies** [Em linha], atual. 2014. a. [Consult. 30 set. 2014]. Disponível em WWW:<URL:https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898>.

APPLE INC. - **XCode Overview: About XCode** [Em linha], atual. 2014. b. [Consult. 8 out. 2014]. Disponível em WWW:<URL:https://developer.apple.com/library/ios/documentation/ToolsLanguages/Conceptual/Xcode_Overview/chapters/about.html#//apple_ref/doc/uid/TP40010215>.

APPLE INC. - **Apple Developer: Swift Overview** [Em linha], atual. 2014. c. [Consult. 30 set. 2014]. Disponível em WWW:<URL:https://developer.apple.com/swift/>.

APPLE INC. - **The Swift Programming Language: A Swift Tour** [Em linha], atual. 2014. d. [Consult. 10 out. 2014]. Disponível em WWW:<URL:https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/GuidedTour.html#//apple_ref/doc/uid/TP40014097-CH2-XID_1>.

APPLE INC. - **SpriteKit Programming Guide: About Sprite Kit** [Em linha], atual. 2014. e. [Consult. 7 out. 2014]. Disponível em WWW:<URL:https://developer.apple.com/library/ios/documentation/GraphicsAnimation/Conceptual/SpriteKit_PG/Introduction/Introduction.html>.

APPLE INC. - **MEO Go para iPhone, iPod touch e iPad na App Store no iTunes** [Em linha], atual. 2014. f. [Consult. 17 out. 2014]. Disponível em WWW:<URL:https://itunes.apple.com/pt/app/meo-go/id394637480?mt=8>.

CASA DOS BITS - **SAPO TEK - TEK Mobile: Aplicações móveis ultrapassam os 70 mil milhões de downloads em 2013** [Em linha], atual. 2013. [Consult. 14 out. 2014]. Disponível em WWW:<URL:http://tek.sapo.pt/tek_mobile/apps/aplicacoes_moveis_ultrapassam_os_70_mil_milho_1303730.html>.

CHARLES, A. - **The history of smartphones: timeline** [Em linha], atual. 2012. [Consult. 26 jul. 2014]. Disponível em WWW:<URL:http://www.theguardian.com/technology/2012/jan/24/smartphones-timeline>.

CHING, C. - **Learn Swift From Objective-C : Variables, Classes, Methods and Properties** [Em linha], atual. 2014. [Consult. 29 out. 2014]. Disponível em WWW:<URL:http://codewithchris.com/learn-swift-from-objective-c/>.

CLIFTON, J. - **Apple's Swift: A Developer's First Impression** [Em linha], atual. 2014. [Consult. 25 out. 2014]. Disponível em WWW:<URL:http://smashingboxes.com/ideas/apples-swift-a-developers-first-impression>.

CORRAL, L.; SILLITTI, A.; SUCCI, G. - Mobile multiplatform development: An experiment for performance analysis. **Procedia Computer Science**. Itália. [Em linha]2012) 736–743. [Consult. 24 ago. 2014]. Disponível em WWW:<URL:http://ac.els-cdn.com/S1877050912004516/1-s2.0-S1877050912004516-main.pdf?_tid=e8e745ba-2baa-11e4-9240-00000aab0f26&acdnat=1408897558_ba734ba2766fc399423768bf22921c4c>.

DALMASSO, I. *et al.* - **Survey, Comparison and Evaluation of Cross Platform Mobile Application Development Tools** [Em linha], atual. 2013. [Consult. 7 set. 2014]. Disponível em WWW:<URL:http://www.eurecom.fr/fr/publication/3951/download/cm-publi-3951_1.pdf>.

DIÁRIO DE NOTÍCIAS - **PT: Fim da golden share** [Em linha], atual. 2011. [Consult. 3 ago. 2014]. Disponível em WWW:<URL:http://www.dn.pt/inicio/economia/interior.aspx?content_id=1929837&page=-1>.

DIAS, P.; LEANDRO, M. - **Investidores - Notícias: PT Multimédia adquire SAPO - o portal Internet líder em Portugal com 7 milhões de páginas vistas por mês** [Em linha], atual. 1999. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/InternetResource/PTSite/PT/Canais/Investidores/Pressrel/Noticias/arquivo/com26081999.htm>.

FAIRBAIRN, C. K.; FAHKRENKRUG, J.; RUFFENACH, C. - **Objective-C Fundamentals**. New York : Manning Publications Co., 2012. ISBN 9781935182535.

FONSECA, N. *et al.* - **Desenvolvimento em iOS: iPhone, iPad e iPod Touch - Curso Completo**. 3ª. ed. Lisboa : FCA - Editora de Informática, Lda., 2013. ISBN 978-972-722-786-0.

FROMMER, D. - **10 Ways The iPhone Changed Smartphones Forever** [Em linha], atual. 2009. [Consult. 23 set. 2014]. Disponível em WWW:<URL:http://www.businessinsider.com/10-ways-the-iphone-changed-smartphones-forever-2009-6?op=1>.

GOOGLE - **Google Analytics: Análise da Web e Relatórios** [Em linha], atual. 2014. [Consult. 6 out. 2014]. Disponível em WWW:<URL:http://www.google.com/analytics/>.

GROSSMAN, L. - **Invention Of the Year: The iPhone - Best Inventions of 2007** [Em linha], atual. 2007. [Consult. 23 set. 2014]. Disponível em WWW:<URL:http://content.time.com/time/specials/2007/article/0,28804,1677329_1678542_1677891,00.html>.

HEWLETT-PACKARD DEVELOPMENT COMPANY L.P. - **HP Virtual Museum: Hewlett-Packard 95LX computer** [Em linha], atual. 2012. [Consult. 25 ago. 2014]. Disponível em WWW:<URL:http://www.hp.com/hpinfo/abouthp/histnfacts/museum/personalsystems/0025/0025history.html>.

JANSSEN, C. - **Techopedia: Mobile Application (Mobile App)** [Em linha], atual. 2014. [Consult. 25 out. 2014]. Disponível em WWW:<URL:http://www.techopedia.com/definition/2953/mobile-application-mobile-app>.

JORNAL DE NOTÍCIAS - **MEO alcançou 1 milhão de subscritores** [Em linha], atual. 2011. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.jn.pt/PaginaInicial/Tecnologia/Interior.aspx?content_id=2121541>.

LEMOS, R. M. B. - **Desenvolvimento de aplicações para smartphone usando tecnologias web**. [S.l.] : Universidade de Aveiro, 2013

MARQUES, D. F. N. - **Comunicações unificadas em dispositivos móveis**. [S.l.] : Universidade de Aveiro, 2013

MEO - **MEO: Prémios** [Em linha], atual. 2014. a. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://meo.pt/diversos/premios>.

MEO - **MEO: Mais Serviços - Apps Mobile** [Em linha], atual. 2014. b. [Consult. 13 ago. 2014]. Disponível em WWW:<URL:http://www.meo.pt/telemovel/mais-servicos/apps>.

MOEDAS, A. M. G. - **Desenvolvimento de aplicações móveis para partilha de conteúdos: um estudo desenvolvido em android no âmbito da plataforma SAPO Campus**. [S.l.] : Universidade de Aveiro, 2011

NEAL, R. W. - **Apple iPhone to iPhone 6: The 7-Year Evolution Of A Game-Changing Smartphone** [Em linha], atual. 2014. [Consult. 23 set. 2014]. Disponível em WWW:<URL:http://www.ibtimes.com/apple-iphone-iphone-6-7-year-evolution-game-changing-smartphone-photos-1533776>.

NETSCOPE - **Ranking Padronizado Netscope** [Em linha], atual. 2013. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.netscope.marktest.pt/ranking/Jul13/Rank_Jul_2013_Visitas.htm>.

PEREIRA, A. M. *et al.* - A historia da Apple Computer. **Pretexto**. Belo Horizonte. 2006) 11–24.

PEREIRA, A. M. C. - **Welcome UA : Desenvolvimento de uma aplicação móvel para a UA**. [S.l.] : Universidade de Aveiro, 2013

PETERS, M. - **HOW TO: Get Started With Google Analytics** [Em linha], atual. 2011. [Consult. 27 out. 2014]. Disponível em WWW:<URL:http://mashable.com/2011/05/23/how-to-use-google-analytics/>.

PORTUGAL TELECOM - **Notícias: Mimo faz 10 anos** [Em linha], atual. 2005. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/internetresource/ptsite/pt/canais/media/destaqueshp/destaques_2005/mimo10anos.htm>.

PORTUGAL TELECOM - **Media - Notícias: 10 anos a inovar** [Em linha], atual. 2009. [Consult. 13 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/InternetResource/PTSite/PT/Canais/Media/DestaquesHP/destaques_2009/10AniversarioPTIN.htm>.

PORTUGAL TELECOM - **Sobre a Portugal Telecom: Portugal Telecom** [Em linha], atual. 2014. a. [Consult. 3 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/InternetResource/PTSite/PT/Canais/SobreaPT/>.

PORTUGAL TELECOM - **Sobre a Portugal Telecom: A nossa história** [Em linha], atual. 2014. b. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/InternetResource/PTSite/PT/Canais/SobreaPT/Quem+Somos/A+nossa+historia/>.

PORTUGAL TELECOM - **Investidores: Empresas do Grupo** [Em linha], atual. 2014. c. [Consult. 3 ago. 2014]. Disponível em WWW:<URL:http://www.telecom.pt/InternetResource/PTSite/PT/Canais/Investidores/Links/EmpresasPT/empresaspt.htm>.

PT INOVAÇÃO E SISTEMAS - **PT Inovação e Sistemas: Sobre Nós** [Em linha], atual. 2014. [Consult. 13 ago. 2014]. Disponível em WWW:<URL:http://www.ptinovacao.pt/pt/sobre.html>.

PÚBLICO - **PT abandona marca TMN e aposta no Meo** [Em linha], atual. 2014. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://www.publico.pt/economia/noticia/pt-abandona-marca-tmn-e-aposta-no-meo-1621239>.

REGO, A. F. - **PT atinge 100 milhões de clientes** [Em linha], atual. 2012. [Consult. 3 ago. 2014]. Disponível em WWW:<URL:http://www.jornaldenegocios.pt/empresas/detalhe/pt_atinge_100_milhoes_de_clientes.html>.

RIVERA, J.; MEULEN, R. Van Der - **Gartner Says Annual Smartphone Sales Surpassed Sales of Feature Phones for the First Time in 2013** [Em linha], atual. 2014. [Consult. 26 jul. 2014]. Disponível em WWW:<URL:http://www.gartner.com/newsroom/id/2665715>.

SAPO - **SAPO: Contactos** [Em linha], atual. 2014. a. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://sobre.sapo.pt/pt-pt/contactos>.

SAPO - **SAPO: Sobre nós** [Em linha], atual. 2014. b. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://sobre.sapo.pt/pt-pt/sobre-o-sapo/sobre-nos>.

SAPO - **Produtos e Serviços: Aplicações Móveis** [Em linha], atual. 2014. c. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://mobile.sapo.pt/apps>.

SHERWOOD, S. - **Google Analytics for iOS** [Em linha], atual. 2013. [Consult. 29 out. 2014]. Disponível em WWW:<URL:http://www.raywenderlich.com/53459/google-analytics-ios>.

SIMÕES, P. - **Portugal Telecom apresenta o novo MEO** [Em linha], atual. 2013. [Consult. 10 ago. 2014]. Disponível em WWW:<URL:http://pplware.sapo.pt/informacao/portugal-telecom-apresenta-o-novo-meo/>.

THE GADGETEER - **Is there still a market for PDAs?** [Em linha], atual. 2014. [Consult. 14 out. 2014]. Disponível em WWW:<URL:http://the-gadgeteer.com/2011/12/21/is-there-still-a-market-for-pdas/>.

TIME INC. - **The Apple Revolution: 10 Key Moments** [Em linha], atual. 2014. [Consult. 14 set. 2014]. Disponível em WWW:<URL:http://content.time.com/time/specials/packages/article/0,28804,1873486_1873491_1873530,00.html>.

UNITED NATIONS - **Mobile Technologies and Empowerment: Enhancing human development through participation and innovation** [Em linha], atual. 2013. [Consult. 25 ago. 2014]. Disponível em WWW:<URL:https://www.undpegov.org/mgov-primer.html>.

UNIVERSIDADE DE AVEIRO - **Prémio Casos Exemplares de Cooperação Universidade-Empresa: COTEC premeia "cooperação exemplar" da Universidade de Aveiro com a PT Inovação** [Em linha], atual. 2013. [Consult. 13 ago. 2014]. Disponível em WWW:<URL:http://uaonline.ua.pt/pub/detail.asp?c=36333>.