Sistema de Divulgação dos Serviços e Produtos da Porto Santo Line na Plataforma Android

PROJECTO DE MESTRADO

Agostinho Adrião Gonçalves Neves
MESTRADO EM ENGENHARIA INFORMÁTICA

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

Setembro | 2012

Sistema de Divulgação dos Serviços e Produtos da Porto Santo Line na Plataforma Android

PROJECTO DE MESTRADO

Agostinho Adrião Gonçalves Neves
MESTRADO EM ENGENHARIA INFORMÁTICA

ORIENTAÇÃO
Leonel Domingos Telo Nóbrega

UNIVERSIDADE da MADEIRA

Sistema de Divulgação dos Serviços e Produtos da Porto Santo Line na Plataforma Android

Agostinho Adrião Gonçalves Neves

(Licenciado em Engenharia Informática pela Universidade da Madeira)

Tese Submetida à Universidade da Madeira para a
Obtenção do Grau de Mestre em Engenharia Informática

Funchal - Portugal

SETEMBRO 2012

ORIENTADOR:

Professor Doutor Leonel Domingos Telo Nóbrega

Professor Auxiliar do Centro de Ciências Exactas e da Engenharia da Universidade da Madeira

ABSTRACT

Porto Santo Line is the company responsible for the shipping route between the islands of Madeira and Porto Santo, offering, in addition to transportation, a set of services and products that are associated with their core business. Its customer base includes not only the residents of both islands, but also many of the tourists visiting the archipelago.

The market positioning of this company is made in the traditional way, which includes a network of branches and a website through which customers can access the products and services that constitute the offering of Porto Santo Line. In an effort to strengthen its market presence and intensify its focus on innovation, the company has search for new ways and new channels to reach their customers.

This project aims to equip the Porto Santo Line with an application that extends its market presence, by offering its customers the ability to access their products and services via mobile devices.

This report documents the development of this project, giving primary emphasis to the process adopted, and the system design and evaluation. Throughout the report, it is highlight the most important considerations in developing a project with these characteristics.

KEYWORDS

Porto Santo Line

Lobo Marinho Cruise

Android

Software Development Project

Graphical User Interface for *smartphones*

Usability Tests

RESUMO

A Porto Santo Line é a empresa responsável pela ligação marítima entre as ilhas Madeira e Porto Santo, oferecendo, para além do transporte, um conjunto de serviços e produtos que estão associado à sua actividade principal. A sua base de clientes inclui não só os residentes de ambas as ilhas, mas também muitos dos turistas que visitam o arquipélago.

O posicionamento desta empresa no mercado é feito das formas tradicionais, onde se inclui uma rede de balcões e um *site* através do qual os clientes podem aceder aos produtos e serviços que constitui a oferta da Porto Santo Line. Num esforço de reforçar a sua presença no mercado e de intensificar a sua aposta na inovação, a empresa tem procurado encontrar novas formas e novos canais para chegar aos seus clientes.

Este projecto visa dotar a Porto Santo Line de uma aplicação que estenda a sua presença no mercado, oferecendo aos seus clientes a possibilidade de aceder aos seus produtos e serviços através de equipamentos móveis.

Este relatório documenta a realização deste projecto, dando principal enfase ao processo adoptado, assim como ao desenho do sistema e sua avaliação. Procurou-se igualmente evidenciar ao longo do relatório, as considerações mais relevantes na elaboração de um projecto com estas características.

PALAVRAS-CHAVE

Porto Santo Line

Navio Lobo Marinho

Plataforma Android

Projecto de Desenvolvimento de *Software*

Interface do Utilizador em *smartphones*

Testes de Usabilidade

AGRADECIMENTOS

O meu obrigado a todas as pessoas que contribuíram para a realização deste projecto.

Manuel Cardoso por partilhar a sua visão, pela confiança depositada e pelo apoio indispensável em todas as fases do projecto.

Professor Leonel Nóbrega pela orientação, inspiração e por estar sempre disponível para ajudar (e pelas várias horas em reuniões).

Dra. Maria João pela contribuição de ideias para o projecto, pela simpatia e transparências nas reuniões.

Dr. Miguel Chaves por disponibilizar acesso à infra-estrutura da Porto Santo Line e pelo apoio técnico.

Amigo Carlos pelas conversas de reflexão sobre os desafios que surgiram e pela dedicação em testar a aplicação.

Os meus pais e irmãos pelo apoio nesta etapa da minha vida.

A minha cara-metade Luísa por ser a minha principal motivação e pela capacidade extraordinária de ouvir-me sem adormecer.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÓNIMOS

CDMA - Code Division Multiple Access

NFC - Near Field Communication

GPS - Global Positioning System

IDE - Integrated Development Environment

SDK - Software Developer's Kit

API - Application Programming Interface

ADT - Android Development Tools

DP - Density-Independent Pixels

MDPI - Medium Dots Per Inch

WYSIWYG - What You See Is What You Get

JSON - JavaScript Object Notation

KML - Keyhole Markup Language

MYSQL - Structured Query Language

CSS - Cascading Style Sheets

XML - Extensible Markup Language

FTP - File Transfer Protocol

SSH - Secure Shell

XP - Extreme Programming

PDF - Portable Document Format

RF - Requisito Funcional

RNF - Requisito Não Funcional

POI - Ponto de Interesse

OMG - Object Management Group

UML - Unified Modeling Language

1. INTRODUÇÃO

"We are targeting innovation. We believe mobile applications are essential."
Larry Page (Co-fundador da Google)

O que começou por ser uma versão móvel do telefone, rapidamente evoluiu para uma ferramenta de produtividade e eficiência. Os *smartphones* revolucionaram o modo como as pessoas acedem à informação, comunicam e adquirem bens e serviços.

A evolução das tecnologias móveis na última década foi notável, com dispositivos a atingir a mesma capacidade de processamento que os computadores convencionais, tornando-se notório a transição de algumas tarefas tradicionalmente executadas nos computadores de secretária, para os *smartphones* transportados no bolso.

Os dispositivos móveis acompanham as pessoas no seu dia-a-dia, auxiliando na execução de tarefas diárias, como na compra de bilhetes de transporte ou em obter direcções.

A plataforma Android possui a maior taxa crescimento no mercado dos sistemas operativos móveis (tabela 1), contando já com milhões de utilizadores em todo o Mundo. Apoiado pela Google, a plataforma Android rivaliza com os sistemas da Microsoft e da Apple. O desenvolvimento de aplicações Android tem acompanhado o forte crescimento da própria plataforma, com 600 mil aplicações prontas para *download* através da loja online *Google Play*.

Com a ausência de fronteiras geográficas e por um custo relativamente baixo, é possível difundir uma marca ou produto por milhares de utilizadores, possibilidade o que tem contribuído para um aumento no número de empresas interessadas em enquadrar os seus produtos e serviços no mercado móvel.

Tendo a consciência da mais-valia única de ter os clientes "reféns" durante 2 horas e 30 minutos no navio (altura na qual os clientes planeiam, procuram informações e decidem o que fazer no destino), surge aqui a oportunidade de ser a Porto Santo Line a detentora do índice descritivo das experiências turísticas disponíveis na ilha do Porto Santo, através do mercado móvel.

1.1. Motivação

Tendo em vista potencializar o incremento de serviços prestados ao segmento turístico de Clientes que visitam a R.A.M. através da Porto Santo Line, existe uma forte lacuna nas informações disponibilizadas aos seus potenciais clientes sobre a Ilha do Porto Santo.

Identificaram-se os nichos comunicacionais mais promissores e potenciáveis: o destino e os serviços a bordo do Navio.

O turista típico que compra os serviços da Porto Santo Line fá-lo por via da tarifa "Cruzeiro de 1 Dia", que comporta o serviço de transferes desde a recepção do Hotel até ao navio e o caminho inverso. Estas tarifas pressupõem, que o turista ao fim de um dia de visita à ilha, regresse nessa mesma data à Madeira. O espaço temporal em que os turistas iniciam a viagem e regressam à Madeira compreende sensivelmente 12 horas.

Existe um hiato temporal em que é deixado ao seu critério do turista as actividades a desenvolver na Ilha do Porto Santo. É exactamente neste hiato que a Porto Santo Line quer potencializar o destino, facilitando o acesso a informações (concentradas, simples, e em diversos idiomas) dando a conhecer ao cliente, uma vasta gama de serviços contratáveis já implementados na Ilha, enfatizando como é óbvio, os serviços prestados por empresas do universo Porto Santo Line.

A dificuldade que hoje existe em encontrar contactos de estabelecimentos comerciais, esclarecimentos de dúvidas sem barreira linguística e essencialmente o mero conhecimento da existência de serviços já disponíveis na Ilha, fez a Porto Santo Line sentir a responsabilidade de satisfazer essa necessidade premente do cliente.

1.2. Objectivos

Com este projecto é pretendido:

- O desenvolvimento e concepção de um sistema, baseado na plataforma Android, para divulgação dos serviços e produtos turísticos da Porto Santo Line.
- Divulgação dos serviços a bordo do navio Lobo Marinho.
- Incorporar um módulo de encaminhamento dos passageiros às atracções indexadas na aplicação, com destaque para os serviços prestados pelas empresas do universo Grupo Sousa.

1.3. Organização da Dissertação

Este documento encontra-se dividido em sete capítulos:

1. **:** Descreve o contexto em que se insere este projecto.

2. **:** Apresenta o estado das tecnologias envolvidas no desenvolvimento, designadamente a tecnologia móvel e a plataforma Android.

3. **:** Neste capítulo é abordada a metodologia seguida e são apresentados os casos de utilização, juntamente com os requisitos, *wireframes* e protótipos da *interface* do utilizador. Por fim é argumentada a escolha da arquitectura "Modelo 3 camadas" para este projecto.

4. **IMPLEMENTAÇÃO:** Descrição das bibliotecas java utilizadas, *API's* externas, código da aplicação e *interface* do utilizador.

5. **TESTES E RESULTADOS:** Apresentação dos erros detectados durante a realização de testes de usabilidade à *interface* do utilizador.

6. **LANÇAMENTO:** Breve descrição do processo de publicação da aplicação no mercado Android.

7. **CONCLUSÕES:** Resumo geral do desenvolvimento realizado e sugestão de funcionalidades adicionais a implementar no sistema.

2. ANDROID E TECNOLOGIAS UTILIZADAS

"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

(Mark Weiser)

O desenvolvimento de um projecto de *software* exige uma análise às tecnologias disponíveis, a fim de tentar realizar as melhores escolhas para o projecto.
Igualmente importante, é estudar e compreender a evolução da plataforma alvo, de forma a enquadrar o próprio projecto na evolução do sistema.

2.1. Comunicações Móveis

Possuir um equipamento móvel deixou de ser um luxo e tornou-se numa necessidade. A inovação tecnológica e a sua difusão global são dois factores de peso para a evolução da tecnologia móvel [3].
Embora o telemóvel tenha sido inicialmente apresentado como um equipamento de comunicação, concebido para efectuar e receber chamadas telefónicas, o seu papel tem expandido de forma a satisfazer as necessidades e exigências dos consumidores.

Assinantes de comunicações móveis no Mundo:
Em finais de 2011 existiam aproximadamente 6 biliões de assinantes. O equivalente a 87% da população mundial. Em 2010 o número era de 5,4 biliões e 4,7 biliões em 2009, o que representa um enorme crescimento no número de assinantes *(The International Telecommunication Union, 2011).*

Equipamentos de comunicação móvel:
Juntamente com o aumento de assinantes, surgiram novas necessidades e exigências por parte do consumidor final. Ler livros no metro, realizar operações bancárias durante o café, reservar viagem ou consultar partidas e chegadas, são operações facilmente executadas a partir de um *smartphone* ou *tablet*. Estes dois tipos de equipamento móvel, *Smartphone* e *Tablet*, são de longe os equipamentos com maior crescimento no número de vendas dos últimos dois anos *(Mobithinking, 2012)*.

Sistemas Operativos dominantes:

A luta por dominar o mercado dos *smartphones* não tem sido fácil nestes últimos 2 anos. Gigantes dos sistemas operativos (como a Microsoft) têm perdido terreno para gigantes de outras áreas (Google). Existem inúmeros factores que contribuem para esta mudança no mercado, como desempenho, funcionalidades, fiabilidade, robustez e segurança.

Independentemente dos motivos exactos desta mudança no mercado móvel, é importante analisar quais os sistemas operativos que estão a ganhar esta luta. No final de contas, ninguém deseja investir tempo e dinheiro a desenvolver aplicações para sistemas em declínio.

Sistema Operativo	*Vendas 2011 (milhões)*	*Mercado 2011*	*Crescimento Anual*
Android	**237.7**	**48.8%**	**244%**
iOS	**93.1**	**19.1%**	**96%**
Symbian	**80.1**	**16.4%**	**-29.1%**
BlackBerry	51.4	10.5%	5.0%
Bada	**13.2**	**2.7%**	**183.1%**
Windows Phone	6.8	1.4%	-43.3%
Outros	5.4	1.1%	14.4%
Total	487.7	100%	62.7%
Fonte: Canalys (Feb 2011)			*via: mobiThinking*

Tabela 1 - Mercado Mundial dos smartphones, ordenados por Sistema Operativo

Com base nos dados da tabela 1, torna-se perceptível que o sistema Android domina o número de vendas, detendo já a maior parte do mercado de *smartphones* desde 2011 e possuindo também a maior taxa de crescimento anual. O sistema da Apple, iOS, apresenta-se como o rival directo do Android, com um crescimento anual de 96% e uma quota de mercado de aproximadamente 19,1%. A plataforma da Samsung, Bada, embora no último ano tenha registado um crescimento de 183,1%, a sua presença no mercado é muito baixa, com apenas 2,7%. O sistema operativo da Nokia, o Symbian e a plataforma Windows Phone da Microsoft, têm registado uma forte queda, essencialmente devido à grande aceitação das plataformas Android e iOs no mercado americano e europeu.

2.2. Evolução da Plataforma Android

O sistema operativo da Google começou com Andy Rubin em 2003, com a criação da empresa Android Inc., que acabou por ser adquirida pela Google em 2005, cujo o objectivo inicial era concorrer contra o BlackBerry. Com a Apple a lançar o iPhone e o seu iOS, o BlackBerry passou para segundo plano e o iOS passou a ser o novo

concorrente directo do Android. Em 2008, a primeira versão do novo sistema operativo da Google, Android 1.0, foi apresentada ao público, impulsionada pelo lançamento do *smartphone* HTC G1 [5]. Apenas três meses depois, e após a correcção de vários *bugs*, a versão 1.1 do sistema operativo foi disponibilizada para instalação.

Android 1.5 – Cupcake:

A versão 1.5 representou a primeira grande revolução na história da evolução do Android. Para além de algumas alterações em termos estéticos como a adição de transparência em alguns *widgets* da Google, foram também adicionas importantes funcionalidades tais como um teclado virtual (figura 1) que até ao momento ainda não era parte integrante do Android. Foi também na versão 1.5 que a Google disponibilizou o *SDK*, permitindo o desenvolvimento de *widgets*, jogos e outras aplicações, por parte de milhões de programadores de todas as partes do mundo.

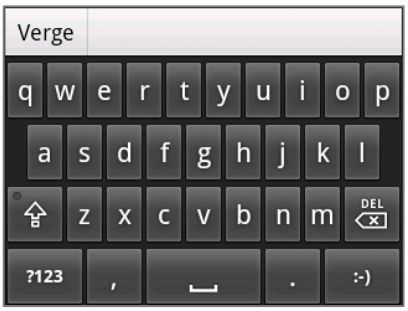


Figura 3 - Teclado Virtual

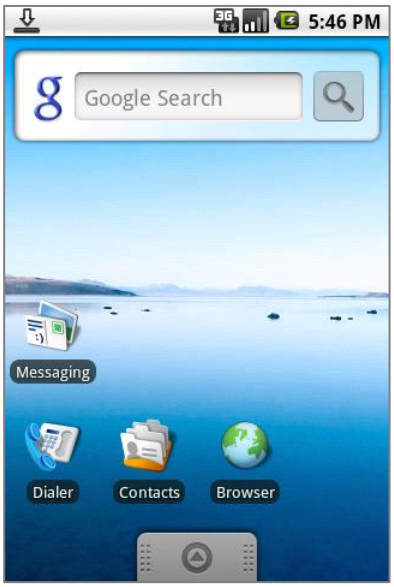


Figura 4 - *Home screen do* Android 1.5

Android 1.6 – Donut:

Talvez a maior alteração nesta versão foi o suporte de ecrãs com diferentes resoluções e formatos, figura 3. Com esta mudança, novos equipamentos móveis começaram a surgir com resoluções de ecrã diferentes dos habituais 320*480, até ao momento norma para muitos fabricantes. Outra grande alteração, que contribuiu imenso para a aceitação no mercado, foi o suporte à tecnologia de comunicação CDMA. Possibilitou a comercialização de equipamentos móveis Android por parte da gigante operadora móvel americana *Verizon* e de várias operadoras do mercado asiático.

Figura 5 - Novas resoluções de ecrã com a versão 1.6

Android 2.0 / 2.1 – Eclair

Em Novembro de 2009, um ano após o lançamento do G1, é lançada a versão 2.0 do Android. A maior evolução visual e arquitectural desde a primeira versão pública. Entre várias novas funcionalidades, as com maior impacto no mercado foi a oferta do *Google Maps Navigation*, oferecendo aos utilizadores uma aplicação de navegação automóvel totalmente gratuita e completa, com a excepção da falta de suporte *offline*. Melhorias foram feitas tanto no teclado virtual como no ecrã de bloqueio, figuras 4 e 5.

Figura 6 - Novo teclado virtual na versão 2.1

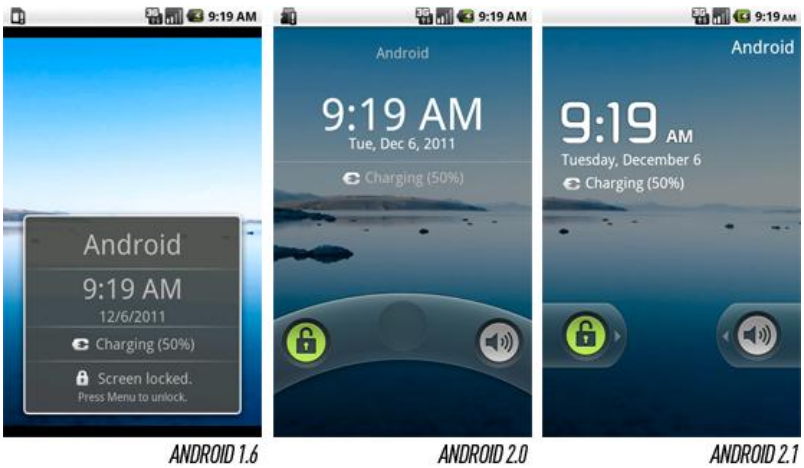


Figura 7 - Evolução do ecrã de bloqueio

A **versão 2.1** foi um passo estratégico de marketing. A Google optou pela HTC para fabricar o primeiro smartphone oficial Google, o Nexus One, apresentado na figura 6. Oferecendo deste modo, pela primeira vez, o sistema operativo Android sem qualquer modificação. Este equipamento tornou-se rapidamente o dispositivo favorito de muitos programadores Android e a base de comparação dos dispositivos que seriam lançados posteriormente.

Figura 8 - Nexus One: O primeiro smartphone oficial da Google

Android 2.2 – Froyo:

Lançado em 2010, primeiro para o Nexus One e depois para os restantes equipamentos de forma gradual, o novo Android apresentava um ecrã inicial redesenhado com base no feedback dos utilizadores (figura 7), focado para a usabilidade e melhor uso do espaço livre do ecrã. Para substituir o antigo painel de botões principais, esta nova versão contava com três botões principais: Chamar, Aplicações, Web Browser.

Várias melhorias foram feitas em termos de desempenho e segurança. O suporte a tecnologia WIFI foi melhorada e os utilizadores podem agora tirar melhor proveito dos *hotspots* gratuitos espalhados pelo mundo.

Figura 9 - – *Home screen* do Android 2.2

Android 2.3 – Gingerbread:

Lançada aproximadamente seis meses após o lançamento da versão 2.2, Gingerbread – a Google apresentou o seu segundo smartphone, desta vez produzido pela Samsung, o Nexus S que veio essencialmente posicionar o Android no mercado dos jogos móveis. Novamente o teclado virtual (figura 8) do Android foi alvo de melhorias, com mais espaço entre as teclas a fim de facilitar a escrita e com suporte multi-toque. A tecnologia NFC (Near Field Communication) passou a estar presente no sistema Android, dando origem a várias aplicações para micro-pagamentos, como revistas ou cafés.

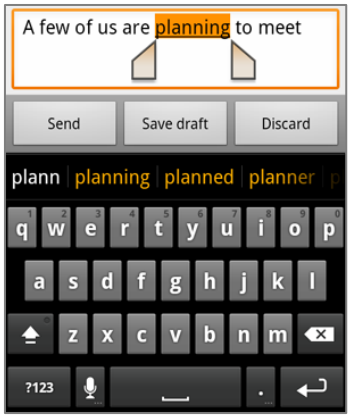


Figura 10 - Teclado virtual no Android 2.3

Android 3.x – Honeycomb:

Concebido para os tablets e não propriamente para os smartphones, Honeycomb contribuiu com um *design* completamente novo (figura 9), apostando ainda mais na usabilidade. Foi com o Xoom, produzido pela Motorola, que o Android entrou no mercado dos *tablets*.

O maior impacto desta versão para os smartphones, foi o início do fim (ou pelo menos da redução) dos botões físicos. O botão físico "*Menu*" deixou de ser um requisito para os equipamentos móveis Android, o que obrigou as aplicações a adaptarem a interface gráfica, adicionando um botão "*Menu*".

Figura 11 - Android 3.0 para tablet

Android 4.0 – Ice Cream Sandwich:

Até ao momento, é a maior actualização do sistema Android. Nova interface, novo *design* gráfico, melhor desempenho e maior segurança são os pontos fortes desta versão.

A grande novidade nesta versão é a "***Action Bar***", figura 10, que redefiniu a forma como o utilizador interage com as aplicações. As principais acções passam a estar todas nesta barra (3), uniformizando assim, a posição das acções essenciais para as aplicações Android. A *action bar* normalizou a utilização do ícone da aplicação como botão de navegação, que deverá funcionar como *link* para o menu principal (2) ou como botão de retroceder (1). É ainda introduzido o botão *overflow (4)*, responsável por conter todas as restantes acções, consideradas menos essenciais e que por esse não devem ocupar espaço na *action bar.*

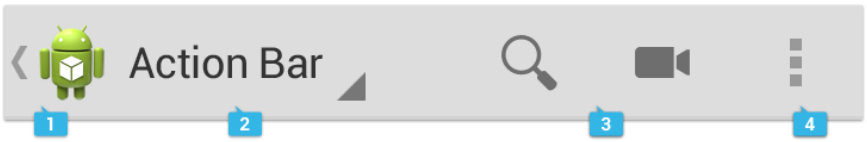


Figura 12 - A Action Bar obrigou ao redesenho das interfaces gráficas de várias aplicações

Outra novidade foi o novo sistema de notificações, que foi alterado de forma a permitir remover notificações de forma individual, algo que não era possível até ao momento, tornando-o um dos melhores sistemas de notificações das plataformas móveis. O ecrã inicial, na seguinte figura, possui agora uma secção "Favoritos" na qual é possível colocar atalhos para qualquer aplicação que o utilizador deseje.

Android Beam é uma nova funcionalidade que a versão 4.0 trouxe consigo. Esta funcionalidade, baseada na tecnologia NFC, permite a troca de ficheiros entre dois equipamentos fisicamente próximos.

Figura 13 - – *Home screen* do Android 4.0

Android 4.1 – Jelly Bean:

A mais recente actualização da Google para o Android, anunciada a 27 de Junho de 2012, na conferência anual Google I/O. Foram realizadas melhorias na usabilidade da interface de utilizador de todo o sistema Android, tanto para os *smartphones* como para os *tablets*. Talvez o melhoramento mais notado nesta versão, foi o aperfeiçoamento do tempo de resposta no ecrã inicial (figura 12), agora muito próximo da famosa suavidade do sistema iOS. Outra grande alteração em termos de funcionalidades, é a possibilidade do utilizador bloquear aplicações de apresentar notificações.

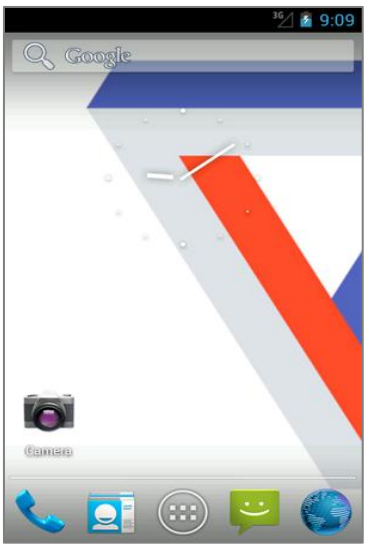


Figura 14 – *Home screen* no Android 4.1

2.3. Segmentação da Plataforma

O lançamento de diferentes versões em relativamente pouco tempo tem as suas desvantagens. A segmentação da plataforma, ou por outras palavras, a existência de inúmeras versões do sistema no mercado, apresenta desafios acrescidos para os programadores Android. Uma aplicação desenvolvida para 2.2 é executável num sistema 4.1, mas o contrário não, é importante decidir para qual versão a aplicação deve ser desenvolvida.

Como o objectivo é alcançar o maior número de utilizadores possíveis, desenvolver para a versão mais antiga irá garantir que a aplicação poderá ser executada pela grande maioria dos utilizadores, mas como vimos na secção anterior, grandes melhorias foram implementadas ao longo das várias versões e por este motivo é essencial encontrar um bom ponto de equilíbrio entre o conjunto de funcionalidades Android que pretendemos incorporar e a versão mínima necessária.

Felizmente a Google monitoriza as activações dos dispositivos Android, registando a versão do sistema que é activado, normalmente ao aceder ao Google Play (Android Market). Estes dados, apresentados na tabela seguinte, são actualizados mensalmente e disponibilizados de forma gratuita no *site* oficial do Android [6].

Versão	Nome Código	API	Distribuição
1.5	Cupcake	3	0.2%
1.6	Donut	4	0.5%
2.1	Eclair	7	4.2%
2.2	Froyo	8	15.5%
2.3 - 2.3.2	Gingerbread	9	0.3%
2.3.3 - 2.3.7		10	60.3%
3.1	Honeycomb	12	0.5%
3.2		13	1.8%
4.0 - 4.0.2	Ice Cream Sandwich	14	0.1%
4.0.3 - 4.0.4		15	15.8%
4.1	Jelly Bean	16	0.8%

Tabela 2 – Segmentação da Plataforma Android em Agosto 2012

Quando visualizamos os mesmos dados no próximo gráfico circular, torna-se claro qual a versão dominante na segmentação actual (2.3.3).

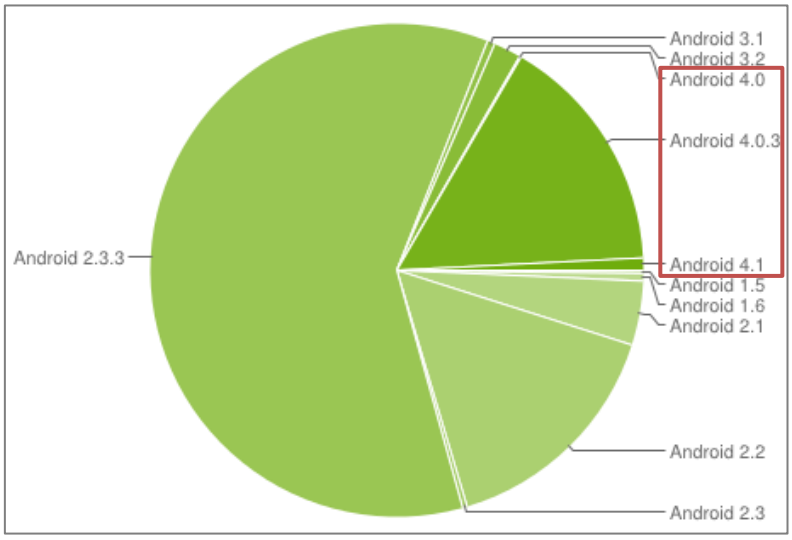


Figura 15 - Segmentação da Plataforma Android em Agosto 2012

Ao comparar com a anterior segmentação de Janeiro de 2012, apresentada seguidamente, é possível verificar que a nova versão 4.0 ganhou rapidamente uma grande fatia do mercado.

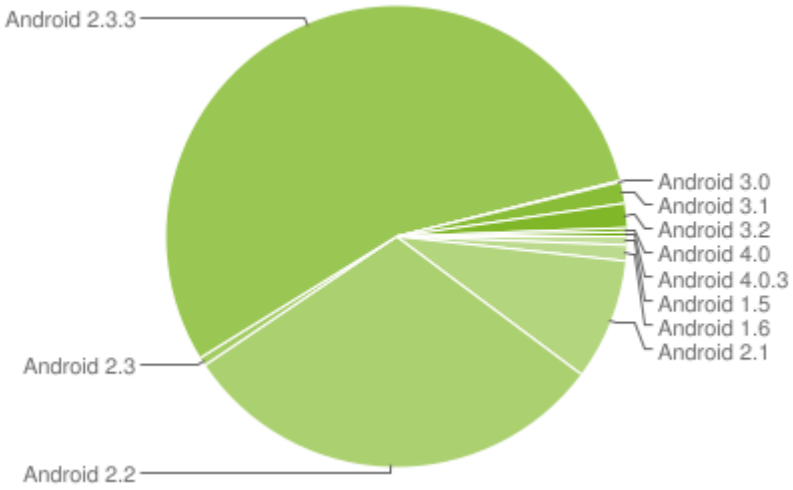


Figura 16 - Segmentação da Plataforma Android em Janeiro 2012

Com base na evolução do sistema Android, nos dados da Tabela 2 e nos requisitos da aplicação (que serão discutidos posteriormente), a versão 2.2 apresenta-se como a melhor escolha para a versão mínima da aplicação, alcançando 79,6% de todos os dipositivos Android existentes e contendo as melhores funcionalidades do sistema.

2.4. Web vs Nativa

APLICAÇÃO WEB MÓVEL:

Consiste numa aplicação web formatada para *smartphones* e *tablets*, acedida pelo *browser* do dispositivo móvel. Consiste essencialmente nas tecnologias HTML, CSS e JavaScript. Com a tecnologia HTML5 é possível criar uma aplicação web que funcione sem acesso à Internet, porém a tecnologia encontra-se em desenvolvimento. A grande vantagem das aplicações web é a compatibilidade multiplataforma, facilitando imenso o desenvolvimento de uma aplicação para vários sistemas operativos (Android, iOs, Windows) e reduzindo drasticamente os custos de manutenção [8]. Porém, as aplicações web perdem na interacção com o *hardware* do equipamento. Como exemplo, uma aplicação web terá acesso limitado ao GPS do equipamento móvel, logo não será a melhor escolha para uma aplicação que necessite de calcular a melhor rota entre o ponto A e o ponto B.

Vantagens:

- Compatibilidade multiplataforma.
- Manutenção económica e facilitada.
- Fácil acesso: Utilizador acede por URL, sem qualquer *download.*
- Desenvolvimento multiplataforma facilitado.

Desvantagens:

- Acesso limitado ao *hardware* do dispositivo.
- Pouca ou nenhuma interactividade com as restantes aplicações instaladas no mesmo dispositivo móvel.
- Excesso de complexidade para implementar determinadas funcionalidades consideradas “básicas” em aplicações nativas.
- Interface de utilizador com menos desempenho: Para equiparar com a velocidade de reacção de uma aplicação nativa, a aplicação web exige maior esforço no desenvolvimento.

APLICAÇÃO NATIVA:

Desenvolvida especificamente para uma plataforma, uma aplicação nativa tem a possibilidade de aproveitar todo o potencial do equipamento para o qual é desenvolvida. Interage facilmente com outras aplicações instaladas no dispositivo, tais como, cliente de email, rede social ou lista de contactos. O utilizador realiza o *download* da aplicação a partir de uma loja online de aplicações (Google Play para

Android), e depois de instalada, receberá automaticamente todas as actualizações lançadas. No momento da elaboração deste documento, as aplicações nativas são a opção mais comum e o tipo de aplicação preferido pela maioria dos utilizadores [7].

Vantagens:

- Utilização de todo o potencial das tecnologias disponíveis no equipamento móvel (GPS, Acelerómetro, Câmara, Bússola).
- Melhor experiência de utilização, com uma interface de utilizador mais fluída.
- Interacção com outras aplicações instaladas no dispositivo.
- Maior visibilidade, as aplicações nativas tornam-se parte integrante da plataforma, apresentando o nome e logo da aplicação em menus de contexto, gestor de aplicações e notificações.
- Desempenho superior para processamento de imagens e vídeos.

Desvantagens:

- A aplicação terá que ser desenvolvida praticamente de forma independente para cada plataforma.
- Maiores custos de desenvolvimento e manutenção.
- A mesma aplicação poderá oferecer funcionalidades diferentes, dependendo dos limites de cada plataforma.
- Requer ambiente de desenvolvimento (IDE, SDK) próprio para cada plataforma.

Ambas as opções oferecem vantagens e desvantagens para o desenvolvimento de uma aplicação móvel. Tudo depende das funcionalidades e requisitos de cada aplicação. Um cliente que pretenda apenas adaptar a imagem da sua empresa ao mundo móvel, sem oferecer qualquer outra funcionalidade, a melhor abordagem a seguir é a aplicação web, pois irá abranger um mercado maior, com custos inferiores. Se por outro lado, o objectivo é usufruir das capacidades tecnológicas dos dispositivos móveis, como GPS, gestão de ficheiros ou execução de várias tarefas em simultâneo, a melhor escolha será uma aplicação nativa.

2.5. Ambiente de Desenvolvimento

Nesta secção é descrito resumidamente o estado de arte de cada ferramenta e tecnologia envolvidos no projecto. O objectivo não é realizar uma descrição exaustiva de cada ferramenta, mas fornecer a informação mínima necessária para compreender a metodologia e organização envolvida no desenvolvimento do projecto.

JAVA STANDARD EDITION DEVELOPMENT KIT (JDK):

A plataforma Android conta com a portabilidade e potência do sistema operativo Linux, com a fiabilidade e portabilidade de uma linguagem de programação de alto nível e com uma poderosa API. As aplicações Android são desenvolvidas em Java, compiladas de acordo com a API Android e convertidas em *Java bytecode* para a Máquina Virtual Dalvik.

Versão:

- Java SE Development Kit 7u6

ECLIPSE IDE JAVA:

Apesar de ser possível escrever código java com um simples editor de texto, esta tarefa é obviamente facilitada com a utilização de um IDE apropriado. A Google recomenda a utilização do Eclipse, embora o Netbeans seja também uma boa opção.

Versão:

- Eclipse IDE for Java Developers - Juno

ANDROID SDK

Fornece bibliotecas java e ferramentas de desenvolvimento para criar e testar aplicações Android.

Versão:

- Android SDK Tools, Revision 20

ANDROID ADT PLUGIN:

Android Development Tools (ADT), é um plugin para o Eclipse, oferecido pela Google, para aumentar a eficiência no desenvolvimento de aplicações Android. Entre várias melhorias que são aplicadas ao Eclipse, é adicionado um editor

WYSIWYG de interfaces de utilizador, um emulador do sistema operativo Android no qual é possível instalar e executar aplicações em várias resoluções de ecrã e um assistente de exportação e publicação das aplicações para o Google Play (Android Market).

Versão:

- ADT 20.0.3

2.5.1. Controlo de Versões – Subversion e Subclipse

Embora este projecto não seja desenvolvido num ambiente empresarial, com uma equipa de programadores, o controlo de versões é igualmente importante. Adicionalmente, manter uma cópia do código num servidor remoto não só garante uma maior segurança, como também torna-se prático para desenvolvimento em diferentes postos de trabalho.

Subversion é um sistema de controlo de versões *open source*, desenvolvido pela *Apache Software Foundation* e destinado tanto para indivíduos como organizações.

Subclipse permite utilizar o *Subversion* directamente no Eclipse. É um *plugin* que possibilita o controlo de versões (*commit, synchronize, revert, …)* através do próprio ambiente de desenvolvimento.

Assembla é um repositório SVN gratuito que permite alojar o código eficientemente. Permite também manter um registo das alterações que foram realizadas no código, para comparação durante o *merge* ou para simples consulta.

A utilização de um sistema de controlo de versões provou ser extremamente útil e de grande valor, principalmente pela possibilidade de reverter para uma versão anterior do código, a fim de detectar e corrigir erros.
Possuir um histórico de alterações, figura seguinte, também é relevante para compreender a evolução do código.

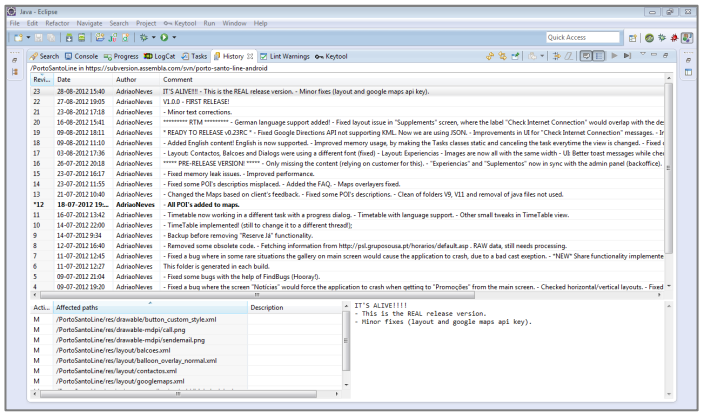


Figura 17 – Subversion: Histórico de versões (formato maior em anexo)

2.6. Outras Tecnologias Utilizadas no Projecto

Sendo uma solução baseada na arquitectura cliente-servidor, é imprescindível a utilização de outras tecnologias. As seguintes tecnologias foram necessárias para a implementação dos módulos de comunicação entre a aplicação cliente (Android) e o painel de administração (servidor). Tendo em conta o público alvo deste documento e a popularidade das seguintes tecnologias, não serão realizadas descrições exaustivas. Sempre que possível, foi optado por tecnologia *open source*, a fim de manter os custos no mínimo possível.

SERVIDOR APACHE:

Alojamento adquirido pelo cliente, Porto Santo Line, exclusivamente para o desenvolvimento deste projecto.

Versão:

- Apache 2.2.22

MYSQL:

Base de dados para armazenamento da informação a ser transferida entre o servidor e o cliente.

Versão:

- MySQL 5.0.95-community

PHP:

Linguagem de script *open source*, cujo código é executado apenas do lado do servidor, gerando HTML que é lido pelo *browser* do utilizador.
Dentro do contexto do projecto, é utilizada para o desenvolvimento da aplicação servidor (painel de administração).

Versão:

- PHP 5.2.17

JSON:

Formato optado para a troca de dados entre o cliente e servidor.

JSOUP:

Biblioteca Java para a extracção de informação a partir de páginas HTML. Compatível com HTML5.

Versão:

- jsoup1.6.3

OUTROS PLUGINS PARA ECLIPSE:

- Analisador de código: PMD 5.0.0
- Analisador de código: FindBugs 2.0.1
- UML: ObjectAid 1.0.10

OUTRAS FERRAMENTAS:

- Edição de imagens: Adobe Photoshop CS5.1
- Elaboração de *wireframes:* Evolus Pencil
- HTML e PHP: Notepad++
- Partilha de ficheiros com o cliente: Dropbox
- Acesso FTP: FileZilla
- Acesso SSH: WinSCP
- Pesquisa: Mendeley Desktop

2.7. Resumo

As tecnologias móveis são já parte integrante da vida quotidiana da maioria da população mundial. Os equipamentos móveis influenciam imenso na forma como as pessoas comunicam entre si e como acedem à informação. É um mercado em forte crescimento, com imensas oportunidades de negócio.

As plataformas Android (Google) e iOS (Apple) são neste momento dominantes no mercado de sistemas operativos móveis, com concorrentes como a Microsoft com o seu Windows Phone, ou a Blackberry a perderem mercado, a plataforma Android é das melhores apostas para quem procura investir tempo e dinheiro em desenvolvimento de aplicações móveis.

A evolução do sistema Android é um verdadeiro caso de sucesso de desenvolvimento de *software*. Através da recolha de *feedback* dos utilizadores e da análise de hábitos de utilização, a Google conseguiu adaptar o sistema às verdadeiras necessidades dos seus utilizadores. Seguir a evolução da plataforma Android, é um excelente modo de aprender com os melhores na área de desenvolvimento de *software*.

A plataforma Android é actualmente utilizada em *smartphones, tablets,* televisões e até em automóveis. Esta rápida evolução representa um risco para os programadores: a aplicação em desenvolvimento tornar-se obsoleta antes do lançamento. Por este motivo, é importante desenvolver a pensar na mudança, não só dos requisitos do cliente, mas também da própria plataforma.

A segmentação da plataforma, originada pelo lançamento de várias versões em relativamente pouco tempo, acrescenta desafios ao desenvolvimento. Qualquer programador deseja implementar a última tecnologia na sua aplicação, mas também deseja alcançar o maior número possível de utilizadores. A disponibilização de bibliotecas de compatibilidade, contribuem para diminuir a actual segmentação.

Embora as aplicações web, devido a sua interoperabilidade, permitam alcançar um maior grupo de utilizadores, pecam na oferta de funcionalidades. As aplicações nativas, neste momento, continuam a oferecer melhores vantagens em comparação com as aplicações web.

Actualmente existem no mercado várias aplicações móveis da área do turismo e viagens, mas nenhuma é desenvolvida exclusivamente para os produtos e serviços da empresa Porto Santo Line. Todavia, as aplicações existentes fornecem inspiração para a resolução de determinados problemas de usabilidade e funcionalidade.

O ambiente de trabalho actual para o desenvolvimento eficiente de uma aplicação móvel para Android, consiste no IDE Eclipse (Juno), Android SDK (v.20), Android ADT (v.20.0.3) e Java SE Development Kit 7u6.

O controlo de versões é prática corrente dos processos de desenvolvimento e o Apache Subversion continua a ser dos mais utilizados no meio, em parte graças à sua fiabilidade e simplicidade de utilização.

3. DESENHO DO SISTEMA

"Arquitectura é a organização fundamental de um sistema incorporada em seus componentes, seus relacionamentos com o ambiente, e os princípios que conduzem seu design e evolução."

(Definição de Arquitectura de Software pelo Padrão ISO/IEEE 1471-2000)

3.1. Metodologia

Dos vários métodos de desenvolvimento existentes, o que mais se aproxima da metodologia aplicada neste projecto é o método "*Agile*".

Embora o método "*Agile*" seja destinado a equipas de desenvolvimento com mais de um programador [10], é possível encontrar várias características deste método [11] que foram adoptadas no desenvolvimento deste projecto. A constante evolução dos requisitos, a adaptação do planeamento, o processo de desenvolvimento evolutivo e a rápida resposta à mudança, são algumas das principais características da metodologia seguida, que são comuns ao método "*Agile"* [10]. Dos vários processos que pertencem à categoria "*Agile*", existem atributos do método "*Extreme Programming*" [12, 13] que podem também ser atribuídos à metodologia seguida no desenvolvimento deste projecto. Tais como:

- Aceitar a mudança;
- Simplicidade e fácil leitura do código;
- Revisão constante do código;
- Contar com alterações nos requisitos do cliente;
- Comunicação frequente e colaboração com o cliente;
- Desenvolvimento de funcionalidades apenas quando necessárias;
- Programar, testar, ouvir, planear;

Embora não tenha sido realizada programação em pares, característica típica de XP (*Extreme Programming*) [13], este método continua a ser o mais próximo da metodologia realmente seguida.

Especial destaque para a comunicação constante com o cliente em todas as fases do projecto. O cliente colaborou de forma activa em todos os aspectos, desde a concepção ao processo de distribuição e publicação da aplicação, ocupando o papel central nas decisões no âmbito da lógica de negócio.

3.2. Casos de Utilização

"*Um caso de utilização é a descrição de uma interacção específica que um utilizador poderá realizar na aplicação*" [14].

Como qualquer especificação, os casos de utilização devem ser o mais simples e directos possíveis, pois todo este levantamento tem por objectivo auxiliar o desenvolvimento de um *software* que satisfaça as necessidades do cliente com a maior eficiência exequível.

Actores:

- **Utilizador:** Qualquer indivíduo que utilize a aplicação Android. Desde parceiros turísticos ou turistas a colaboradores da Porto Santo Line.
- **Administrador:** Colaborador da Porto Santo Line com acesso ao painel de administração (servidor).

Seguidamente é apresentado um **diagrama de casos de uso**, que sumariza as interacções entre o utilizador e o sistema.

Posteriormente são listados os **modelos de tarefas**, fornecendo uma melhor compreensão de qual deverá ser o comportamento do sistema perante os pedidos do utilizador.

Considerando a semelhança entre alguns casos de utilização, serão apresentados neste documento os modelos de tarefas dos casos de utilização que possuem comportamento distinto, enquanto os casos com comportamento semelhante serão agregados.

Com base nas especificações dadas pelo cliente, e pelo *site* da própria empresa, foi possível determinar as funcionalidades principais da aplicação. Este diagrama, apresentado na figura seguinte, foi actualizado ao longo do desenvolvimento.

Figura 18 - Diagrama de casos de uso

Para o caso de utilização “Consultar Suplementos”, é importante verificar se existem actualizações na lista de suplementos em vigor e nos preços de cada suplemento. Para a execução destas tarefas, é necessário uma ligação de Internet. Se não existir ligação, uma mensagem de erro deve substituir os preços.

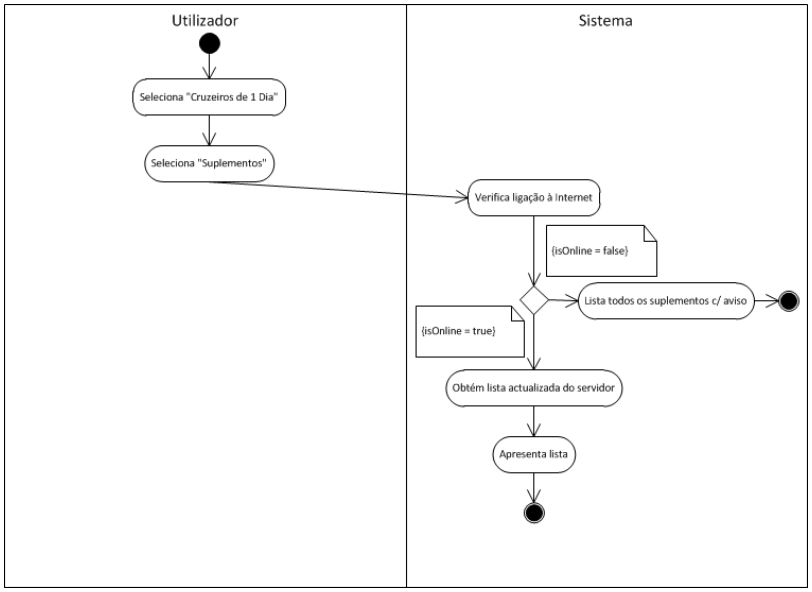


Figura 19 - Modelo de Tarefas "Consultar Suplementos"

Para o caso de utilização “Consultar Horários”, também é importante obter uma ligação à Internet, neste caso para o sistema obter os dados directamente do *site* da empresa. Importante verificar o idioma do utilizador, a fim de obter os dias da semana correctamente.

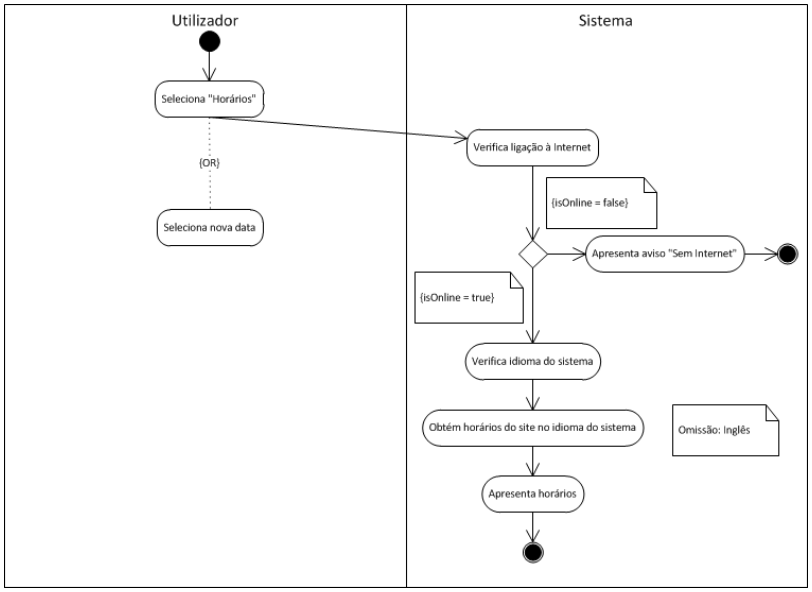


Figura 20 - Modelo de Tarefas "Consultar Horários"

Obter uma rota entre dois pontos num mapa, requer verificar a posição geográfica actual do utilizador. A ligação à Internet é necessária para obter as imagens que constituem o mapa (vista de satélite).

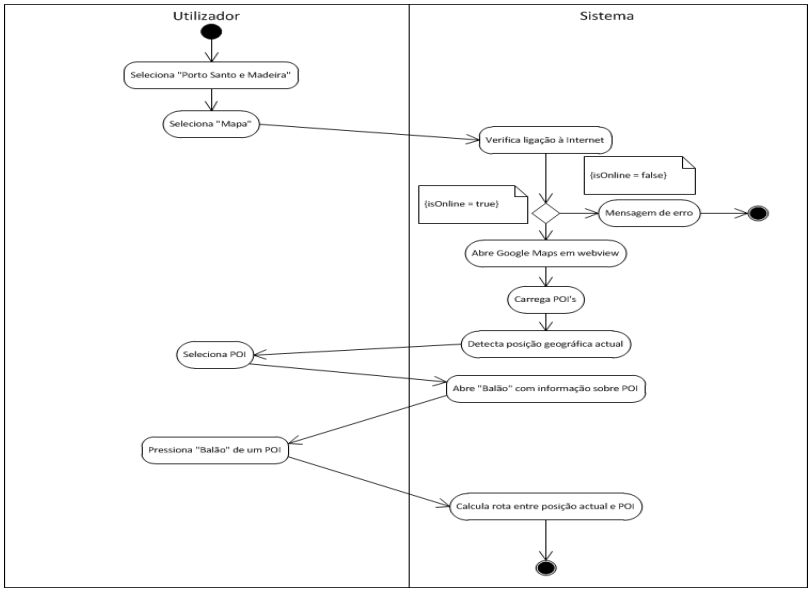


Figura 21 - Modelo de Tarefas "Obter rota"

O caso de utilização “Consultar Promoções”, idêntico ao caso “Consultar Notícias”, obtém os dados directamente do *site* da empresa.

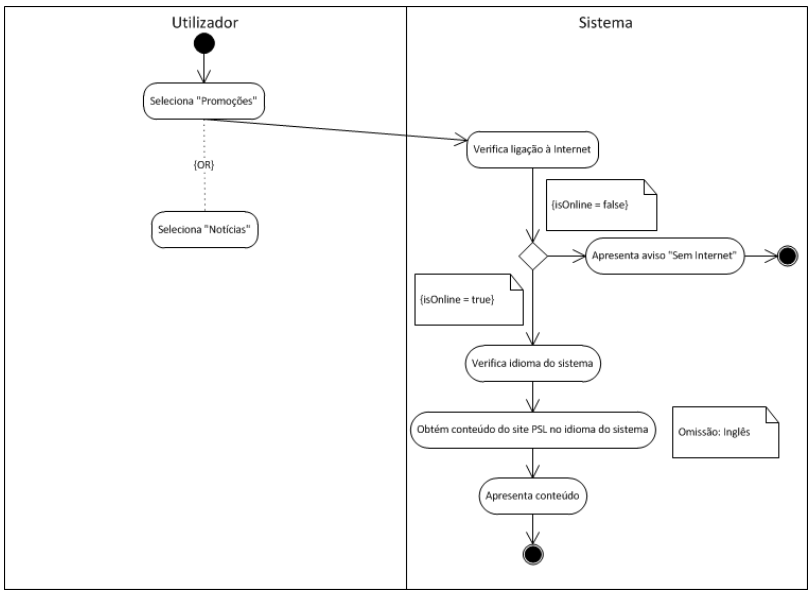


Figura 22 - Modelo de Tarefas "Consultar Promoções"

As tarifas encontram-se disponíveis apenas em PDF no *site* da empresa, pelo que será necessário realizar o *download* dos respectivos ficheiros. A fim de suportar a consulta em modo *offline* e de reduzir o tráfego de dados, os ficheiros devem ser guardados no dispositivo.

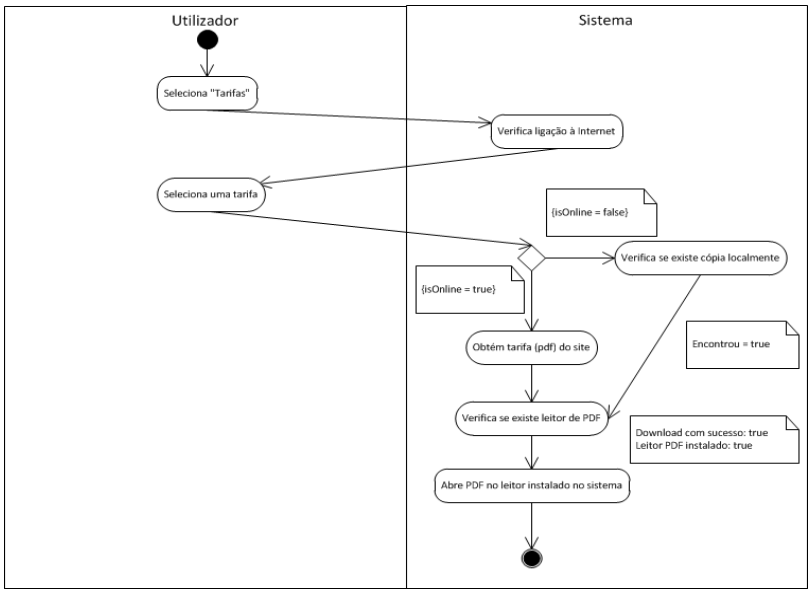


Figura 23 - Modelo de Tarefas "Consultar Tarifas"

3.3. Requisitos

As várias reuniões com os *stakeholders,* foram essenciais para o levantamento dos requisitos.
Para este processo foram realizadas reuniões com a Dra. Maria João, Directora Departamento Comercial e Marketing, com o Dr. Miguel Chaves, Director Departamento Informático e Dr. Manuel Cardoso do Departamento Comercial e Marketing. As reuniões com o orientador do projecto, Professor Dr. Leonel Nóbrega, também contribuíram para o processo, com especial atenção para os requisitos não funcionais.

Como seria de esperar, os requisitos sofreram várias alterações durante o processo de desenvolvimento. Estas alterações ocorreram por vários motivos, como novas decisões administrativas e comerciais, alterações na plataforma Android e também devido ao processo normal de aprendizagem.

Neste documento é apresentada a última versão dos requisitos.

3.3.1. Funcionais

F1. Promover a divulgação do pacote “Cruzeiros de 1 Dia”.

F2. Listar todos os suplementos que podem ser adicionados ao pacote “Cruzeiros de 1 dia”.

F3. A listagem dos suplementos do pacote “Cruzeiros de 1 Dia” deve incluir título, uma descrição, imagem e preço para cada suplemento.

F4. Edição dos preços dos suplementos de F3 por um administrador do sistema.

F5. Fornecer breve informação sobre restaurantes, alojamento e actividades da ilha Porto Santo.

F6. Divulgar promoções em vigor, de acordo com a respectiva secção do *site* oficial da Porto Santo Line.

F7. Anunciar notícias de acordo com a respectiva secção do *site* oficial da Porto Santo Line.

F8. Permitir a consulta dos horários das partidas do navio Lobo Marinho III.

F9. Disponibilizar as tarifas simples de época alta e época baixa.

F10. Disponibilizar as tarifas respectivas aos pacotes especiais.

F11. Listar os serviços disponíveis a bordo do navio Lobo Marinho III.

F12. Os serviços de F13 devem conter título, breve descrição, imagem do serviço e imagem com a localização.

F13. Fornecer descrição geral sobre a ilha Porto Santo.

F14. Fornecer descrição geral sobre a ilha Madeira.

F15. Listar as “Experiências” disponíveis na ilha Porto Santo.

F16. As “Experiências” de F15 devem conter título, descrição, imagem e preço.

F17. Edição dos preços das “Experiências” de F16 por um administrador do sistema.

F18. Disponibilizar um mapa interactivo da ilha Porto Santo.

F19. Disponibilizar um mapa interactivo da ilha Madeira.

F20. Os mapas de F18 e F19 devem incluir os Pontos de Interesse indicados pela empresa Porto Santo Line.

F21. Destacar os Ponto de Interesse de F20 que pertençam ao Grupo Sousa.

F22. Calcular a melhor rota para cada Ponto de Interesse em F20.

F23. Cada Ponto de Interesse de F20 deve possuir um título e opcionalmente uma imagem e uma breve descrição.

F24. Listar os balcões que constam na respectiva secção do site oficial da Porto Santo Line.

F25. Cada balcão de F24 deve ser localizável nos mapas da aplicação.

F26. Os “Armazéns de Carga” de F24 devem possuir um botão que permita ao utilizador entrar em contacto através do número de telefone.

F27. Disponibilizar email, fax e telefone do “Contact Center”.

F28. Para F27 disponibilizar um botão que permita ao utilizador entrar em contacto através do número de telefone.

F29. Todos os endereços de email indicados na aplicação devem responder ao “Tap” do utilizador, abrindo a aplicação de email com o respectivo email do destinatário já preenchido.

F30. O Painel de Administração deve permitir a activação ou desactivação de cada suplemento.

F31. O Painel de Administração deve permitir a activação ou desactivação de cada experiência.

F32. Fornecer secção FAQ com base na respectiva secção do site oficial da Porto Santo Line.

F33. Permitir a partilha nas redes sociais do *link* da página do site oficial PSL associada ao conteúdo que o utilizador encontra-se a visualizar. Exemplo: No ecrã horários, utilizar poderá partilhar o *link* da página “Horários” da página oficial da Porto Santo Line.

F34. A aplicação deve iniciar com a apresentação da empresa Porto Santo Line, incluindo imagem do navio Lobo Marinho III, na forma de “*Splash Screen*”.

F35. A aplicação deverá suportar os seguintes três idiomas: Português, Inglês e Alemão.

3.3.2. Não Funcionais

NF1. [Eficiência] As “Notícias” devem ser recolhidas automaticamente do site oficial da Porto Santo Line.

NF2. [Eficiência] As “Promoções” devem ser recolhidas automaticamente do site oficial da Porto Santo Line.

NF3. [Eficiência] As “Tarifas” devem ser recolhidas automaticamente do site oficial da Porto Santo Line.

NF4. [Eficiência] Os “Horários” devem ser recolhidos automaticamente do site oficial da Porto Santo Line.

NF5. [Implementação]: A aplicação deverá ter como alvo a versão 2.2 do Android;

NF6. [Capacidade] Cada imagem utilizada na aplicação deve ser inferior a 150KB.

NF7. [Desempenho] As imagens utilizadas na aplicação devem ser convertidas para o formato PNG.

NF8. [Eficiência] A aplicação deverá detectar automaticamente qual o idioma a utilizar.

NF9. [Usabilidade] Os botões têm o tamanho mínimo de 43DP.

NF10. [Usabilidade] O separador activo é destacado com o texto em branco e fundo a azul.

NF11. [Usabilidade] Todos os separadores, com a excepção do separador activo, possuem texto azul e fundo castanho.

NF12. [Organização] As cores utilizadas na aplicação devem corresponder à palete de cores utilizadas no site oficial da Porto Santo Line.

NF13. [Organização] O tipo de letra a utilizar na aplicação é *YanoneKaffeesatz*.

NF14. [Fiabilidade] Os preços devem ser sempre verificados com a base de dados.

NF15. [Fiabilidade] Se não existir ligação à Internet, os preços não devem ser apresentados.

NF16. [Organização] As “Tarifas” são publicadas em PDF.

NF17. [Eficiência] A aplicação deve utilizar a aplicação de leitura de ficheiros PDF já instalada no dispositivo.

NF18. [Eficiência] Na ausência de uma aplicação de leitura de ficheiros PDF, a aplicação deve reencaminhar o utilizador para a aplicação Adobe Reader no Google Play.

NF19. [Eficiência] A funcionalidade “Partilhar” deve recorrer às respectivas aplicações já instaladas no dispositivo.

NF20. [Disponibilidade] As “Tarifas” devem ser guardadas no dispositivo, durante a primeira consulta, a fim de serem consultadas em modo *offline.*

NF21. [Capacidade] As “Tarifas” são guardadas preferencialmente no cartão de memória externo.

NF22. [Interoperabilidade] Os mapas da aplicação devem recorrer à API Google Maps.

NF23. [Interoperabilidade] As rotas devem ser obtidas recorrendo à API Google Directions.

NF24. [Usabilidade] O utilizador pode cancelar o cálculo da rota.

NF25. [Usabilidade] Se não existir ligação à Internet o estado "*offline*" deve ser apresentado nos ecrãs que requeiram comunicação com o exterior.

NF26. [Usabilidade] A mensagem "*offline*" de NF25 deve possuir um atalho para as definições *WIFI* do sistema.

NF27. [Desempenho] A aplicação deve recorrer à última posição geográfica conhecida de forma a calcular rotas.

NF28. [Desempenho] Em caso da última posição geográfica ser desconhecida, a aplicação recorre ao GPS, WIFI e GPRS (por esta ordem) para obtenção da posição mais recente.

NF29. [Desempenho] A obtenção da localização geográfica deve decorrer em segundo plano (*thread* separada).

NF30. [Desempenho] O *download* das "Tarifas" deve decorrer em segundo plano (*thread* separada).

NF31. [Desempenho] A consulta dos "Horários" deve decorrer em segundo plano (*thread* separada).

NF32. [Desempenho] A verificação dos preços deve decorrer em segundo plano (*thread* separada).

NF33. [Desempenho] A obtenção das "Notícias" deve decorrer em segundo plano (*thread* separada).

NF34. [Desempenho] A obtenção das "Promoções" deve decorrer em segundo plano (*thread* separada).

NF35. [Eficiência] Inglês é o idioma por omissão.

NF36. [Eficiência] O arranque da aplicação é feito com a apresentação do *splash screen*.

NF37. [Desempenho] A duração da apresentação do *splash screen* deverá ser limitada a 3 segundos.

NF38. [Usabilidade] O *splash screen* é ignorado se o utilizador pressionar o ecrã.

NF39. [Entrega]: O código fonte da aplicação deverá ser comentado e organizado de forma a facilitar a recodificação para iOS.

NF40. [Interoperabilidade] A aplicação deve comunicar com uma base de dados MySQL.

NF41. [Segurança] O Painel de Administração deve possuir um processo de autenticação (*login*).

NF42. [Interoperabilidade] O Painel de Administração deve ser acessível a partir de qualquer sistema operativo.

3.4. Aplicações Relacionadas

Investigar quais as soluções que existem actualmente no mercado, que satisfazem as necessidades do cliente, antes do desenvolvimento da própria solução, contribui para uma melhor compreensão da visão do cliente e dos objectivos que pretende alcançar com a aplicação.
Em cada aplicação seleccionada, foram identificados os aspectos de maior relevância para o cliente. Esta informação tornou-se útil para a criação dos *mockups* e posterior desenvolvimento da interface do utilizador.

Importante mencionar que o cliente final, Porto Santo Line, não possui qualquer aplicação móvel, o que aumenta a importância de analisar as aplicações dos parceiros e concorrentes.

Seguidamente são apresentadas algumas das análises realizadas durante esta fase.

Para o caso de utilização “Obter Rota”, o cliente mostrou especial atenção pela forma como a aplicação Rali Vinho Madeira apresenta trajectos entre dois pontos, figura seguinte.

Figura 24 - Apresentação de uma rota entre dois pontos

A navegação por separadores permite categorizar a informação de forma rápida. Embora este exemplo tenha sido seguido na primeira fase do desenvolvimento deste

projecto, mais tarde tornou-se obsoleto, sobretudo com a aceitação da nova interface gráfica do Android 3.0, na qual os separadores como apresentados na figura 16, deixam de fazer parte das boas práticas de design estabelecidas pela Google. Esta transição (para a nova interface 3.0), foi um ponto importante durante o desenvolvimento, que exigiu esforço adicional de estudo e desenvolvimento. Este tópico será aprofundado na secção 4 deste documento.

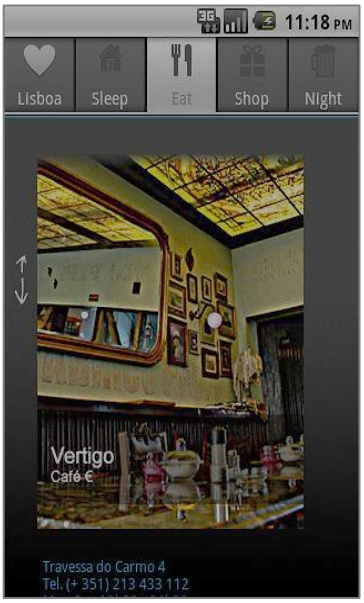


Figura 25 - Painel de navegação baseado em separadores

Conteúdo a apresentar é consistido essencialmente por texto e imagens. Não é necessário áudio nem vídeo. A figura seguinte capturou a atenção do cliente, devido ao modo como o conteúdo é apresentado.

Figura 26 - Disponibilização de conteúdo

Os menus expansíveis, como representados na figura seguinte, permitem a disponibilização de conteúdo sem mudar de ecrã, evitando transições e animações desnecessárias, que apenas dificultam o acesso à informação.

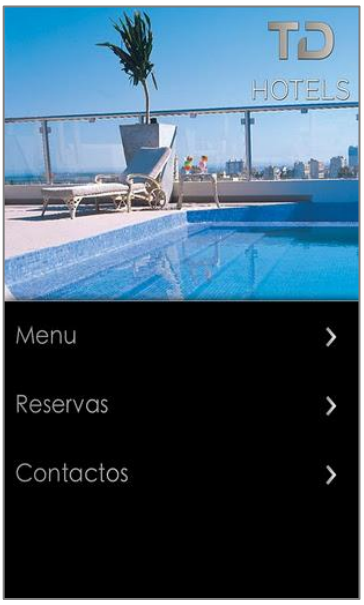


Figura 27 - Menus expansíveis

3.5. Wireframes

Foram criados *wireframes* com o objectivo de definir o fluxo da aplicação e o agrupamento de dados. Como esperado, ocorrem várias modificações durante esta fase.

Importante salientar o envolvimento do cliente na concepção dos *wireframes* e dos protótipos da interface gráfica. A primeira versão fora baseada nos requisitos do cliente, nos casos de utilização e em aplicações existentes. Durante as reuniões com o cliente, os *wireframes* foram apresentados e as modificações requeridas pelo cliente anotadas para implementação, que posteriormente seriam alvo de nova análise por parte da Porto Santo Line.

No momento da elaboração dos *wireframes*, a interface gráfica Android 2.0 continuava como a interface dominante e portanto a concepção foi baseada nessa versão. Na secção seguinte deste documento, “Protótipo da Interface do Utilizador”, é discutida a necessidade que surgiu de abandonar o *layout* adoptado nos *wireframes,* a fim de adaptar o projecto à nova versão da interface gráfica da plataforma Android. Apesar desta enorme mudança, a distribuição do conteúdo e o

modelo da agregação dos dados foram, parcialmente, utilizados na nova interface, pelo que continua a ser relevante a apresentação neste documento dos *wireframes*.

Embora os ecrãs seguintes não correspondam à versão final da interface com o utilizador, são importantes para compreender a evolução que a aplicação sofreu ao longo do seu desenvolvimento, adaptando-se a novos requisitos do cliente, assim como a novas tecnologias e metodologias entretanto lançadas e adoptadas pelo mercado.

Ecrã "Principal":

Tendo como objectivo criar uma interface simples sem sobrecarregar o utilizador com demasiadas opções, o menu principal (figura seguinte) da aplicação fornece acesso directo às principais funcionalidades. A concepção deste ecrã contribuiu de certo modo, para definir a distribuição do conteúdo. Especial atenção à altura mínima de cada botão, a fim de permitir uma fácil selecção.

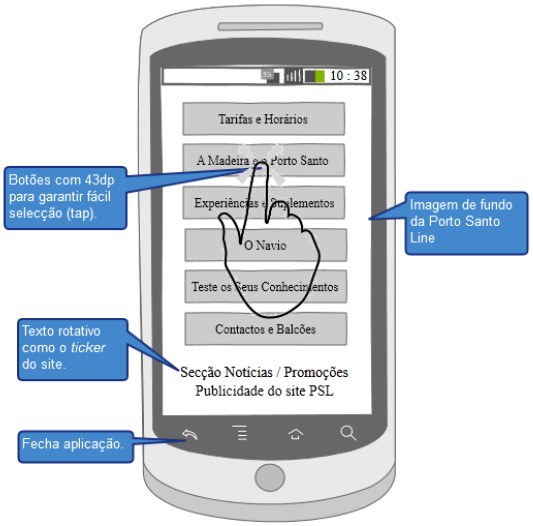


Figura 28 - Ecrã "Principal" (*Home screen*)

Ecrã "Tarifas":

- Objectivo: Permitir a consulta das tarifas de viagem.

Inicialmente agrupado com o separador "Horários" por decisão comercial, este ecrã (figura 27) acabaria por ser totalmente redesenhado na nova interface gráfica. Contudo, a ideia de usar menus expansíveis para apresentar conteúdo extenso, foi aproveitada em outro ecrã.

Figura 29 - Ecrã "Tarifas"

Ecrã "Horários":

- Objectivo: Disponibilizar a tabela dos horários de partida do navio.

Recorrendo à data do sistema, o ecrã inclui imediatamente os horários para o mês corrente, requerendo apenas interacção caso o utilizador permita consultar outro mês. Este ecrã, figura 28, corresponde à página mais consultada do site da empresa.

Figura 30 – Ecrã "Horários"

Ecrã “Ilhas”:

- Objectivo: Fornecer informação turística do arquipélago, enquanto destaca os produtos e serviços do Grupo Sousa, ao qual pertence a empresa Porto Santo Line.

Conteúdo apresentado na horizontal (como indicado na figura seguinte), devido a utilização de um mapa interactivo para acesso a outros ecrãs. Resultou em problemas de interacção e foi redesenhado na versão final.

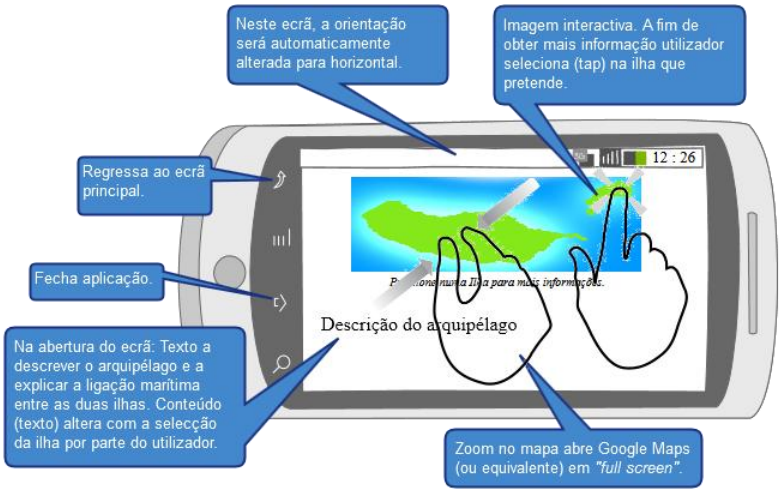


Figura 31 - Ecrã "Ilhas"

Ecrã “Experiências e Suplementos”:

- Objectivo: Promover produtos e serviços turísticos da Porto Santo Line.

Conteúdo apresentado em forma de lista, como indicado seguidamente, e separado visualmente com a ajuda do tamanho da imagem e do título destacado de cada produto.

Figura 32 - Ecrã "Experiências e Suplementos"

Ecrã “O Navio”:

- Objectivo: Informar os serviços disponíveis a bordo do navio.

Reutiliza o *layout* do ecrã anterior. Apesar de existir apenas um grupo de informação a apresentar, foi mantido o separador “Serviços” a fim de manter a coerência na navegação entre ecrãs.

Figura 33 - Ecrã "O Navio"

Ecrã “Contactos e Balcões”:

- Objectivo: Fornecer meios de contactar directamente os balcões.

Reutilizando novamente o mesmo *layout* dos anteriores ecrãs, o utilizador familiariza-se mais facilmente com a interface. Neste ecrã, figura 32, é importante a separação dos balcões por localização. Para separar as acções “Telefonar” e “Ver no mapa”, foi planeada a utilização de “menus de contexto”, que surgiriam sempre que o utilizador realizasse um toque prolongado num dos balcões.

Figura 34 - Ecrã "Contactos e Balcões"

Ecrã “Definições”:

- Objectivo: Permitir a persistência das opções de utilização.

Para facilitar a integração da aplicação na plataforma Android, é utilizado o mesmo *layout* que o ecrã das definições do sistema operativo (versão 2.0).

Figura 35 - Ecrã "Definições"

3.6. Protótipo da Interface do Utilizador

Durante a execução desta fase, a Google lançou novas versões do SDK, que alteraram a forma como os utilizadores interagem com as aplicações Android. A alteração com maior impacto foi a nova interface gráfica 4.0, discutida no capítulo 2, que tornou-se o *design* dominante das principais aplicações no mercado Android. Este novo *design* exigiu o redesenho da interface do utilizador de várias aplicações populares, incluindo as aplicações Gmail, Facebook, Twitter e TripAdvisor.

Com os *wireframes* prontos e aprovados pelo cliente e com o protótipo da interface do utilizador em desenvolvimento, era necessário decidir se a aplicação Porto Santo Line seria lançada com base no *design* 2.0 do Android, ou redesenhar a interface adaptando a aplicação ao novo e popular *design* 4.0 (ICS). Seguindo os princípios da metodologia *Agile*, foi decido, juntamente com o cliente, aceitar a mudança e desenvolver o protótipo da interface do utilizador, com base nas novas directrizes de usabilidade publicadas pela Google [19].

Protótipo recorrendo ao *Android Development Tools* (ADT) para Eclipse:

A criação de um protótipo só será útil se for possível simular a interacção do utilizador com o sistema. É importante, nesta fase, conseguir observar como os utilizadores interagem com a interface, identificar e corrigir erros de usabilidade e fornecer um meio do cliente (e utilizadores) navegarem pelos vários ecrãs da aplicação, verificando onde e como o conteúdo e funcionalidades são disponibilizadas. Este processo pode ser executado recorrendo a HTML, Flash e até PowerPoint. Para recuperar o tempo despendido com o novo *design* e tendo em conta a relativa facilidade de utilização, foi utilizado o editor de interfaces de utilizador incluído no ADT. Deste modo não só foi economizado tempo, pois o protótipo foi facilmente convertido para interface real, como permitiu uma maior eficácia nos testes de usabilidade.

O protótipo e a sua evolução:

Com o principal propósito de ser submetido a análises detalhadas e testes de usabilidade, o protótipo sofreu várias alterações, evoluindo até a sua versão "*final*" e posteriormente convertido para a interface real e completamente funcional da aplicação.

Seguidamente são apresentados a evolução de alguns ecrãs, com destaque para o menu principal, pela grande quantidade de modificações realizadas neste ecrã.

Ecrã “Principal:

Figura 36 - Ecrã "Principal" v1

Figura 37 - Ecrã "Principal" v2

Figura 38 - Ecrã "Principal" v3

Figura 39 - Ecrã "Principal" v4

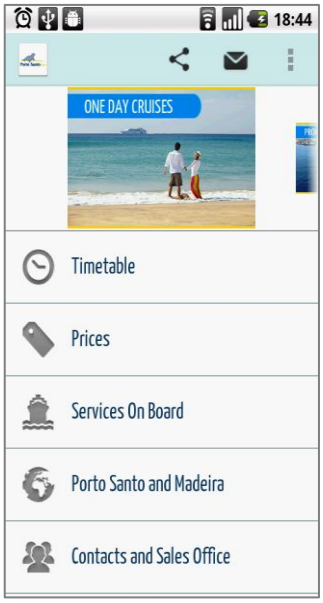


Figura 40 - Ecrã "Principal" v5

Entre a versão 4 e 5, foi tomada a decisão de redesenhar toda a interface do utilizador, aplicando o novo *design* da plataforma Android, com destaque para cores mais claras e adição da “*action bar*”.

Em termos de organização de conteúdo, novas decisões comerciais por parte do cliente, implicou reorganizar o conteúdo já existente e adicionar novas secções, como a zona de destaque para Cruzeiros de 1 Dia e Promoções.

Evolução dos restantes ecrãs:

Figura 41 - "Experiências" (antes)

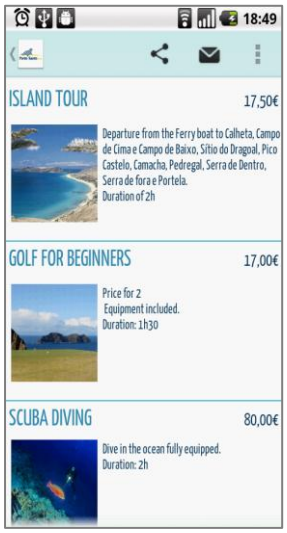


Figura 42 – “Experiências" (depois)

Figura 43 - "Horários" (antes)

Figura 44 - "Horários" (depois)

Figura 45 - "Madeira" (antes)

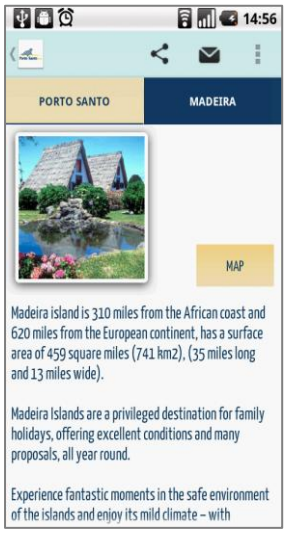


Figura 46 - "Madeira" (depois)

Figura 47 - "Balcões" (antes)

Figura 48 - "Balcões" (depois)

Novamente as alterações no protótipo são perceptíveis, resultado de sucessivas análises em conjunto com o cliente e testes de usabilidade realizados com pequenos grupos de utilizadores, como aprofundado no capítulo 5 deste documento.

3.7. Arquitectura Física do Sistema

Antes de decidir a real arquitectura a seguir, é necessário ter uma ideia geral dos vários sistemas envolvidos no projecto.

Neste projecto, tratando-se essencialmente de uma aplicação móvel com *backoffice* (doravante denominado Painel de Administração), é indispensável uma comunicação com o servidor / base de dados para obtenção de conteúdo a apresentar na aplicação móvel. Este conteúdo pode ser alterado por um utilizador da Porto Santo Line (com privilégios de administrador), através do acesso ao Painel de Administração (PHP/HTML). A aplicação móvel recorre também ao já existente *site* da empresa para a obtenção de dados actualizados. Isto diminui a carga de trabalho na empresa, não alterando o *workflow* dos colaboradores durante o processo de actualização de promoções, horários e notícias. A aplicação Android comunica ainda com a API do *Google Maps*, via JSON, para o cálculo de rotas e obtenção do mapa com vista satélite. Como funcionalidade típica da WEB 2.0, também há comunicação com as aplicações sociais que o utilizador possua instaladas no dispositivo. Na figura seguinte é apresentada a arquitectura física do sistema.

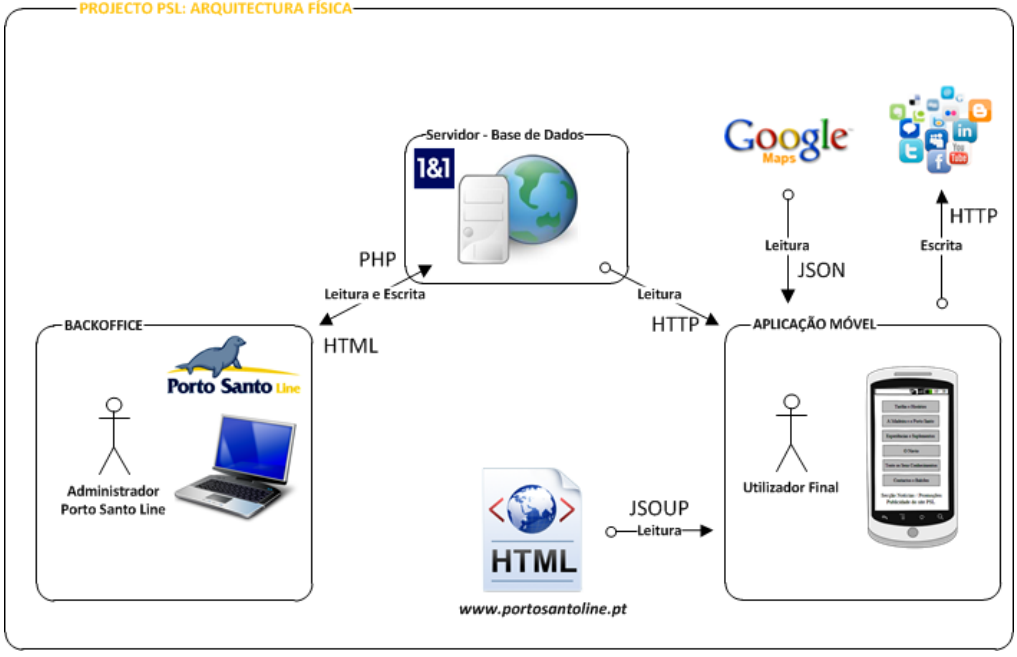


Figura 49 - Arquitectura Física (Geral)

3.8. Arquitectura - Modelo 3 Camadas

Com base no modelo de negócio e necessidades do cliente, é necessária uma arquitectura que permita a evolução do sistema a nível de estrutura, requisitos e funcionalidades, suportando e implementando funções de negócio próprias da empresa. Para atingir este nível de flexibilidade, o modelo 3 camadas (figura 48) apresenta-se como a melhor escolha, pois permite que a alteração numa camada não interfira com as restantes, potencia a separação das responsabilidades e facilita a manutenção do código [20].

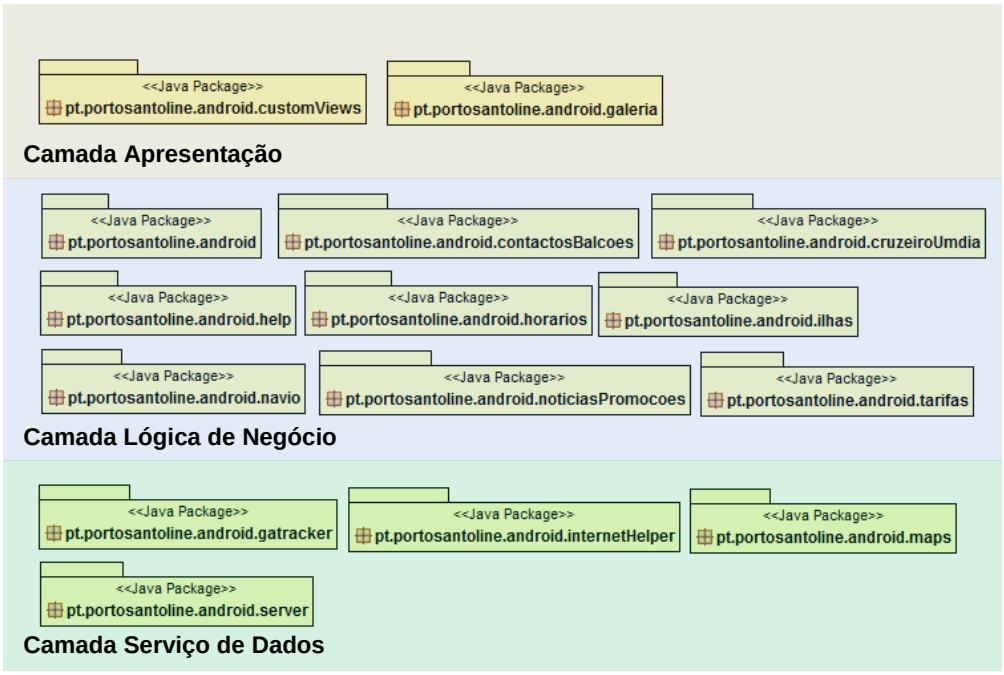


Figura 50 - Visão geral das camadas da aplicação Android

3.8.1. Camada Apresentação

A interacção com o utilizador é realizada nesta camada. É responsável por apresentar conteúdo e validar *inputs*.

Estão incluídas nesta camada as *interfaces* do utilizador:

- Da aplicação Android.
- Do Painel de Administração (*Backoffice*).

Aplicação Android:

A *interface* deve ser o mais constante possível, reutilizando o maior número de ecrãs possíveis e desenvolvendo código flexível. Esta abordagem simplifica a utilização da aplicação, pois o utilizador adapta-se com uma maior facilidade, e em simultâneo aproveita as capacidades de *cache* do sistema operativo, diminuindo assim a utilização dos recursos do dispositivo.

Por existirem ecrãs de diferentes tamanhos e resoluções, é importante conceber *interfaces* gráficas o mais independentes possíveis do tipo de ecrã. A utilização da unidade "*dp*" (*density-independent pixels*) é portanto imperativa para a especificação de dimensões dos objectos gráficos.

Futuramente no momento que forem necessárias alterações na apresentação ou organização da informação, é possível restringir as modificações nesta camada sem alterar as restantes.

Painel de Administração:

Como a aplicação recolhe informação actualizada directamente do *site* da empresa, o Painel de Administração é apenas para o conteúdo que não é possível actualizar automaticamente (preços e activação de suplementos) e com actualizações pouco frequentes (aproximadamente 4 acessos por mês). Isto implica que a *interface* será pouco utilizada, mas quando em uso, deverá permitir facilmente e em pouco tempo realizar as tarefas de actualização. Com este objectivo, a *interface* deverá ser o mais simples possível, possuindo apenas os elementos gráficos estritamente necessários para a execução das tarefas. Por ser nesta camada que os dados são inseridos por parte do utilizador, a *interface* deverá ser responsável pela verificação dos *inputs*, como por exemplo, controlar a inserção de preços *null*.

Esta camada inclui também todos os ficheiros XML responsáveis por definir o *layout* da interface do utilizador.

3.8.2. Camada Lógica de Negócio

As regras de negócio são implementadas nesta camada [20]. São definidas as funcionalidades da aplicação. Todos os dados nesta camada são voláteis, requerendo acesso à camada de Serviços de Dados para o armazenamento dos dados.

A fim de manter um maior nível de flexibilidade e capacidade de expansão, a implementação das funcionalidades é realizada por módulos (*packages* em Java) baseada na regra de negócio que implementa. Exemplificando, o código responsável por disponibilizar as tarifas, encontra-se separado do código que disponibiliza os horários. Porém, como em ambas as regras de negócio é necessária uma ligação à Internet, existe um outro módulo, utilizado em ambos os módulos anteriores, que é responsável por verificar e reportar o estado da ligação à rede. Esta abordagem permite que a mudança de uma regra de negócio não afecte as restantes. Permite também a adição de novas lógicas sem alterar as que já existem. Independente do "dispositivo cliente", *tablet*, *smartphones* ou até televisão, a lógica de negócio é sempre a mesma, mudará apenas a camada de apresentação.

Nesta camada encontra-se o *core* da aplicação, que incluí a implementação em java da aplicação Android e PHP/HTML do Painel de Administração.

A camada lógica recebe os pedidos da camada apresentação, processa os pedidos com base nas regras de negócio e se necessário solicita acesso ao repositório de informação, através da camada serviços de dados.

3.8.3. Camada Serviços de Dados

Importante salientar que esta camada não representa os servidores nem os dados, mas sim os serviços de acesso aos dados [20].
Esta camada separa as operações de acesso aos dados da camada de negócio e apresentação. Alterar o modo de acesso à base de dados não afecta as restantes camadas.

Durante o desenvolvimento deste projecto, a Google alterou o método de aceder ao seu repositório de mapas, que contém os dados necessários para calcular a rota entre a posição do utilizador e um POI no mapa. Mais concretamente foi descontinuada a troca de informações no formato KML, passando o formato JSON a ser o método favorito para a troca de dados. Pelo facto do projecto basear-se no modelo 3 camadas, esta mudança externa afectou apenas a camada serviços de dados, não alterando a camada apresentação nem a camada lógica de negócio.

Na aplicação Android, todo o código java responsável por recolher dados do *site* e da base de dados, encontra-se num *package* separado, com classes próprias e genéricas.

No Painel de Administração, esta camada contém os ficheiros PHP com as *strings MySql* que manipulam a base de dados.

3.8.4. Diagrama de Classes

O diagrama de classes apresentado na figura seguinte, é respectivo apenas à um caso de utilização e encontra-se disponível em anexo para melhor leitura.
O diagrama cumpre com a especificação UML 2.0 da OMG [21].

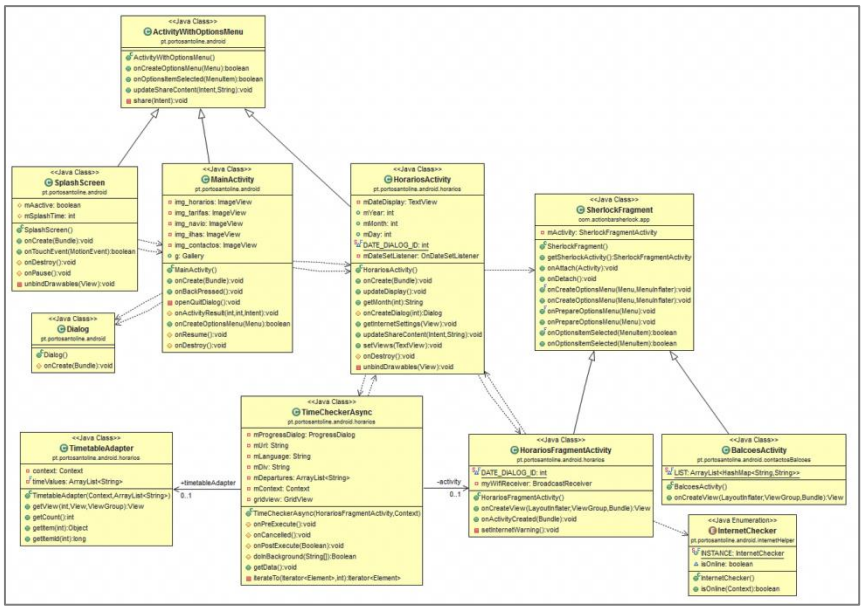


Figura 51 - Diagrama de Classes para a funcionalidade " Consultar Horários"

(formato A4 em anexo)

Com a ajuda do diagrama, é possível verificar as relações entre as várias classes envolvidas na implementação do caso de utilização “Consultar Horários”. É importante ter em conta que algumas relações são impostas pelo sistema Android, como por exemplo a relação entre a superclasse “*SherlockFragment*” e a sua subclasse “*HorariosFragmentActivity*”, devido à utilização do pacote de compatibilidade, que permite a utilização da “*action bar”* em sistemas Android 1.5+.
Esta subclasse também comunica com a classe responsável pelo estado da ligação à Internet, “*InternetChecker*”, que é implementada segundo o padrão de

desenho *Singleton,* a fim de manter a coerência do estado da ligação em todos os ecrãs da aplicação.

3.9. Resumo

A metodologia a utilizar num desenvolvimento de *software*, depende imenso do tipo de projecto que está em causa e da equipa que irá conceber e executar o plano. Seja qual for o método optado, a comunicação entre os elementos da equipa e a organização são factores determinantes para o sucesso ou insucesso do projecto e o cliente é parte central desta equipa. No desenvolvimento do projecto aqui discutido, o cliente foi um elemento chave em todos os aspectos. O método "*Agile"* destaca exactamente a importância de manter uma comunicação constante com o cliente, mas também salienta a grande quantidade de mudanças que ocorrem durante o desenvolvimento de *software*. A mudança é inevitável, haverá sempre novos requisitos e alterações a implementar. Porém, as mudanças na arquitectura do *software* devem ser evitadas, realizando a escolha mais apropriada para o caso.

A escolha da arquitectura para um projecto, envolve um conjunto de decisões estratégicas que afectarão o *software* em todo o seu ciclo de vida. É importante realizar uma análise ao modelo de negócio do cliente, realizar o levantamento dos requisitos, estudar os casos de utilização e tentar ao máximo prever a evolução do programa. Uma mudança na arquitectura poderá implicar grandes e dispendiosas alterações no produto. O modelo 3 camadas provou ser uma excelente arquitectura para este projecto. Permitiu responder às necessidades do cliente, acolhendo as mudanças de requisitos e funcionais exigidas durante o processo de desenvolvimento. Cada *stakeholder* possui os seus interesses e pontos de vista do *software*, um gestor de projecto encontra-se mais preocupado com análises de risco, qualidade e distribuição de tarefas, enquanto o utilizador centra-se mais nas funcionalidades e o cliente em orçamentos e prazos. Todos estes factores deverão ser tomados em consideração para a escolha da arquitectura do *software.*

A criação de *wireframes* e protótipos da interface do utilizador, fornecem informações valiosíssimas não só para a concepção do ambiente gráfico da aplicação, como também para o desenvolvimento em concreto do *software*. O protótipo é extremamente útil na detecção de erros de concepção em termos de usabilidade, mas também é uma poderosa ferramenta de comunicação com o cliente. Se uma imagem vale por mil palavras, um protótipo vale por mil casos de utilização.

4. IMPLEMENTAÇÃO

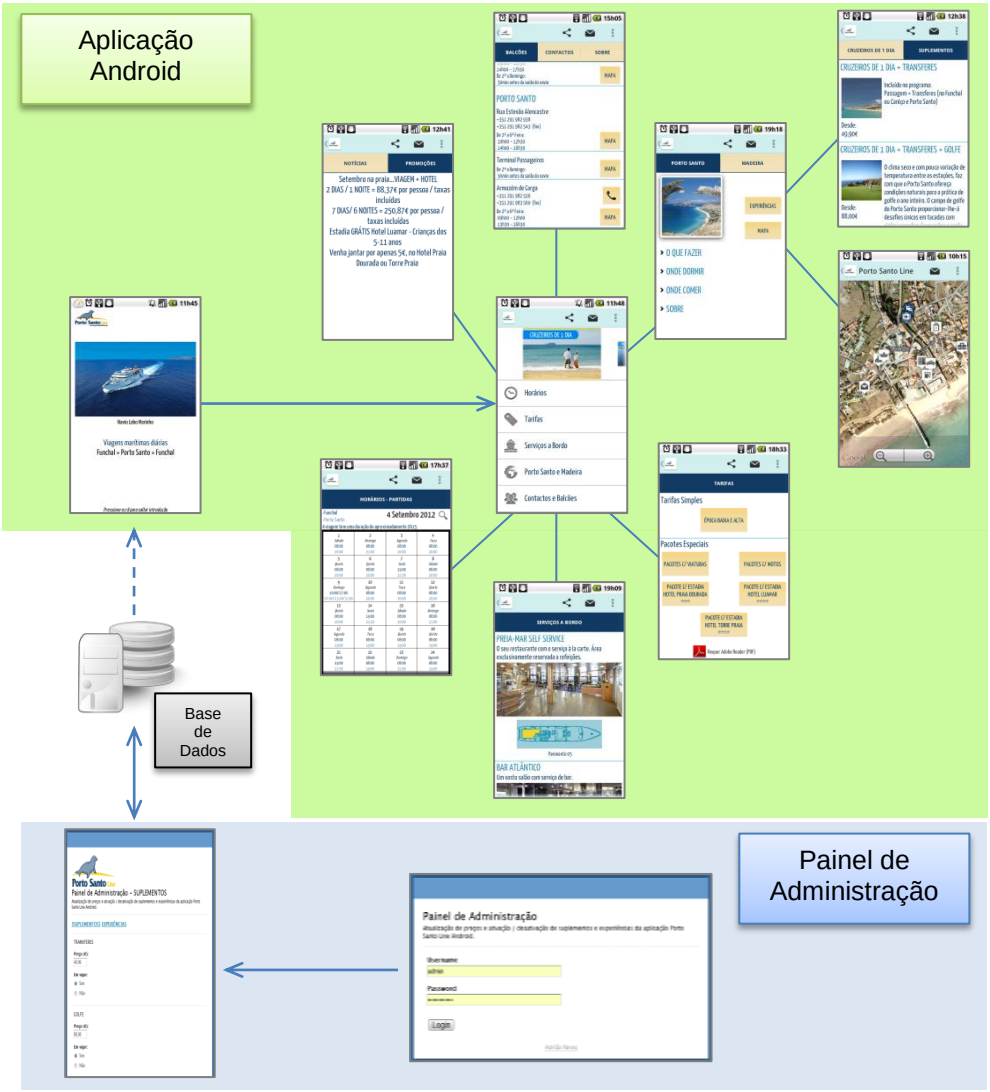
"Measuring programming progress by lines of code is like measuring aircraft building progress by weight."

(Bill Gates)

4.1. Interface do Utilizador

Seguidamente são discutidos os aspectos mais importantes que influenciaram a concepção das *interfaces* gráficas da aplicação Android e do Painel de Administração. Para melhor compreensão, as imagens das *interfaces* em concreto são apresentadas juntamente com o código na secção 4.2 e 4.3 deste documento.

Em termos muitos gerais, as duas interfaces são compostas por vários ecrãs:



4.1.1. Aplicação Android

Como referido nos capítulos anteriores, a *interface* gráfica das aplicações Android sofreu uma grande alteração durante a sua concepção do projecto, devido ao novo *design* lançado pela Google para a versão 4.0 do seu sistema operativo.

Porém, desenvolver uma aplicação para a versão Android 4.0 implicaria renunciar aproximadamente a 83,30% dos utilizadores desta plataforma (ver *Tabela 2, Cap. 2*). Após análise de outras aplicações no mercado, foi observado a crescente tendência na actualização das *interfaces* para o novo *design*, todavia, mantendo a compatibilidade com as versões anteriores do sistema operativo. A solução utilizada nestes casos, encontrada após uma breve pesquisa, passa pela utilização da biblioteca de compatibilidade lançada pela Google [22], em conjunção com uma biblioteca *open-source* [23] criada especificamente para permitir a utilização do novo *design* em Android 1.5 ou superior.

O desenvolvimento da *interface* seguiu, sempre que possível, as orientações da Google [19], contribuindo para uma *interface* coerente com a plataforma.

No contexto de desenvolvimento para Android, um "ecrã" corresponde ao estado da *interface* num determinado momento, também denominado por "janela" em aplicações *desktop*, e por "página" em aplicações *web.*

A *interface* do utilizador da aplicação Android é apresentada juntamente com a explicação de cada funcionalidade, na secção 4.2.

4.1.2. Painel de Administração

A *interface* do Painel de Administração disponibiliza os elementos estritamente necessários para a execução das tarefas para que foi concebido, tornando o processo de actualização dos suplementos e experiências o mais rápido e simples possível.

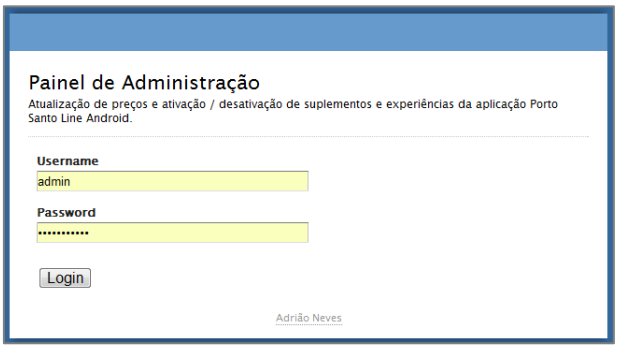


Figura 52 - *Login* no Painel de Administração

Recorrendo às funcionalidades do *browser,* os dados de *login* podem ser memorizados, acelerando o processo.

Figura 53 - Edição dos Suplementos

Os suplementos e experiências são listados verticalmente e com ordem de tabulação, a fim de possibilitar uma rápida inserção de dados sem esforço em identificar a posição de cada item.

4.2. Funcionalidades da Aplicação Android

No sistema Android, as *interfaces* podem ser criadas recorrendo unicamente a XML , a Java ou com uma mistura de ambos. Neste projecto, devido a necessidade de ser necessário alterar o *layout* em tempo de execução, optou-se por utilizar XML para o *layout* base da aplicação e Java para o comportamento associado à *interface*, como ocultar botões ou secções completas com base nas informações recebidas do servidor.

4.2.1. Ecrã de Introdução

A fim de cumprir com o requisito funcional 34 (RF34), durante o arranque da aplicação, é apresentado um ecrã com uma breve descrição da principal função da empresa - transporte marítimo entre Porto Santo e Madeira. Este ecrã, figura seguinte, é apresentado durante 3 segundos (RNF37) ou pode ser ignorado premindo o ecrã, abrindo imediatamente o ecrã principal (RNF38).

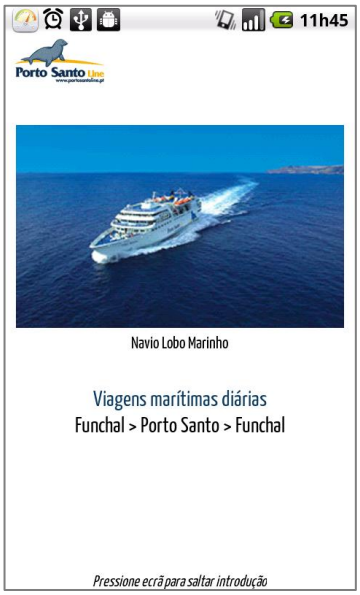


Figura 54 - Ecrã de introdução

4.2.2. Menu Principal

Interface:

Apresentado após o ecrã de introdução, o ecrã principal (figura 53) da aplicação é dividido em três secções:

1. *Actionbar.*
2. Área de destaque.
3. Menu.

A *actionbar* (1) fornece acesso directo às funcionalidades “Partilhar”, “Consultar notícias” e, através do botão *overflow,* aos ecrãs “Ajuda” e “Sobre”.

A área de destaque (2) consiste numa galeria de imagens, com navegação horizontal, fornecendo acesso às funcionalidades “Cruzeiros de 1 Dia” e “Consultar Promoções”. É visível uma pequena secção do botão seguinte ou anterior, a fim de indicar ao utilizador a existência de mais opções.

Os restantes ecrãs são acessíveis através do menu (3) em forma de lista com navegação vertical. Cada item do menu tem a altura recomendada pela Google para um botão, i.e., 43dp. Este valor é o mais indicado para uma selecção fácil, sem ocupar demasiado espaço do ecrã [19].

Figura 55 - Ecrã principal

O ícone da aplicação na *actionbar* (1), não tem qualquer acção neste ecrã, porém nos seguintes será adicionada uma seta e passará a funcionar como botão para retroceder para o ecrã anterior.

Para a maioria dos dispositivos com ecrãs *mdpi (medium dots per inch)*, não será necessário realizar *scroll* vertical.

XML:

Botões, imagens, galerias, textos e outros elementos, são declarados em XML, recorrendo ao SDK do Android. Na linha 27 da figura seguinte, são definidos os parâmetros da galeria de imagens, que constitui a área de destaque apresentada anteriormente.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:asg="http://schemas.pt.portosantoline.android.galeria"
4      android:id="@+id/mainView"
5      android:layout_width="wrap_content"
6      android:layout_height="fill_parent">
7
8      <!-- Body -->
9
10     <RelativeLayout
11         android:layout_width="fill_parent"
12         android:layout_height="wrap_content"
13         android:layout_gravity="top"
14         android:background="@color/branco"
15         android:fadeScrollbars="true"
16         android:tileMode="repeat" >
17
18         <ImageView
19             android:id="@+id/dtop"
20             android:layout_width="fill_parent"
21             android:layout_height="wrap_content"
22             android:layout_above="@+id/gallery1"
23             android:scaleType="fitXY"
24             android:src="@drawable/linedivider" />
25
26         <!-- FEATURED -->
27         <pt.portosantoline.android.galeria.AnimatedSizingGallery
28             android:id="@+id/gallery1"
29             android:layout_width="fill_parent"
30             android:layout_height="wrap_content"
31             android:alwaysDrawnWithCache="true"
32             android:animationCache="true"
33             android:persistentDrawingCache="all"
34             android:spacing="56dp"
35             android:splitMotionEvents="true" />
```

Figura 56 - Parte do XML que define o *layout* do ecrã principal

Código:

A implementação deste ecrã, em termos de funcionalidades, consiste essencialmente em fornecer modos de acesso aos restantes ecrãs, através da galeria de imagens e de botões. Tendo em consideração o tempo que é necessário para obter um sinal GPS, é realizado um pedido de actualização da posição do utilizador neste ecrã, a fim de acelerar o cálculo das rotas no mapa. Este pedido é processado numa tarefa diferente da responsável pela aplicação, não afectando directamente o desempenho ou fluidez nas transições entre ecrãs.

Na figura seguinte é possível verificar a aplicação do *layout* xml descrito acima, através do comando *setContentView(R.layout.main)*. A galeria de imagens

definida no *layout* por XML, é preenchida com imagens fornecidas por um *ImageAdapter,* definido noutra classe.

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //Set the ActionBar settings
    requestWindowFeature(Window.FEATURE_ACTION_BAR_OVERLAY);
    getSupportActionBar().setDisplayShowTitleEnabled(false);

    //Layout
    setContentView(R.layout.main);

    //The font
    Typeface tf = Typeface.createFromAsset(getAssets(),"fonts/YanoneKaffeesatz-Regular.ttf");

    //Get the galley (featured)
    g = (Gallery) findViewById(R.id.gallery1);
    ImageAdapter a = new ImageAdapter(this);
    g.setAdapter(a);
```

Figura 57 – Parte do método onCreate da classe MainActivity

A figura 56 apresenta como é iniciado o processo de localização geográfica do dispositivo. Essencialmente é invocado o método *getLocation(Context, LocationResult)* no objecto *userLocation,* que por sua vez irá proceder com a tarefa de localização. Esta tarefa é discutida com mais pormenor na secção 4.2.12.

```
    //Find the exact current location
    LocationResult locationResult = new LocationResult(){
        @Override
        public void gotLocation(Location location){
        }
    };
    UserLocation userLocation = new UserLocation();
    userLocation.getLocation(this.getApplicationContext(), locationResult);
}
```

Figura 58 - Iniciando o processo de localização geográfica

4.2.3. Cruzeiros de 1 Dia e Suplementos

Interface:

O requisito funcional (RF1), define a necessidade de promover o pacote de viagem “Cruzeiros de 1 Dia”, juntamente com uma lista dos suplementos que podem ser adquiridos em conjunto (RF2).

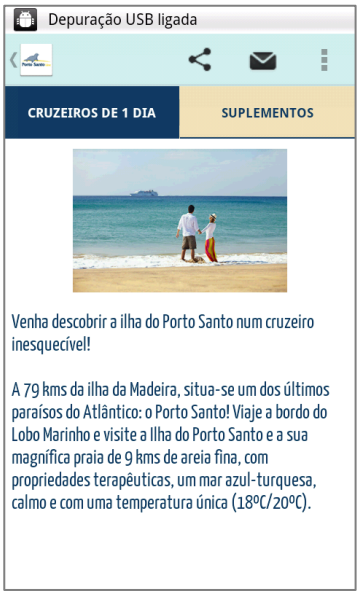


Figura 59 - Cruzeiros de 1 Dia

Figura 60 - Suplementos

Neste ecrã, figuras 57 e 58, é possível observar a implementação dos requisitos não funcionais RNF13 e RNF14, que essencialmente exigem que a *interface* da aplicação respeite as cores e tipo de letra do *site* oficial da empresa (figura seguinte).

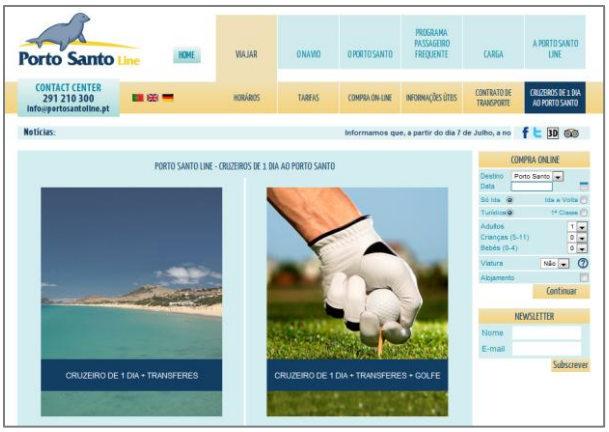


Figura 61 - *Site* oficial Porto Santo Line

XML:

O novo design e as novas orientações da Google [19], abandonam o uso das *interfaces* baseadas em separadores, preferindo o uso de fragmentos. A diferença encontra-se numa maior separação do desenvolvimento de cada fragmento, tanto do XML como do código Java.

Seguidamente, figura 60, é possível verificar a implementação por fragmentos. Cada ecrã passa a incluir um *FrameLayout* onde todo o conteúdo será apresentado.

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

    <RelativeLayout
        android:id="@+id/relativeLayout1"
        android:layout_alignParentBottom="true"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
    </RelativeLayout>


    <FrameLayout
        android:id="@+id/frameLayout1"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_width="match_parent"
        android:layout_above="@+id/relativeLayout1">
    </FrameLayout>

</RelativeLayout>
```

Figura 62 - XML para o layout do ecrã Cruzeiros de 1 Dia

O conteúdo desse *FrameLayout* é definido em outro ficheiro XML, apresentado na figura seguinte, que poderá conter qualquer elemento gráfico, neste caso uma imagem seguida de um texto.

```xml
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/cruzeiro_scrollerView"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >

        <ImageView
            android:id="@+id/imageView1"
            android:layout_width="200dp"
            android:layout_height="150dp"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
            android:padding="2dp"
            android:scaleType="fitCenter"
            android:src="@drawable/cruzeiro" />

        <pt.portosantoline.android.customViews.CustomTextView
            android:id="@+id/cruzeiro_tv2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/imageView1"
            android:gravity="left"
            android:padding="6dp"
            android:text="@string/cruzeiro_desc"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:textColor="@color/azulTextoPSL"
            android:textStyle="normal" />
    </RelativeLayout>

</ScrollView>
```

Figura 63 - XML para o fragmento "Cruzeiro de 1 Dia"

Código:

Existem duas funções importantes neste ecrã:

- Detectar estado da ligação à Internet.
- Actualizar lista dos suplementos se Internet disponível.

O processo de actualização dos preços é realizado numa tarefa separada, de forma análoga ao processo de localização geográfica. Isto permite ao utilizador continuar a utilizar a *interface* enquanto o processo de actualização é

executado em segundo plano. A figura 63 representa parte da classe responsável por lançar uma tarefa dedicada para estabelecer comunicação com o servidor e obter a lista de suplementos actualizada. Estes passos são executados através da interface *ServerInstance* (java) criada para comunicar com o servidor, através do método *getSuplementos().* Após a conclusão da tarefa com o servidor (*onPostExecute*), é iniciado o processo de filtragem dos dados para consequente apresentação.

```
private class GetListTask extends AsyncTask {

    /**
     * Let's make the http request and return the result as a String.
     */
    protected String doInBackground(Object... args) {
        return ServerInterface.getSuplementos();
    }

    /**
     * Parse the String result
     */
    protected void onPostExecute(Object objResult) {
        // check to make sure we're dealing with a string
        if(objResult != null && objResult instanceof String) {
            String result = (String) objResult;

            // this is used to hold the string array, after tokenizing
            ArrayList<String> responseList = new ArrayList<String>();

            // we'll use a string tokenizer, with "," (comma) as the delimiter
            StringTokenizer tk = new StringTokenizer(result, ";");

            // let's build the string array
            while(tk.hasMoreTokens()) {
                responseList.add(tk.nextToken());
            }
            try {
                //Get the TEXTviews
                ArrayList<View> viewsList = new ArrayList<View>();
```

Figura 64 - Processo de actualização é executado em nova tarefa

4.2.4. Promoções e Notícias

Interface:

O ícone da aplicação na *actionbar* (figura seguinte, ponto 1) funciona agora como botão de retroceder.

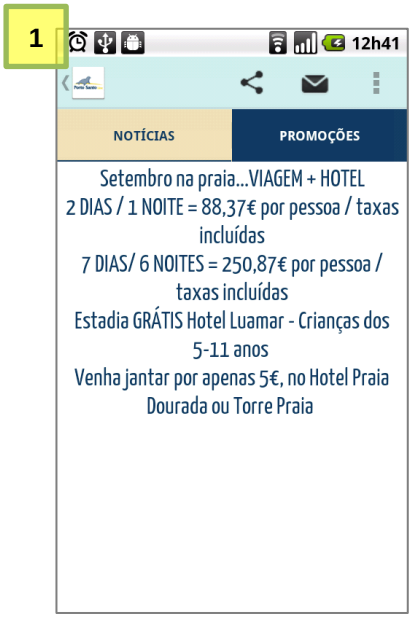


Figura 65 - Promoções

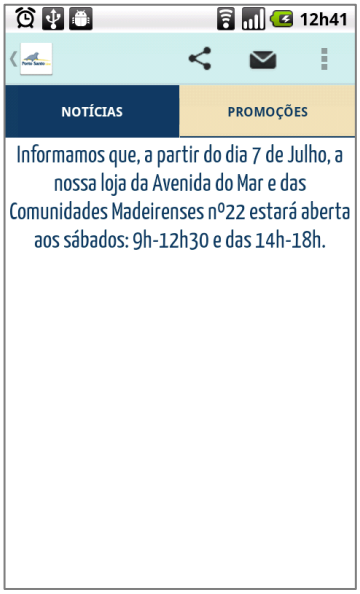


Figura 66 - Notícias

XML:

A fim de aplicar o mesmo tipo de letra utilizada no *site* da empresa, foi necessário redefinir o elemento *TextView*, dando origem ao elemento *CustomTextView.* A figura 66 demonstra como esse elemento é invocado em todos os ecrãs que contenham texto, adicionando o nome do *package* do projecto (*pt.portosantoline.android*), seguido da sua localização (*customViews*) e nome (*CustomTextView*).

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/portosanto_scrollerView"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" >

        <pt.portosantoline.android.customViews.CustomTextView
            android:id="@+id/news_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
            android:clickable="true"
            android:drawablePadding="6dp"
            android:gravity="center"
            android:onClick="getInternetSettings"
            android:padding="1dp"
            android:text="@string/noInformation"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:textColor="@color/azulTextoPSL"
            android:textStyle="bold" />
    </RelativeLayout>

</ScrollView>
```

Figura 67 - *TextView* personalizado com o tipo de letra da Empresa

Código:

Também neste ecrã, é necessária a verificação do estado da ligação à Internet a fim de obter o respectivo conteúdo do *site* da empresa, recorrendo à biblioteca JSOUP. No caso de existir uma ligação activa à Internet, é verificado se o idioma do sistema corresponde a um dos três idiomas suportados pelo *site* da empresa, e se for esse o caso, o código internacional do idioma é concatenado ao URL da página "Notícias" ou "Promoções", dependendo do fragmento activo.

A figura seguinte indica parte do código responsável por obter o idioma do sistema e definir o URL apropriado. Na primeira linha é pedido ao sistema para devolver o idioma em utilização, *Locale.getDegault().getLanguage*, que seguidamente é comparado com os códigos internacionais do Português (PT), Alemão (DE) e Inglês (EN). Se o idioma do sistema não coincidir com nenhum dos códigos, é utilizado o idioma Inglês por omissão.

```
String mSysLang = Locale.getDefault().getLanguage();
if (mSysLang.equalsIgnoreCase("pt") || mSysLang.equalsIgnoreCase("en") || mSysLang.equalsIgnoreCase("de")) {
    mURL = mURL.concat("?lang="+mSysLang);
} else {
    mURL = mURL.concat("?lang="+mLanguage);
}
```

Figura 68 - Detectando idioma do sistema

Com o URL criado segundo a condição acima indicada e recorrendo ao JSOUP, é em primeiro lugar estabelecida uma ligação com o servidor a fim de obter a respectiva página HTML, através da instrução *Jsoup.connect(URL).get().* Da página é seleccionada o conteúdo definido por *mDiv,* e convertido para texto, para seguidamente ser apresentado no *CustomTextView* declarado no XML acima indicado. Como controlo, existe *connectionOk* que é auxilia a verificar se o processo foi executado com sucesso.

```
@Override
protected String doInBackground(String... mUrl) {
    //Fetch and parse HTML with JSOUP
    try {
        Document doc = Jsoup.connect(this.mUrl).get();
        Element link = doc.select(mDiv).first();
        websiteText = link.text();
        connectionOk = true;

    } catch (Exception e) {
        connectionOk = false;
    }
    return websiteText;
}
```

Figura 69 - Fetch ao site da Porto Santo Line

4.2.5. Partilha nas Redes Sociais

A implementação da funcionalidade de partilha nas redes sociais, aproveita o conceito de partilha de informação entre várias aplicações do sistema Android. De forma resumida, permite que a aplicação Porto Santo Line, através do uso de *intents*, envie dados para a aplicação Facebook, desde que ambas instaladas no mesmo dispositivo. Esta técnica permite total independência das API's das várias redes sociais, diminuindo o custo de manutenção da aplicação.

Interface:

A figura seguinte demonstra a utilização do *dialog* da plataforma Android e de um exemplo de aplicações para partilha.

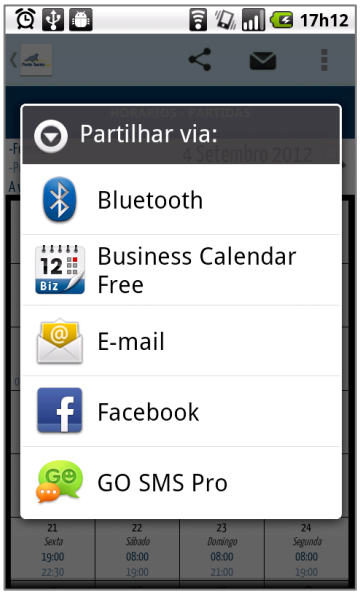


Figura 70 - Partilhar URL dos horários

Código:

Na figura 70 é demonstrado como é criado um *intent* que inclui uma *string* com o URL, neste caso da página Horários, que posteriormente é partilhado através da aplicação escolhida pelo utilizador.

```
private void share(Intent shareIntent) {
    shareIntent.setType("text/plain");
    shareIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_RESET_TASK_IF_NEEDED);
    shareIntent.putExtra(Intent.EXTRA_SUBJECT, getString(R.string.defaultSubjectShare));
    startActivity(Intent.createChooser(shareIntent, getString(R.string.shareWith)));
}
```

Figura 71 - Partilhando dados entre aplicações no sistema Android

4.2.6. Horários

Interface:

A tabela de horários é apresentada de forma a incluir o máximo de informação no menor espaço possível. Em vez de incluir o nome do cais de saída em cada célula, é utilizado as cores como meio de associação, figura 71. Os testes de usabilidade, discutidos no capítulo 5, demonstraram a eficiência desta técnica.

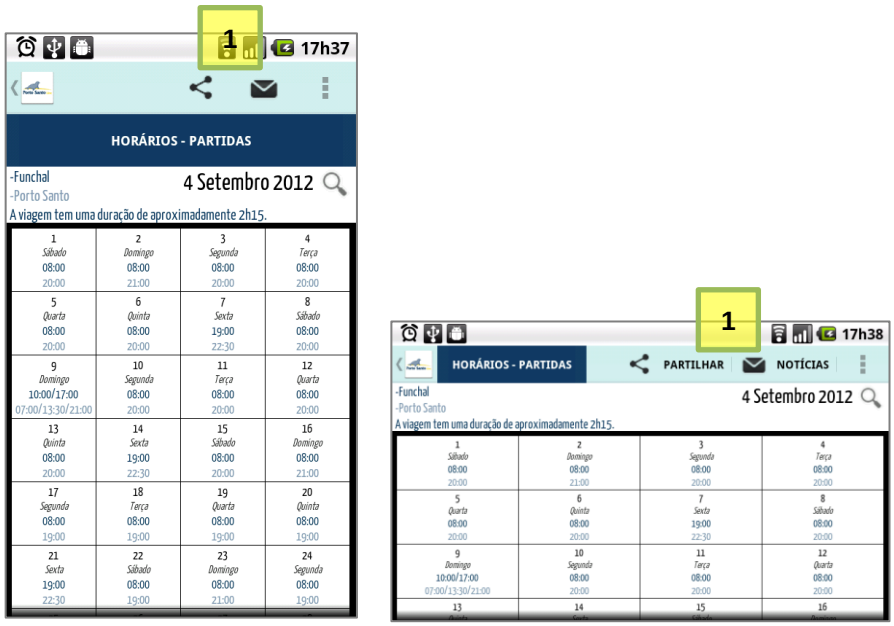


Figura 72 - Ecrã "Horários" em modo vertical e horizontal

Destaque para a *actionbar (1)* na figura 71, que ao detectar que existe mais espaço disponível no ecrã, devido à mudança na orientação do *smartphone,* passa a incluir texto para além dos ícones.

A figura seguinte, mostra o *dialog* no qual o utilizador insere a data que pretende consultar. A figura 73 por sua vez, apresenta o que sucede quando o utilizador selecciona uma das datas no horário (da figura 71). Essencialmente é apresentado com mais detalhes, a hora de saída do navio a partir de cada cais.

Figura 73 - Pesquisa por data

Figura 74 - Apresentação por dia

Código:

De forma análoga aos ecrãs “Notícias” e “Promoções”, os dados relativos à tabela de horários é obtida directamente do *site* da Porto Santo Line, com a detecção prévia do idioma, a fim de apresentar os dias da semana correctamente. Como indicado na figura seguinte, o processo começa por estabelecer ligação à pagina HTML definido por mURL e criado com base no idioma do sistema. Tratando-se de uma tabela HTML, a página é convertida para um *Document* (classe própria da biblioteca JSOUP), que permite manipular todos os elementos da página HTML. Neste caso, o pretendido é obter apenas a tabela com os horários, definida por *mDiv*, e filtrada pelo método *select(mDiv).*

```
public void getData() {
    mDepartures.clear();

    try {
        //Let's talk with the server and ask for the timetable for the selected month and year
        Document doc = Jsoup.connect(mUrl)
                .data("mes", String.format("%02d", ((HorariosActivity)this.activity.getActivity()).mMonth+1))
                //                .data("mes", (Integer.toString( ((HorariosActivity)getActivity()).mMonth+1)
                .data("ano", (Integer.toString( ((HorariosActivity)this.activity.getActivity()).mYear) ))
                .data("lang", "en") //LANGUAGE!
                .userAgent("Mozilla")
                .post();

        //We just want the timetable
        Element table = doc.select(mDiv).first();
        Iterator<Element> rawIterator = table.select("td").iterator();
        //We don't want to write ite.next() 9 times, do we?
        Iterator<Element> ite = iterateTo(rawIterator, 9);

        //Fetch all the rows, each row we save as line
        while (ite.hasNext()) {
            mDepartures.add(ite.next().text());
        }
        ite = null;
    } catch (Exception e) {
    }
}
```

Figura 75 – Fetching com a ajuda do JSOUP

4.2.7. Tarifas

Interface:

Mantendo o mesmo *layout* do *site* da empresa, as tarifas são disponibilizadas através de botões agrupados pelo tipo de pacote turístico, figura 75, e apresentadas através da aplicação PDF disponível no sistema, figura 76.

Figura 76 - Tarifas

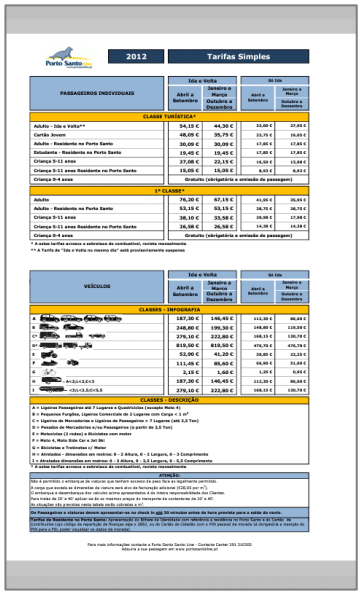


Figura 77 - Visualização de uma tarifa (PDF)

Código:

As principais funções deste módulo:

- Verificar ligação à Internet.
- Verificar idioma do sistema.
- Obter os ficheiros PDF no idioma correcto, através do *site* da empresa.
- Guardar os ficheiros no cartão de memória para consulta em modo *offline.*
- Verificar se existe aplicação capaz de abrir ficheiros PDF.
- Através da aplicação Google Play, reencaminhar utilizador para o *download* da aplicação Adobe Reader, caso seja necessário.

Tratando-se de ficheiros e não de texto, a obtenção das tarifas é realizada através do protocolo *HTTP*. Como apresentado na figura seguinte, após estabelecido o canal de comunicação entre a aplicação e o servidor, o ficheiro é transferido para o cartão de memória. É dado sempre preferência pela utilização do cartão de memória, embora mais lento que a memória local, pelo

facto de existirem ainda muitos equipamentos móveis com pouca memória local.

```
@Override
protected String doInBackground(String... mUrl) {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state) || Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {

        try {
            //Build directory structure
            File mfolder = new File(sdPath);
            mfolder.mkdirs();
            URL url = new URL(this.mUrl);
            InputStream input = url.openStream();
            try {
                OutputStream output = new FileOutputStream (sdPath + pdfName);
                try {
                    byte[] buffer = new byte[500000];
                    int bytesRead = 0;

                    while ((bytesRead = input.read(buffer, 0, buffer.length)) >= 0) {
                        output.write(buffer, 0, bytesRead);
                    }
                } finally {
                    output.close();
                }
            } finally {
                input.close();
            }

        } catch (Exception e) {
        }
    } else {
        Toast.makeText(this.mContext, R.string.verifiqueCartao, Toast.LENGTH_SHORT).show();
    }
    return null;
}
```

Figura 78 - Transferir PDF

A figura 78 demonstra como é feito a pesquisa da tarifa localmente, caso não existe ligação à Internet. Desde que seja realizado pelo menos uma consulta à tarifa, o PDF passa a estar disponível para consulta em modo *offline.*

Novamente através do conceito de *intent*, facilmente é identificado se existe ou não uma aplicação capaz de abrir o ficheiro PDF, definindo em primeiro lugar o formato do ficheiro pretendido, *intent.setDataAndType(uri,"application/pdf")* e posteriormente por invocar a execução da aplicação que suporta o respectivo formato. Se o pedido invocar *null*, é porque não existe aplicação disponível e então é aberta a aplicação Google Play, para *download* do Adobe Reader.

```
//Checks if there is a PDF reader, then opens it or send to Adobe Reader at Google Play
public void getPdf (String pdfPath) {
    String pdfReader = "com.adobe.reader";
    boolean mOnline = InternetChecker.INSTANCE.isOnline(this.mContext);

    try {
        File file = new File(pdfPath);

        PackageManager packageManager = this.mContext.getPackageManager();
        Intent testIntent = new Intent(Intent.ACTION_VIEW);
        testIntent.setType("application/pdf");
        @SuppressWarnings("rawtypes")
        List list = packageManager.queryIntentActivities(testIntent, PackageManager.MATCH_DEFAULT_ONLY);
        if (list.size() > 0 && file.isFile()) {
            Intent intent = new Intent();
            intent.setAction(Intent.ACTION_VIEW);
            Uri uri = Uri.fromFile(file);
            intent.setDataAndType(uri, "application/pdf");
            intent.setFlags((Intent.FLAG_ACTIVITY_CLEAR_TOP));

            this.mContext.startActivity(intent);
        }
        else if(mOnline){
            Uri marketUri = Uri.parse("market://details?id=" + pdfReader);
            Intent marketIntent = new Intent(Intent.ACTION_VIEW, marketUri);
            this.mContext.startActivity(marketIntent);
        }
    } catch (Exception e) {
    }
}
```

Figura 79 - Procura aplicação instalada que suporte ficheiros PDF

4.2.8. Serviços a Bordo

Interface:

Bloqueando o ecrã na orientação vertical, garante uma consulta no formato de lista, apresentado um serviço após serviço (figura 79), sem o aborrecimento do ecrã mudar de orientação (e reconstruir o *layout*) durante a consulta.

Figura 80 - Serviços a Bordo

Código:

Este é o ecrã com menos código da aplicação, pelo que será utilizado para demonstrar a implementação por fragmentos, comum nos restantes ecrãs.

Cada ecrã na aplicação é invocado através de uma *activity*. Cada *activity* possui um ou mais fragmentos, que por sua vez possuem o seu próprio conteúdo, como representado pela figura seguinte.

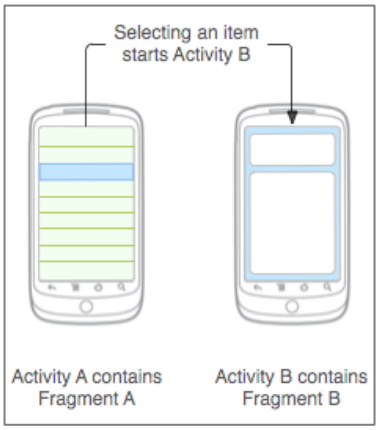


Figura 81 - Fragmentos (*fonte: developer.android.com)*

No caso deste ecrã, existe a classe *NavioActivity*, que representa a actividade principal, descrita na figura seguinte:

```
public class NavioActivity extends ActivityWithOptionsMenu {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.navio);

        //Set the ActionBar settings
        getSupportActionBar().setDisplayShowTitleEnabled(false);
        getSupportActionBar().setNavigationMode(ActionBar.NAVIGATION_MODE_TABS);
        getSupportActionBar().setHomeButtonEnabled(true);
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);


        //SERVICOS A BORDO
        ActionBar.Tab tab = getSupportActionBar().newTab();
        tab.setText(R.string.servicesOnBoard);
        //The fragment
        SherlockFragment fragmentServicos = new ServicosFragmentActivity();
        //The listener
        tab.setTabListener(new TabListener(fragmentServicos));
        getSupportActionBar().addTab(tab);
    }

    //Change the subject and text to share
    @Override
    public void updateShareContent(Intent shareIntent, String mText ) {
        shareIntent.putExtra(Intent.EXTRA_TEXT, "http://goo.gl/gqd6e");
    }
}
```

Figura 82 - Classe para *NavioActivity*

Depois existe a classe *ServicosFragmentActivity*, na figura 82, que representa por sua fez o fragmento “Serviços a Bordo”:

```
public class ServicosFragmentActivity extends SherlockFragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {

        View result = inflater.inflate(R.layout.servicos_fragment, container, false);
        return result;
    }


}
```

Figura 83 - Classe para o fragmento que contém os serviços a bordo

Na eventualidade de surgir a necessidade de adicionar um novo fragmento “Dados do Navio”, bastaria adicionar quatro linhas de código na classe *NavioActivity*, e criar a nova classe representativa do novo fragmento, que definiria (através do *override*) o seu método *onCreateView*, que é responsável pela apresentação do respectivo *layout.*

4.2.9. Porto Santo e Madeira

Interface:

Para este ecrã o cliente enviou uma grande quantidade de informação turística, que facilmente ocuparia todo o ecrã da maioria dos *smartphones* disponíveis de momento. A solução passou por utilizar menus expansíveis, figuras 83 e 84, para apresentar apenas o texto que o utilizador pretende consultar.

Figura 84 - Porto Santo

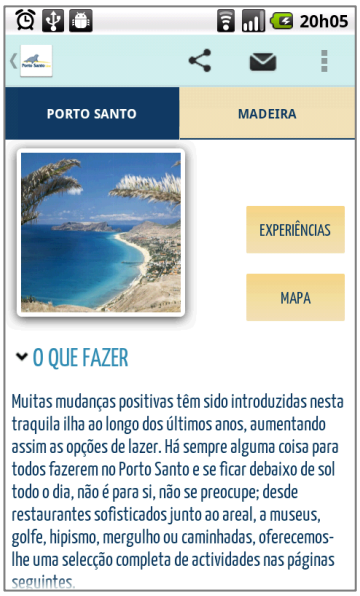


Figura 85 - Menu expansível

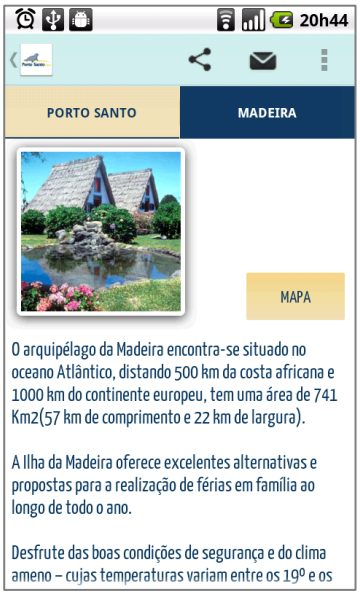


Figura 86 - Madeira

Código:

Nem todos os elementos gráficos estão disponíveis no SDK do Android, como é o caso dos menus expansíveis. O comportamento típico deste *widget*, nome pelo qual são conhecidos os elementos de uma *interface* gráfica em Android, foi desenvolvido especificamente para este ecrã, alterado o estado do atributo *Visibility* dos elementos *CustomTextView,* como é possível verificar na figura seguinte.

```
// Listener for the "Onde Ficar" expandable CustomTextView
public void verOndeficar(View v) {
    CustomTextView tv1 = (CustomTextView) findViewById(R.id.portosanto_menu_ondeficar);
    CustomTextView tv2 = (CustomTextView) findViewById(R.id.portosanto_tv_ondeficar);

    if (tv2.getVisibility() != View.VISIBLE) {
        tv2.setVisibility(View.VISIBLE);
        tv1.setCompoundDrawablesWithIntrinsicBounds(R.drawable.arrow_down, 0, 0, 0);
    }

    else {
        tv2.setVisibility(View.GONE);
        tv1.setCompoundDrawablesWithIntrinsicBounds(R.drawable.arrow_left, 0, 0, 0);
    }
}
```

Figura 87 – Definindo os estados de um menu expansível

4.2.10. Experiências no Porto Santo

Interface:

Reutilizando a *interface* dos "Suplementos", são apresentadas as várias "Experiências" disponíveis na ilha do Porto Santo, juntamente com o preço e descrição de cada (figura 87).

Figura 88 - Experiências no Porto Santo

Código:

Os preços são sempre verificados na abertura deste ecrã, desde que exista uma ligação à Internet estabelecida. No caso de não existir uma ligação disponível, a aplicação efectuará um pedido ao sistema operativo para tentar obter conexão, com base nas definições de rede em vigor. Esta técnica é válida para qualquer ecrã da aplicação que necessite de uma ligação à Internet.

De forma análoga aos "Suplementos", esta tarefa é lançada numa *thread* separada. Recorrendo à *Interface* (Java) do servidor, apresentada na figura seguinte, é executado o método *getExperiencias()*, que devolve os preços actualizados para apresentar no ecrã.

```
/**
 * Let's make the http request and return the result as a String.
 */
protected String doInBackground(Object... args) {
    return ServerInterface.getExperiencias();
}
```

Figura 89 - Recorrendo à Interface (Java) para obter preços actualizados

4.2.11. Mapa: Pontos de Interesse

Interface:

A apresentação de vários Pontos de Interesse geograficamente próximos, visíveis nas figuras 89 e 90, aumentou a dificuldade na tarefa de destacar os estabelecimentos do Grupo Sousa (F21).

Figura 90 - Acedendo ao mapa do menu "Porto Santo"

Figura 91 - Zoom por *gesture* ou painel

Os objectivos principais deste ecrã são:

- Auxiliar o utilizador na localização dos hotéis, restaurantes e lojas do Grupo Sousa.
- Dar a conhecer alguns pontos turísticos da ilha Porto Santo.

Tendo em conta que a lista de Pontos de Interesse aumentou consideravelmente desde o levantamento dos requisitos, este problema não foi equacionado no plano de desenvolvimento. Por esse motivo, foi seleccionada a solução com menor impacto no desenvolvimento do projecto, que consistiu na simples criação de ícones diferentes para os estabelecimentos do Grupo Sousa (figuras anteriores).

Adicionalmente, tratando-se de um mapa, a *interface* reconhece os gestos habituais de zoom (1). Se o utilizador preferir, poderá utilizar o painel de zoom para maior controlo na acção.

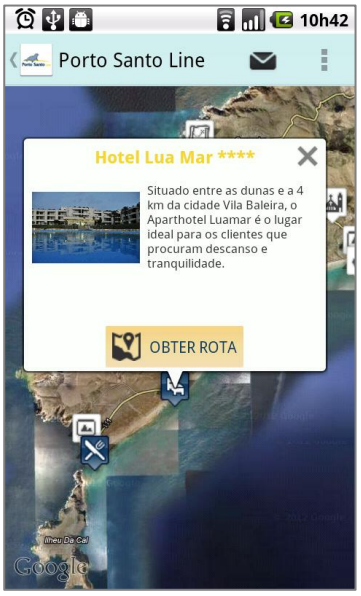


Figura 92 - Detalhes do Hotel Lua Mar

Ao seleccionar um Ponto de Interesse, o mapa é centrado nesse ponto e são apresentados os respectivos detalhes, juntamente com a opção de obter uma rota para trajecto via automóvel (figura 91).

Código:

O mapa é obtido através da biblioteca *Google Maps*, incluída no SDK do Android, que possibilita a comunicação com o respectivo serviço *online,* através de uma chave única de identificação (mais informação sobre a chave no capítulo 6 - Lançamento).

O mapa é constituído por várias camadas (*ArrayList*). Na figura seguinte, é possível observar a criação de algumas dessas camadas. Cada camada possui uma categoria de Pontos de Interesse (*CustomOverlayItem*), por exemplo, os restaurantes estão numa camada diferente dos hotéis. Isto possibilita controlar quais os ícones que devem surgir em primeiro plano, mas, sobretudo, permite expandir o número de Pontos de Interesse sem afectar os já existentes. O aumento de *POI's* no mapa poderá tornar indistinguíveis os ícones, tornando difícil a tarefa de encontrar um determinado ponto no mapa. A criação de várias camadas possibilita a futura implementação de um filtro por categoria de *POI*, aumentado assim o nível de usabilidade deste ecrã e solucionando o problema.

```
// *** RESTAURANTS ***
ArrayList<CustomOverlayItem> restaurants = new ArrayList<CustomOverlayItem>();
restaurants.add(new CustomOverlayItem(
        new GeoPoint(33024730, -16379349),
        getString(R.string.ponta_calheta),
        getString(R.string.ponta_calheta_desc),
        R.drawable.calhetas,
        true));

restaurants.add(new CustomOverlayItem(
        new GeoPoint(33055836, -16338151),
        getString(R.string.salinas),
        getString(R.string.salinas_desc),
        R.drawable.salinas,
        true));

restaurants.add(new CustomOverlayItem(
        new GeoPoint(33055973, -16337949),
        getString(R.string.pizza_areia),
        getString(R.string.pizza_areia_desc),
        R.drawable.pizza_areia,
        true));

//*The OVERLAY with icons
Drawable drawable2 = getResources().getDrawable(R.drawable.restaurant);
CustomItemizedOverlay restaurantsLayer = new CustomItemizedOverlay(drawable2, mapView);

for (Iterator<CustomOverlayItem> i = restaurants.iterator(); i.hasNext(); ) {
    restaurantsLayer.addOverlay(i.next());
}
//Add layer
mapOverlays.add(restaurantsLayer);
```

Figura 93 - Camada com os restaurantes

4.2.12. Mapa: Cálculo de Rota

Interface:

Incorporado na funcionalidade "Consultar mapa", o botão "Obter Rota" oferece a possibilidade ao utilizador de visualizar a melhor rota entre a sua localização e qualquer Ponto de Interesse no mapa, incluindo as lojas e balcões da Porto Santo Line.

Após o cálculo da rota, a *interface* centra o mapa na posição actual do utilizador, ajustando o zoom e adicionando um ícone com uma seta no ponto de partida (figura seguinte).

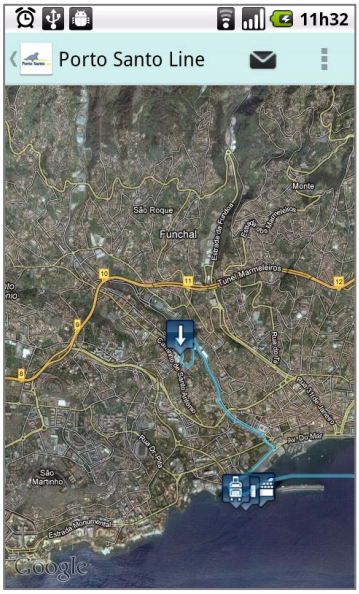


Figura 94 - Rota entre UMa (Madeira) e Hotel Lua Mar (Porto Santo)

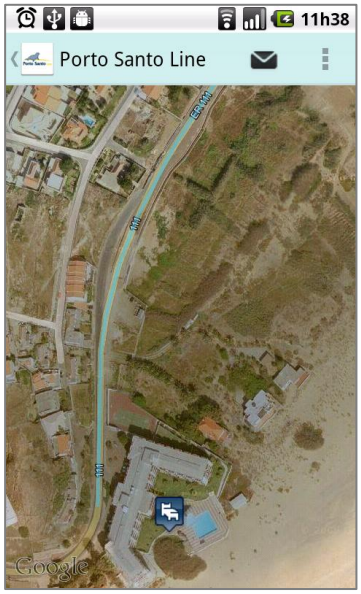


Figura 95 - Rota calculada até o Hotel Lua Mar

A rota é marcada com azul transparente a fim de não bloquear a leitura dos nomes das vias de transporte nem bloquear a visualização do próprio mapa (figura 94).

Código:

Este módulo é dividido em duas principais fases:

1. Obter coordenadas da posição actual do dispositivo (origem).
2. Obter a rota para o POI seleccionado (destino).

A posição geográfica de um dispositivo Android pode ser adquirida através de *GPS*, "*WiFi MAC ID"* ou *"Cell ID"*. Cada tecnologia apresenta as suas vantagens e desvantagens. Resumidamente:

- GPS: Possui alto nível de precisão mas é lento em obter coordenadas.
- MAC ID / CELL ID: Menos preciso, mas extremamente rápido.

A solução implementada, parcialmente apresentada na figura seguinte, tenta utilizar o melhor de cada tecnologia. No caso de existir já um sinal GPS fixo, a aplicação, durante a abertura do mapa, fará uma solicitação ao sistema operativo para obter uma nova actualização da posição geográfica do dispositivo, através do GPS. Porém, se não existir um sinal GPS activo e existir

uma ligação WiFi válida, no pedido é indicado a preferência pelo dispositivo WiFi, apoiado pela rede do operador, como fonte das coordenadas.

```
class GetLastLocation extends TimerTask {
    @Override
    public void run() {
        lm.removeUpdates(locationListenerGps);
        lm.removeUpdates(locationListenerNetwork);

        Location net_loc=null, gps_loc=null;
        if(mGpsEnabled)
            gps_loc=lm.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        if(mNetworkEnabled)
            net_loc=lm.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

        //if there are both values use the latest one
        if(gps_loc!=null && net_loc!=null){
            if(gps_loc.getTime()>net_loc.getTime())
                locationResult.gotLocation(gps_loc);
            else
                locationResult.gotLocation(net_loc);
            return;
        }

        if(gps_loc!=null){
            locationResult.gotLocation(gps_loc);
            return;
        }
        if(net_loc!=null){
            locationResult.gotLocation(net_loc);
            return;
        }
        locationResult.gotLocation(null);
    }
}
```

Figura 96 - Verificando qual a melhor opção para obter as coordenadas (código parcial)

É esperado que independentemente da tecnologia utilizada, esta tarefa poderá exigir mais de 10 segundos para ser executada com sucesso. Este facto é importante pelo motivo do sistema Android forçar o fecho de qualquer aplicação que não responda por mais de 10 segundos, situação conhecida por ANR [24]. Por este motivo, a tarefa de obter as coordenadas geográficas, é executada numa *thread* separada e processada em segundo plano, devolvendo uma resposta à *thread* principal após a sua conclusão. As *threads* são sempre terminadas após concluírem a tarefa atribuída (com sucesso ou não).

Após obter as coordenadas da posição actual, é iniciada a segunda fase do processo, que consiste em obter a rota para o POI seleccionado. Utilizando a API do serviço *Google Directions,* através de JSON, é possível obter um conjunto de pontos que juntos formam a rota entre duas coordenadas geográficas.

Como medida de precaução, é verificado se o utilizador encontra-se em território português, impedindo que um utilizador fora de Portugal sobrecarregue o serviço com um pedido de rota.

4.2.13. Balcões: Contactar e Localizar

Interface:

Objectivo é fornecer uma lista dos balcões e armazéns da empresa (figura 97), usufruindo do mapa para efeitos de localização. Adicionalmente permitir que o utilizador entre em contacto com os armazéns em dois "*taps*" (figura 98).

Figura 97 - Lista de balcões e armazéns

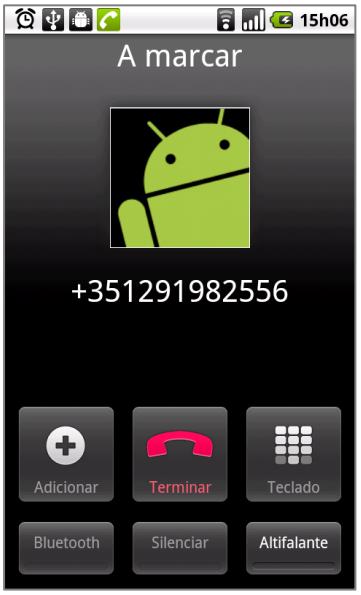


Figura 98 - A iniciar chamada telefónica

Código:

É reutilizado o mesmo mapa nos restantes ecrãs da aplicação, centrando o ecrã no respectivo balcão.

Para realizar a chamada telefónica correcta, é primeiro verificado qual o botão que invocou o método, e recorrendo ao conceito de *intent* já explicado anteriormente, o número de telefone é associado ao novo *intent,* através do método *setData,* seguido da invocação aplicação *dialer* do próprio sistema Android, como descrito na figura seguinte.

```
public void makeCall(View v) {
    Intent intent = new Intent(Intent.ACTION_CALL);
    switch (v.getId()) {
    case R.id.call1:
    {
        intent.setData(Uri.parse("tel:+351291210300"));
        break;
    }
    case R.id.call2:
        intent.setData(Uri.parse("tel:+351291214880"));
        break;

    case R.id.call3:
        intent.setData(Uri.parse("tel:+351291982938"));
        break;

    case R.id.call4:
        intent.setData(Uri.parse("tel:+351291982556"));
        break;

    case R.id.call5:
        intent.setData(Uri.parse("tel:+351291210300"));
        break;
```

Figura 99 - Utilizando o mesmo *Intent* para os vários números de telefone

4.2.14. Contactos Gerais

Interface:

A figura seguinte apresenta o ecrã que disponibiliza os meios de comunicação com o *contact center* da Porto Santo Line.

Código:

A figura 100, demonstra como o cliente de email instalado no dispositivo Android é invocado, reutilizando o mesmo *intent* criado anteriormente para as chamadas telefónicas, alterando os atributos com os dados necessários para o envio de um email. Como assunto do email, é enviado um número aleatório, utilizando *Random(1000),* para facilitar o rastreamento do incidente.

```
    case R.id.call6:
        Random mRandom = new Random(1000);
        Calendar mCalendar = Calendar.getInstance();
        String ticketNumber =  Integer.toString(mCalendar.get(Calendar.YEAR)) +
                Integer.toString(mCalendar.get(Calendar.MONTH)+1) +
                Integer.toString(mCalendar.get(Calendar.DAY_OF_MONTH)) +
                Integer.toString(mRandom.nextInt());

        intent = new Intent(android.content.Intent.ACTION_SEND);
        String[] recipients = new String[]{"info@portosantoline.pt"};
        intent.putExtra(android.content.Intent.EXTRA_EMAIL, recipients);
        intent.putExtra(android.content.Intent.EXTRA_SUBJECT, "Ticket ID: " + ticketNumber);
        intent.putExtra(android.content.Intent.EXTRA_TEXT,R.string.contactCenterEmailText );
        intent.setType("text/plain");
        Intent.createChooser(intent, "Porto Santo Line Contact Center");
        break;

    default:
        return;
    }
    startActivity(intent);
    intent = null; //Help the GC to remove the garbage(free memory)
}
```

Figura 100 - Mesmo *intent* com atributos diferentes

4.2.15. Ecrã "Sobre" e menu Ajuda

Interface:

No ecrã da figura 101, é fornecido informação geral sobre a aplicação (versão), ligação ao *site* da empresa (utiliza *o browser* incluído no sistema) e créditos. No segundo ecrã, figura 102, é disponibilizado um FAQ, com as perguntas e respostas seleccionadas pelo cliente.

Figura 101 - Sobre

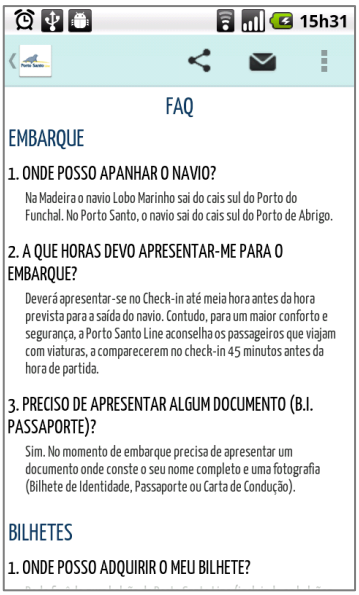


Figura 102- FAQ

4.2.16. Recolha de Dados Estatísticos de Utilização

A administração da empresa Porto Santo Line mostrou especial interesse em obter dados estáticos sobre a utilização da aplicação Android.

Durante a fase de análise do projecto, foi observado a utilização do serviço *Google Analytics* no *site* da Porto Santo Line, pelo que foi optado por implementar o mesmo serviço na aplicação móvel (figura 103). Duas vantagens imediatas:

- Cliente familiarizado com o serviço *Google Analytics*.
- Recolha de dados estatísticos centralizados numa única plataforma

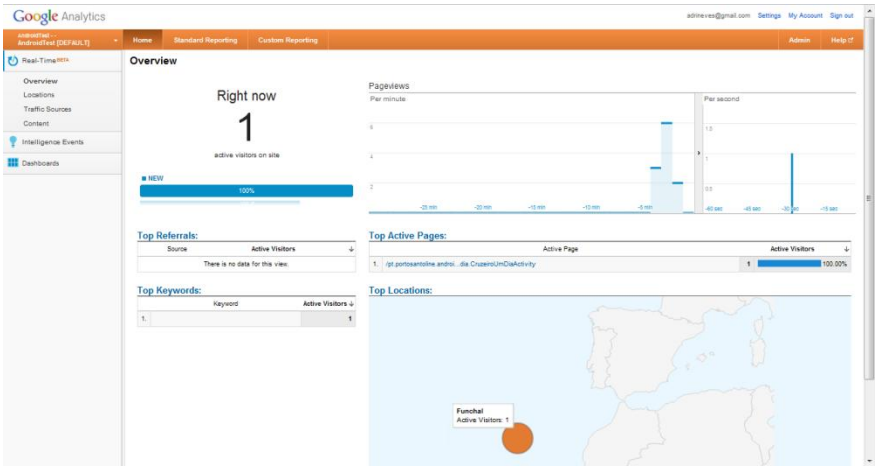


Figura 103 - Monitorização em tempo real da utilização da aplicação recorrendo ao *Google Analytics*

Código:

Bibliotecas Java: *EasyTracker* e *GoogleAnalytics.*

Tirando partido do conceito de herança da programação orientada a objectos, como é possível verificar na figura seguinte, em que uma das classes da biblioteca *EasyTracker*, a *TrackedSherlockFragmentActivity* é convertida numa subclasse da superclasse *SherlockFragmentActivity*, integrando a funcionalidade de monitorização em todos os ecrãs da aplicação que usem fragmentos.

```
public class TrackedSherlockFragmentActivity extends SherlockFragmentActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Only one call to setContext is needed, but additional calls don't hurt
        // anything, so we'll always make the call to ensure EasyTracker gets
        // setup properly.
        EasyTracker.getTracker().setContext(this);
    }

    @Override
    protected void onStart() {
        super.onStart();

        // This call will ensure that the Activity in question is tracked properly
        // based on the setting of ga_auto_activity_tracking parameter.  It will
        // also ensure that startNewSession is called appropriately.
        EasyTracker.getTracker().trackActivityStart(this);
    }

    @Override
    protected void onStop() {
        super.onStop();

        // This call is needed to ensure time spent in an Activity and an
        // Application are measured accurately.
        EasyTracker.getTracker().trackActivityStop(this);
    }
}
```

Figura 104 - Extendo a superclasse "SherlockFragmentActivity"

4.2.17. Idiomas Suportados

Sendo uma aplicação turística, é imprescindível a tradução do conteúdo para os principais idiomas do público-alvo. Com base no levantamento dos requisitos e no conteúdo já existente no *site* oficial da empresa, a aplicação suporta os seguintes idiomas:

- Inglês
- Português
- Alemão

Implementação:

A tradução de conteúdo na plataforma Android consiste em organizar o texto em ficheiros *XML* distribuídos por pastas *values*, figura seguinte, concatenadas com o nome internacional do idioma (“*de*” para alemão, “pt” para português).

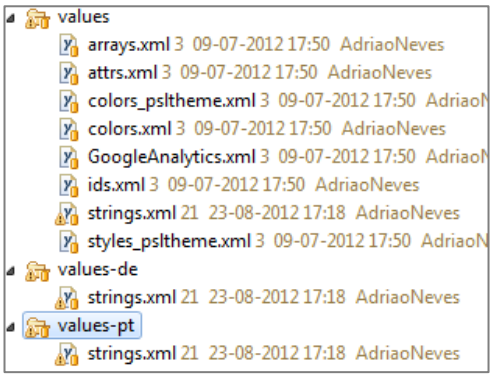


Figura 105 - Tradução do conteúdo

De forma análoga, são criadas as imagens que possuem texto incorporado para cada idioma.

Destaque para a colaboração da Porto Santo Line na disponibilização do conteúdo em alemão.

4.3. Funcionalidades do Painel de Administração

Alguns dos produtos turísticos da Porto Santo Line, estão disponíveis apenas em determinadas épocas. Assim sendo, tornou-se claro a necessidade de controlar não apenas os preços, como também determinar quais os produtos em vigor.

Objectivos:

- Actualização de preços (Suplementos e Experiências).
- Activar ou desactivar “Experiências” e “Suplementos”.

4.3.1. Actualização dos Suplementos e das Experiências

Código:

Do lado do servidor existe o ficheiro *server.php,* figura 106, que recebe os pedidos da aplicação Android e obtém a respectiva informação da base de dados *MySQL*. Essencialmente, verifica qual o comando recebido, se *getSuplementos* ou *getExperiencias*, e acede à respectiva tabela na base de dados *MySQL*.

```php
<?php
include  ('edit/config/connect.php');
/** INTERFACE FOR ANDROID APPLICATION */

// get the command
$command = $_REQUEST['command'];


// determine which command will be run
if($command == "getSuplementos") {
    // return the list
    // Table
    $result = mysql_query('SELECT * FROM suplementos') or die(mysql_error());

    while($row = mysql_fetch_array($result))
    {
        echo $row['preco'] . ";" . $row['activo'] . ";";
    }

} else if($command == "getExperiencias") {
    // return the list
    $result = mysql_query('SELECT * FROM experiencas') or die(mysql_error());

    while($row = mysql_fetch_array($result))
    {
        echo $row['preco'] . ";" . $row['activo'] . ";";
    }

}
else {
    echo "";
```

Figura 106 - server.php

Do lado do cliente (aplicação Android), existe a classe *ServerInterface,* parcialmente apresentada na figura seguinte, que funciona como *interface* entre os dois programas. Esta classe é responsável por estabelecer a ligação remota, enviar o respectivo comando, *getSuplementos* ou *getExperiencias*, e processar a resposta do servidor.

```java
import java.io.DataInputStream;

public class ServerInterface {
    // Declared Constants
    public static final String SERVER_URL = "http://67.23.226.119/~pslineor/admin/server.php";

    /**
     * Gets the list from the server.
     * @return A string containing a comma-delimited list
     */
    public static String getSuplementos() {
        /*
         * Let's construct the query string. It should be a key/value pair. In
         * this case, we just need to specify the command, so no additional
         * arguments are needed.
         */
        String data = "command=" + URLEncoder.encode("getSuplementos");
        return executeHttpRequest(data);
    }

    public static String getExperiencias() {
        /*
         * Let's construct the query string. It should be a key/value pair. In
         * this case, we just need to specify the command, so no additional
         * arguments are needed.
         */
        String data = "command=" + URLEncoder.encode("getExperiencias");
        return executeHttpRequest(data);
    }
```

Figura 107 - ServerInterface

5. TESTES E RESULTADOS

"Ever Tried. Ever failed. No matter. Try again. Fail again. Fail better."

(Samuel Beckett, "Worstward Ho")

Testar o *software* nem sempre é uma tarefa fácil, mas é um passo essencial no processo de desenvolvimento, que garante se o produto "final" corresponde aos requisitos do cliente e dos utilizadores.
Um teste não corrige erros, mas diminui a ocorrência com que surgem.

Devido ao facto da equipa de desenvolvimento acumular conhecimento sobre a aplicação e do seu funcionamento, os testes devem ser realizados por outra equipa. Obviamente que neste projecto, tendo em conta os recursos humanos disponíveis, os testes foram criados pelo programador e realizados com pessoas que não estiveram directamente envolvidas no desenvolvimento.

Os testes seguintes foram realizados com a colaboração de seis pessoas, divididas em dois grupos de três elementos. Os utilizadores de teste foram divididos em grupos, a fim de evitar a transferência e acumulação de experiência, fenómeno caracterizado pela experiência que é adquirida por um utilizador em cada interacção com o sistema. Um número maior de pessoas para a realização dos testes seria melhor, porém é igualmente importante possuir a capacidade de adaptar o desenvolvimento de um projecto aos recursos disponíveis.

Este conjunto de testes tem por objectivo verificar o correcto funcionamento de cada caso de utilização. Inclui uma lista dos passos que idealmente devem ser seguidos e uma descrição dos resultados esperados. No fim de cada teste, os utilizadores são questionados sobre o nível de satisfação na execução da tarefa. Foi criado um caso de teste para cada caso de utilização [15]. Para além de testarem se os requisitos da aplicação são cumpridos, os seguintes testes também procuram detectar defeitos de usabilidade [18].

Os testes de usabilidade foram concebidos e executados tendo em consideração o método "*Think-Aloud*" e as orientações disponíveis em *http://www.testingstandards.co.uk* [18].

5.1. Testes de Usabilidade – "Think-Aloud"

De forma individual, é solicitado a cada utilizador a execução de uma funcionalidade da aplicação, enquanto transmite em voz alta os motivos das suas decisões, o comportamento esperado do sistema em cada acção e como interpreta o que vê. Durante este processo, o utilizador é atenciosamente observado, a fim de detectar incertezas nas suas escolhas ou erros de decisão [16, 17].

Nas tabelas seguintes, "Caso de teste", são resumidas as observações realizadas durante o processo. Entre outras informações, são indicados os passos necessários para executar a tarefa, seguido dos resultados esperados e obtidos. As observações incluem essencialmente o resultado principal de cada teste, essencialmente indicam o que é necessário corrigir na *interface*.

CASO DE TESTE	**#1 – Obter preço do suplemento "Excursão"**
Descrição	Os suplementos são serviços turísticos (como transferes) que podem ser adicionados ao pacote "Cruzeiros de 1 Dia". Neste teste, o utilizador deve consultar a lista de todos os suplementos disponíveis para compra, deslizar a lista até ao fim e indicar o preço do suplemento "Cruzeiros de 1 Dia + Transferes + Excursão".
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida.
Caso de Utilização	Obter lista de suplementos
Requisitos	F1, F2, F3.
Passos	1. Seleccionar "Cruzeiros de 1 Dia". 2. Seleccionar "Suplementos" 3. Deslizar lista até o fim.
Resultados esperados	Utilizador deve indicar o preço: 70,00€.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Dificuldade em encontrar o menu "Cruzeiros de 1 Dia".

Tabela 3 - Caso de teste #1: Obter preço do suplemento "Excursão" com Internet

Para não prolongar demasiado esta secção, os restantes casos de teste estão disponíveis na secção "Anexos" deste documento.

Na secção seguinte, "*Resumo*", são discutidas as conclusões e os impactos dos casos de teste na *interface* da aplicação Android.

5.2. Avaliação por parte do Cliente

Antes do lançamento oficial da aplicação, cabe ao cliente, mesmo que acompanhando o processo de desenvolvimento e constantemente avaliando o progresso e decisões tomadas, realizar uma avaliação geral ao *software.* Será esta avaliação a determinar a aceitação ou rejeição do sistema. Certamente cada cliente terá os seus próprios critérios, porém é expectável que a validação da implementação dos requisitos acordados seja o factor com maior peso na decisão.

No caso deste projecto, a aprovação também representa a autorização para iniciar o processo de lançamento, como a aplicação oficial do Lobo Marinho, no mercado de aplicações do Android.

5.3. Resumo

Os testes de usabilidade devem sempre ser realizados por pessoas que não seguiram o desenvolvimento do projecto. Isto demonstrou-se essencial para detectar falhas na interface do utilizador.

No **caso de teste #1**, foi observado dificuldade nos utilizadores em identificarem o botão “Cruzeiros de 1 Dia”. Tendo em conta que a leitura é feita de cima para baixo, da esquerda para a direita (pelo menos a maioria do público alvo desta aplicação), foram feitas alterações na imagem do botão, a fim de facilitar a leitura e consequente interpretação (figuras 108 e 109).

Figura 108 – Botão “Cruzeiros de 1 Dia” antes do caso de teste #1

Figura 109 – Botão “Cruzeiros de 1 Dia” depois do caso de teste #1

A figura do botão “**Promoções**” foi alvo das mesmas alterações.

O **caso de testes #2** mostrou que o ícone com a lupa não é suficiente para indicar ao utilizador que existe uma acção associada ao ícone. A data mostrou-se mais identificativa da acção que o ícone da lupa. Falta adicionar profundidade ao botão e o fundo deveria ser de outra cor (figura 110).

Figura 110 - Erro de usabilidade por corrigir

O **caso de testes #6** demonstra que seria mais eficiente, em termos de usabilidade, abrir imediatamente o POI, sempre que é chamado directamente, como no caso dos balcões e armazéns.

Com os dados do **caso de testes #7**, foi observado que os utilizadores não detectam o mapa com a facilidade pretendida. A adição de um ícone ao botão “Mapa” é uma forma de garantir melhor visibilidade, ajudando o utilizador a identificar a acção de forma mais eficiente.

Os **casos de teste #9 e #11** identificaram um grave erro de usabilidade: um ícone para duas acções distintas (figura 111). Embora a solução seja simples, ícones diferentes, este erro acaba por demonstrar como importante é a realização dos testes de usabilidade.

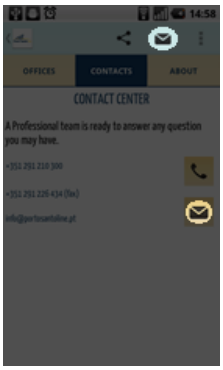


Figura 111 - Grave erro no desenho da interface

A fim de cumprir com a data acordada de lançamento, alguns dos testes acima descritos foram realizados após o lançamento da aplicação no mercado, e por esse motivo, a implementação das respectivas soluções ficaram agendas para a próxima versão da aplicação.

6. LANÇAMENTO

"If a software project becomes too large, it will collapse into a black hole. Time and money are absorbed but nothing ever comes out".

(Brian Russell's Laws of Software Relativity)

O próprio processo de lançamento da aplicação deve ser planeado e executado da melhor forma possível. Todo o trabalho realizado, todo o tempo despendido e todo o dinheiro investido no projecto, são colocados em risco na fase do lançamento da primeira versão pública.

6.1. Google Play (Android Market)

O mercado oficial das aplicações (e jogos) da plataforma Android é conhecido por Google Play, anteriormente denominado por Android Market. É neste mercado, uma loja *online* de software, que as aplicações são disponibilizadas para os aproximadamente 900 milhões utilizadores Android [4]. O mercado é acessível por *browser* em qualquer dispositivo com ligação à Internet e pela própria aplicação oficial Google Play, instalada de raiz em todos os dispositivos Android.

Assinatura privada e chave API Google Maps

Converter a aplicação para uma versão *release*, implica alterar assinar o ficheiro APK (*Android Application Package* – formato do ficheiro utilizado para distribuir e instalar aplicações Android) com o certificado privado do programador. O sistema Android utiliza o certificado como meio de verificar os autores das aplicações. Com base na impressão digital única de cada certificado (figura 112), é gerada a chave privada que permite a utilização da API Google Maps. A mudança do certificado implica a actualização desta chave, a fim de manter os mapas da aplicação funcionais.

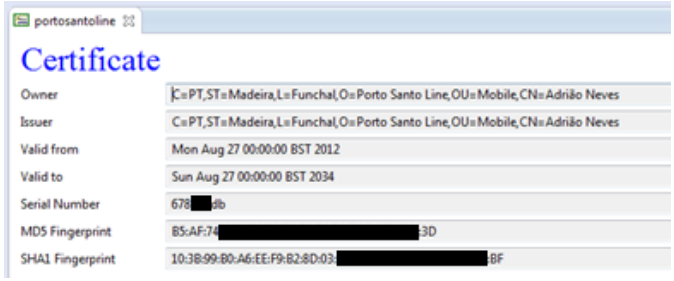


Figura 112 - Certificado utilizado na assinatura da aplicação

Testar a primeira versão pública

É importante testar o funcionamento da aplicação em modo de lançamento, assinada com o certificado privado e com as novas chaves das *API's*, a fim de verificar que a aplicação continua a comunicar com os serviços externos, como o caso do serviço Google Maps, essencial para obter as imagens de satélite do mapa da aplicação.

Material gráfico

Para usufruir em pleno das funcionalidades de publicidade da loja online Google Play, são necessárias *screenshots* da aplicação, versão de alta resolução do ícone da aplicação (imagem de capa) e duas imagens que funcionem como *banner* da aplicação (figuras 113 e 114).

Pelo motivo da loja existir na versão *mobile* e na versão *web*, as imagens devem evitar o uso de texto, pois num equipamento móvel com ecrã reduzido a leitura torna-se difícil.

Figura 113 - *Banner* da aplicação

Figura 114 - *Banner* da aplicação em formato *mobile*

Aplicação disponível para *download*

Recorrendo à conta Google da Porto Santo Line (figura 115), criada para este efeito, é lançada a versão 1.0.0 da aplicação no mercado oficial Android (figura 116).

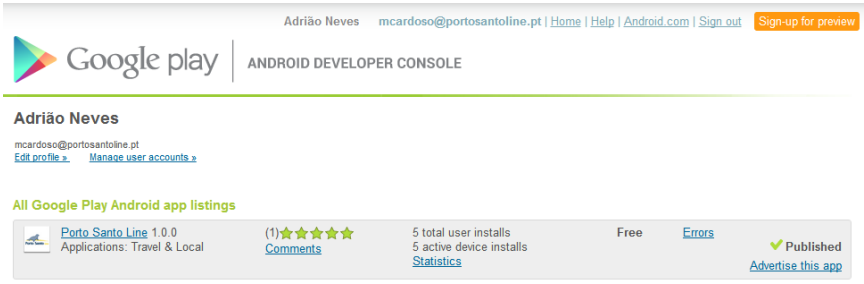


Figura 115 – Google Play: Consola do programador Android

Figura 116 - Aplicação no Google Play

Marketing

Apesar do Google Play realizar alguma publicidade da aplicação, é essencial realizar divulgação nos canais próprios da empresa Porto Santo Line.

Foi disponibilizado ao cliente a seguinte figura (juntamente com o código HTML necessário) para efeitos de *marketing.* No momento da elaboração deste documento, a sua inserção no *site* www.portosantoline.pt aguarda autorização oficial.

Figura 117 - *Link* para a página da aplicação no Google Play

7. CONCLUSÕES

Com o desenvolvimento deste projecto foi possível aplicar e aprofundar os conhecimentos adquiridos ao longo do curso. Ofereceu, também, a oportunidade de conhecer novas realidades e de obter novos conhecimentos. As várias reuniões com o cliente, Porto Santo Line, não só foram importantes para a concepção do projecto, como representaram uma excelente oportunidade de aprender a interagir com os vários *stakeholders* de um projecto de *software*.

Desenvolver este projecto para a Porto Santo Line exigiu compreender a sua actividade produtos e serviços, o que seria impossível de realizar sem a colaboração e envolvimento dos colaboradores da empresa. Manter o cliente envolvido durante todo o processo de desenvolvimento, nem sempre é fácil, mas oferece melhores garantias de que o produto final cumprirá com os requisitos e suaviza o impacto das mudanças.

Adaptar alguns aspectos da metodologia *Agile*, auxiliou a transpor desafios comuns em muitos projectos de *software*, particularmente os desafios relacionados com a mudança. Independente do rigor e qualidade aplicados nas fases de análise e concepção, as mudanças acabarão por surgir. Por este motivo, é importante conceber sistemas que aceitem a mudança, com o mínimo de custos possíveis.

Compreender a evolução da tecnologia móvel, mostrou-se fundamental para identificar tendências tecnológicas e tomar decisões a pensar no desenvolvimento futuro do sistema. Antes de criar algo, é importante conhecer o que já existe. A análise de algumas aplicações Android com finalidades turísticas, indicou o ponto de partida para a criação dos *wireframes* e protótipos, elementos indispensáveis para identificar problemas de usabilidade e também como ferramenta de comunicação entre os *stakeholders*. A usabilidade nunca foi tão analisada e valorizada como actualmente. A *interface* do utilizador pode traçar o fracasso de um *software*, independentemente número de funcionalidades que oferece.

A fase de desenvolvimento implicou aprofundar os conhecimentos de *Java*, da plataforma Android, de formatos de transferência de dados como JSON, de técnicas de *fetching*, entre outras tecnologias envolvidas no projecto. A utilização do *Subclipse* (SVN) como sistema para controlo de versões, foi revelador. Controlo de versões era algo evitado, devido sobretudo a má avaliação no grau de dificuldade de utilização deste tipo de sistemas.

Transitar do desenvolvimento de programas *desktop* para aplicações móveis, apresenta os seus próprios desafios. Ecrãs de pequenas dimensões, desempenho limitado e autonomia de bateria, são alguns dos vários factores menosprezados em desenvolvimento para *desktop*, mas fundamentais para as plataformas móveis.

Para além dos desafios próprios de uma plataforma móvel, surgiram também questões sobre como certas funcionalidades poderiam afectar negativamente o funcionamento normal da empresa. Foi o caso da funcionalidade "Comprar bilhete *online*", que durante o seu desenvolvimento e testes de usabilidade preliminares, foi decidido não proceder com a implementação, devido a criar problemas no processo de *check-in* dos passageiros. Estes problemas estariam relacionados com problemas em ler o código de barras do bilhete directamente do ecrã de um *smartphone* e com questões legais. Tendo em conta a fase avançada da implementação da funcionalidade, que possuía já implementado um módulo para gerir os bilhetes através do dispositivo, talvez o mais indicado seria realizar alguns testes no terreno com um pequeno grupo de passageiros, antes de tomar uma decisão. Por outro lado, é compreensível que devido às questões legais que esta funcionalidade levantou, fosse tomada a decisão de não investir mais recursos e cancelar a implementação da compra de bilhetes pela aplicação Android.

Os restantes testes de usabilidade forneceram *feedback* essencial para aperfeiçoar o *software* antes do lançamento oficial. Embora fosse preferível obter um grupo de testes maior, foi possível realizar os testes mínimos à *interface*, identificando imediatamente alguns erros e problemas de usabilidade. A implementação de algumas das soluções foi adiada para a próxima versão, a fim de evitar mais adiamentos no lançamento oficial. Num ambiente puramente profissional, provavelmente seria preferível adiar novamente a data de lançamento e implementar todas as correcções necessárias. Seria uma decisão a tomar pelo gestor de projecto.

O lançamento oficial da aplicação no Google Play foi realizada após validação por parte do cliente, através da conta adquirida para o efeito. A empresa Porto Santo Line suportou o custo financeiro da conta *Android Developer* e da aquisição do servidor de *webhosting,* utilizado para hospedar o Painel de Administração.

Continua a ser mantido um canal de comunicação com o cliente, a fim de realizar um acompanhamento das fases seguintes do lançamento e do *marketing* da aplicação.

Com a criação deste sistema, a Porto Santo Line passa agora a marcar presença no mercado Android, divulgando os seus serviços e produtos numa das maiores plataformas móveis da actualidade.

7.1. Trabalho Futuro

Há sempre espaço para melhorias, pelo que o planeamento de todo o sistema foi pensando na expansão da aplicação e na implementação de novas funcionalidades.

Para além da fase normal de manutenção do *software*, que inclui identificação e correcção de *bugs*, rectificação de erros de usabilidade na *interface* e alterações com base no *feedback* dos utilizadores, existe um conjunto de funcionalidades cuja implementação contribuirá para uma melhor divulgação dos serviços e produtos da empresa.

- Compra de bilhetes *online:* Embora exista questões legais e técnicas por resolver, a possibilidade de comprar o bilhete directamente de um *smartphone* continua a ser aliciante.

- Gestão de bilhetes: Com a compra de bilhetes torna-se necessário permitir apagar, consultar e transferir os bilhetes adquiridos pela aplicação. A gestão seria limitada aos bilhetes existentes no dispositivo, apenas devido ao facto do actual *site* não possuir um sistema de registo de utilizadores.

- Venda de suplementos: Com uma base de dados dos utilizadores, já seria praticável a venda de suplementos ao bilhete “Cruzeiros de 1 Dia”, directamente da aplicação.

- Realidade aumentada: Permitir ao utilizador apontar a câmara do *smartphone* para a ilha do Porto Santo, durante a viagem marítima, e obter informações sobre os hotéis, restaurantes e outros pontos de interesse turísticos.

Qualquer funcionalidade a acrescentar, requer uma análise cuidada de como afectará a execução das tarefas na empresa e das questões legais próprias do mercado turístico e do transporte marítimo.

A tecnologia deve ser implementada sempre para melhorar e nunca para dificultar a execução de tarefas.

8. REFERÊNCIAS

[1] F. Darwin, *The Android Developer ' s Cookbook: Building Applications, and Android Sdk* , Addison Wesley, 2011.

[2] Weiser, Mark. *The Computer for the Twenty-First Century*", Scientific American 265.3: 94-104, 1991.

[3] Sharma, C. (2008). "*Mobile Services Evolution"*, 1 – 11, 2008-2018

[4] The International Telecommunication Union, 2011.

[5] Z. Chris, *Android: A visual history*, *The Verge*, 2011.

[6] Google Developers, *http://developer.android.com/about/dashboards/index.html,* acedido em Agosto de 2012.

[7] Charland, Andre, and Brian Leroux. "*Mobile Application Development: Web vs native.*" Communications of the ACM 54.5, 2011.

[8] Lionbridge, "*Mobile Web Apps vs Mobile Native Apps : How to Make the Right Choice"*, 2012.

[9] S. Kathy, B. Bert, *Head First Java*. O'Reilly Media, 2005.

[10] Beck, Kent, "*Manifesto for Agile Software Development*". Agile Alliance, 2001.

[11] "*Design Patterns and Refactoring*", University of Pennsylvania, webpage: UPenn-Lectures-design-patterns, 2003.

[12] "*Extreme Programming*" (lecture paper), USFCA.edu, webpage: USFCA-edu-601-lecture.

[13] Beck, K. "*Extreme Programming Explained: Embrace Change".* Addison-Wesley, 1999.

[14] Stellman, A., Greene, J., Stellman, A., & Greene, J. (n.d.). "*Software Requirements Specification – Outline Introduction"*, 1 –4.

[15] K. Brad, Carnegie Quality*, http://www.carnegiequality.com/testing/test-script-template,* acedido em 2012.

[16] G. Perlman, G. K. Green, & M. S. Wogalter (Eds.) "*Human Factors Perspectives on Human-Computer Interaction: Selections from Proceedings of Human Factors and Ergonomics Society Annual Meetings"*, 1983-1994, Santa Monica, CA: HFES, 1995, pp. 191-195.

[17] Wright, P.C. & Monk, A.F. "*Evaluation for design"*. In Sutcliffe, A. & Macaulay, L. (Eds), People and computers V, Cambridge University Press, pp. 345-358, 1989.

[18] Testing Standards, *http://www.testingstandards.co.uk/usability_guidelines.htm*, acedido em 2012.

[19] Google User Interface Guidelines (2012), *http://developer.android.com/guide/practices/ui_guidelines/index.html*, acedido em 2012.

[20] S. Paulo, “Arquitectura de Aplicação & modelo 3 camadas”, Instituto Superior de Engenharia do Porto, 2011.

[21] UML 2.0 OMG (2012), *http://www.uml.org,* acedido em 2012.

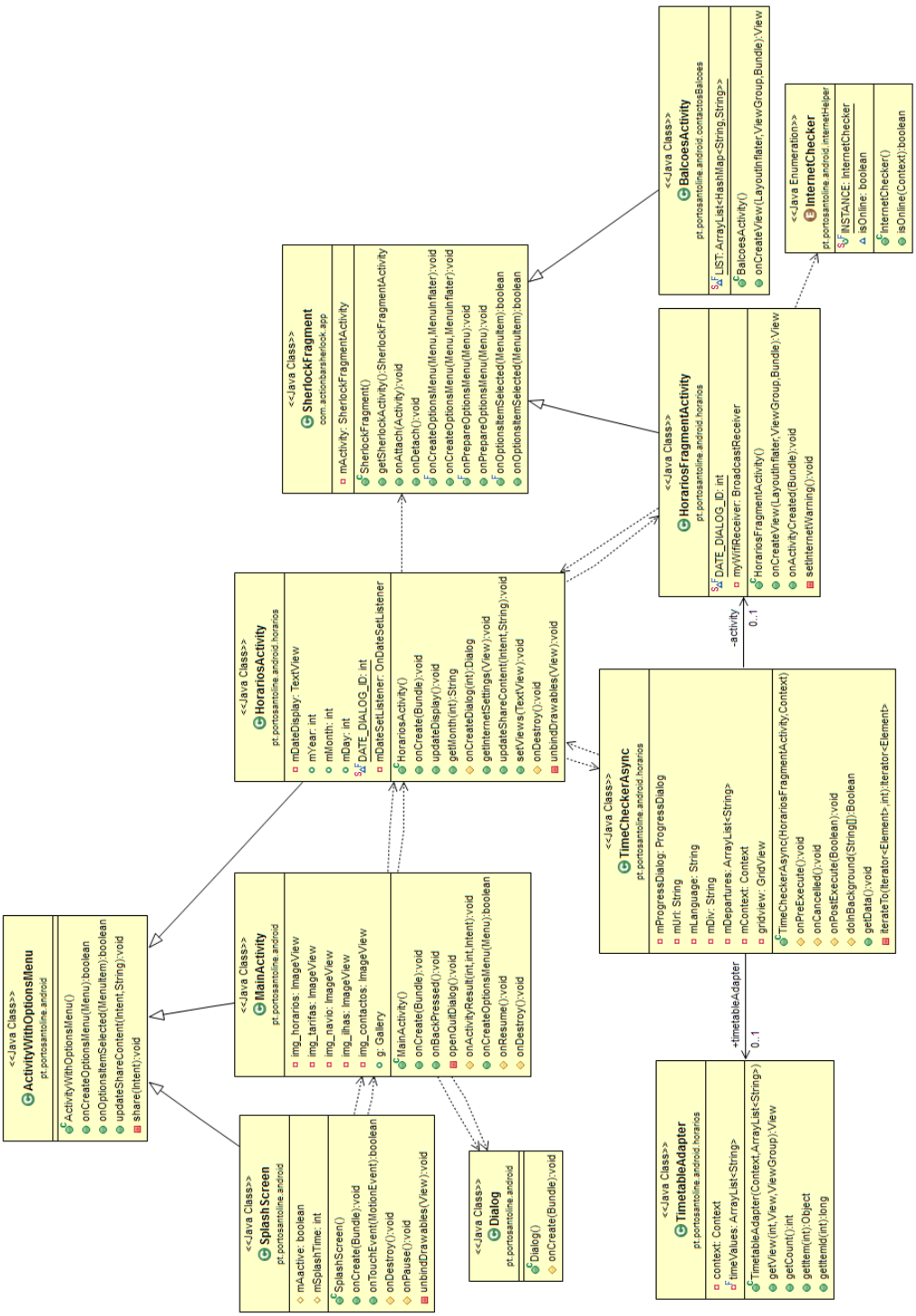
[22] Support Libray (2011), *http://developer.android.com/tools/extras/support-library.html*, acedido em 2011.

[23] ActionBarSherlock (2011), *http://actionbarsherlock.com*, acedido em 2011.

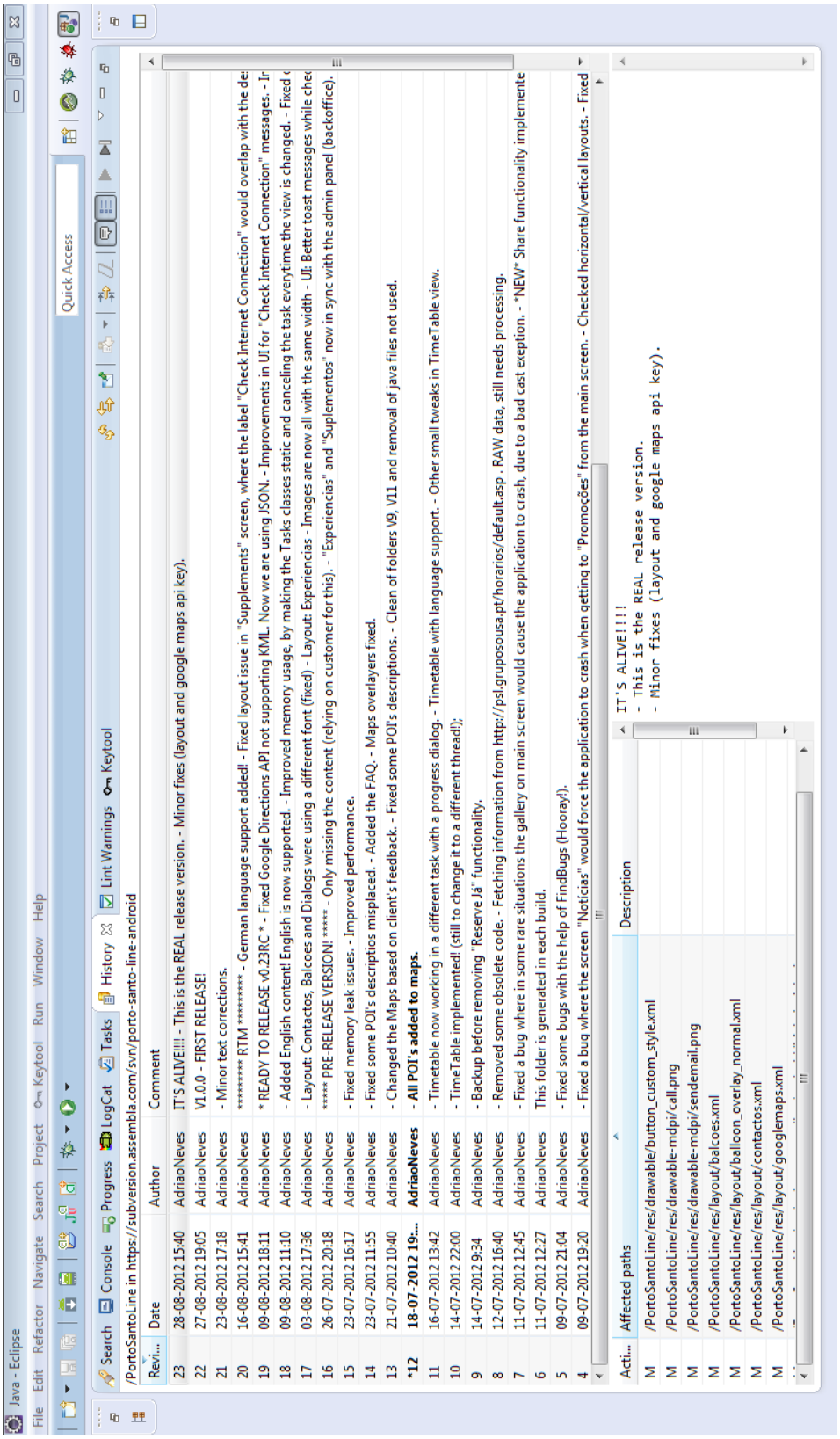
[24] Designing for Responsiveness (2011), *http://developer.android.com/guide/practices/responsiveness.html,* acedido em 2011.

9. ANEXOS

Anexo 1 – Diagrama de Classes “Consultar Horários”



Anexo 2 – Subversion: Histórico de versões



Anexo 3 – Casos de Teste

CASO DE TESTE	#2 – Consultar hora de saída a 17 de Setembro 2012
Descrição	Consultar a hora de saída do navio, a partir do cais do Funchal, do dia 17 de Setembro de 2012 (Segunda-Feira).
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida.
Caso de Utilização	Consultar horários
Requisitos	F8
Passos	1. Seleccionar “Horários”. 2. Seleccionar a “lupa”. 3. Introduzir a data e submeter. 4. Opção: Seleccionar o dia na tabela.
Resultados esperados	Utilizador deve indicar a hora: 08:00
Resultados obtidos	**Tarefa executada com sucesso.**
Observações	Dificuldades em encontrar a “lupa”.

Tabela 4 - Caso de teste #2: Consultar hora de saída a 17 de Setembro 2012

CASO DE TESTE	#3 – Consultar preço para 2 pessoas + 1 viatura em Janeiro
Descrição	A Porto Santo Line possui tarifas especiais para o transporte de motos e outras viaturas. Existem também pacotes especiais com estadia em Hotel incluída no preço da passagem. Neste teste, o utilizador deve consultar preço para 2 pessoas + 1 viatura em Janeiro.
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida. • Adobe Reader instalado.
Caso de Utilização	Consultar tarifas
Requisitos	F9, F10
Passos	1. Seleccionar “Tarifas”. 2. Seleccionar ”Pacotes c/ Viaturas”.
Resultados esperados	Utilizador deve indicar o preço: 120,00€
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Sem observações relevantes.

Tabela 5 - Caso de teste #3: Consultar preço para 2 pessoas + 1 viatura em Janeiro

CASO DE TESTE	#4 – Obter preço para a experiência “Passeios a cavalo”
Descrição	A Porto Santo Line, juntamente com os seus parceiros comerciais, oferecem um conjunto de “experiências” turísticas na ilha do Porto Santo. Para este teste, o utilizador deve obter o preço para

	a experiência “Passeios a cavalo”.
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida.
Caso de Utilização	Obter lista de experiências
Requisitos	F15, F16
Passos	1. Seleccionar “Porto Santo e Madeira”. 2. Seleccionar ”Experiências”.
Resultados esperados	Utilizador deve indicar o preço: 20,00€
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Apenas dois de seis utilizadores encontraram o botão “Experiências” na primeira tentativa.

Tabela 6 - Caso de teste #4: Obter preço para a experiência "Passeios a cavalo"

CASO DE TESTE	**#5 – Verificar se existe uma tabacaria a bordo**
Descrição	O utilizador deverá verificar se entre os vários serviços disponíveis a bordo do navio, existe uma tabacaria.
Configuração	• Aplicação aberta no menu principal.
Caso de Utilização	Consultar lista de serviços a bordo.
Requisitos	F11, F12
Passos	1. Seleccionar “Serviços a bordo”. 2. Navegar na lista até encontrar “Tabacaria”
Resultados esperados	Utilizador diz que existe uma tabacaria a bordo. Idealmente indica em qual pavimento do navio.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Sem observações relevantes.

Tabela 7 - Caso de teste #5: Verificar se existe uma tabacaria a bordo

CASO DE TESTE	**#6 – Localizar Terminal de passageiros no Funchal**
Descrição	Existem dois modos de obter a localização do terminal de passageiros ou de outro balcão, pelo mapa de qualquer ilha ou através do menu “Contactos e Balcões”. Neste teste é esperado utilizar o segundo método.
Configuração	• Aplicação aberta no menu principal.
Caso de Utilização	Localizar balcões no mapa
Requisitos	F24, F25.
Passos	1. Seleccionar “Contactos e Balcões”. 2. Navegar na lista até encontrar “Terminal de Passageiros”. Confirmar que é o terminal do Funchal. 3. Seleccionar o respectivo botão “Mapa”. 4. Dentro do mapa, seleccionar o ícone respectivo ao terminal de passageiros.
Resultados esperados	Utilizador consegue localizar o terminal correcto no mapa.

Resultados obtidos	Tarefa executada com sucesso.
Observações:	Dois em seis utilizadores não seleccionaram o POI após a abertura do mapa.

Tabela 8 - Caso de teste #6: Localizar Terminal de passageiros no Funchal

CASO DE TESTE	#7 – Obter rota entre posição actual e Hotel Lua Mar
Descrição	Para este teste, é necessário simular o cenário em que o utilizador está na ilha do Porto Santo e pretende descobrir qual a melhor rota entre a sua posição geográfica actual e o Hotel Lua Mar.
Configuração	• Aplicação aberta no menu principal. • Sinal GPS previamente obtido. • Ligação à Internet estabelecida.
Caso de Utilização	Obter rota entre posição actual e POI.
Requisitos	F18, F19, F20, F21, F22, F23.
Passos	1. Seleccionar “Porto Santo”. 2. Seleccionar “Mapa”. 3. Identificar o Hotel Lua Mar. 4. Seleccionar o Hotel Lua Mar. 5. Seleccionar “Obter rota”.
Resultados esperados	Utilizador identifica o respectivo hotel. Sistema calcula rota entre os dois pontos.
Resultados obtidos	Tarefa executada com sucesso.
Observações:	Dificuldade em descobrir como aceder ao mapa. A identificação do hotel, com alguns utilizadores, não foi intuitiva.

Tabela 9 - Caso de teste #7: Obter rota entre posição actual e Hotel Lua Mar

CASO DE TESTE	#8 – Telefonar para o armazém no Porto Santo
Descrição	Existe um armazém de carga no cais do Funchal e um no cais do Porto Santo. Cada armazém possui um número de telefone próprio. O utilizador neste teste deve entrar em contacto, através de chamada telefónica, com o armazém do Porto Santo.
Configuração	• Aplicação aberta no menu principal.
Caso de Utilização	Telefonar – Balcão
Requisitos	F26
Passos	1. Seleccionar “Contactos e Balcões”. 2. Identificar o armazém de carga do Porto Santo. 3. Seleccionar o botão “*telefonar*”, representado por um ícone de um auscultador.
Resultados esperados	Utilizador identifica o armazém de carga do Porto Santo. Utilizador associa o ícone “*auscultador*” com a acção “*telefonar*”.

	Utilizador inicia chamada seleccionando o botão com o ícone do auscultador. Sistema inicia a chamada telefónica.
Resultados obtidos	Tarefa executada com sucesso.
Observações:	Sem observações relevantes.

Tabela 10 - Caso de teste #8: Telefonar para o armazém no Porto Santo

CASO DE TESTE	**#9 – Consultar notícias**
Descrição	Em caso de avaria do navio, cancelamento de viagens, mudanças no horário de partidas, os clientes são informados (no mínimo) através da secção "Notícias" do site oficial da Porto Santo Line. A aplicação móvel é agora outro meio para a divulgação dos anúncios da empresa.
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida.
Caso de Utilização	Consultar notícias
Requisitos	F7
Passos	1. Seleccionar "Notícias" na "*Action bar*".
Resultados esperados	Utilizador associa o ícone "*envelope*" com a acção "*ler notícias*". Sistema obtém conteúdo do *site* oficial e apresenta.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Dois dos seis utilizadores não conseguiram identificar o ícone "Notícias" com a facilidade esperada.

Tabela 11 - Caso de teste #9: Consultar notícias

CASO DE TESTE	**#10 – Consultar promoções**
Descrição	As promoções da empresa são divulgadas nas redes sociais, no *site* oficial e agora também na aplicação móvel.
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida.
Caso de Utilização	Consultar promoções
Requisitos	F6
Passos	1. Seleccionar "Promoções".
Resultados esperados	Utilizador identifica o botão "Promoções" na zona de destaque do menu principal. Sistema obtém conteúdo do *site* oficial e apresenta.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Sem observações relevantes.

Tabela 12 - Caso de teste #10: Consultar promoções

CASO DE TESTE	**#11 – Enviar email para o "contact center"**

Descrição	A Porto Santo Line possui uma linha de apoio ao cliente, disponível para contacto directo através da aplicação. O utilizador neste teste deve encontrar o número de telefone e realizar a chamada directamente da aplicação.
Configuração	• Aplicação aberta no menu principal. • Ligação à Internet estabelecida. • Aplicação de email instalada e configurada.
Caso de Utilização	Enviar email para contact center.
Requisitos	F27, F29.
Passos	1. Seleccionar “Contactos e Balcões”. 2. Seleccionar “Contactos”. 3. Associar o ícone de envelope à acção “Enviar email”. 4. Seleccionar cliente de email. 5. Enviar email com o texto “Teste”.
Resultados esperados	Utilizador identifica o ícone e a acção correspondente. Utilizador selecciona aplicação “E-Mail”. Utilizador escreve e envia email. Sistema envia email e regressa à aplicação.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Mesmo ícone para duas acções diferentes: “*Ler notícias*” e “*Enviar email*”.

Tabela 13 - Caso de teste #11: Enviar email para *contact center*

CASO DE TESTE	**#12 – Partilhar “Promoções” no Facebook**
Descrição	Cada ecrã da aplicação é associado ao conteúdo de uma página do *site* oficial da empresa. A opção de partilha irá enviar o URL da respectiva página para a aplicação seleccionada. Neste caso de teste, o utilizador deverá partilhar o URL das promoções em vigor, através do Facebook.
Configuração	• Aplicação aberta no menu principal. • Ligação á Internet estabelecida. • Aplicação Facebook instalada.
Caso de Utilização	Partilhas nas redes sociais.
Requisitos	F27, F29, F33.
Passos	1. Seleccionar “Promoções”. 2. Identificar e seleccionar o ícone “Partilhar”. 3. Seleccionar aplicação Facebook. 4. Partilhar URL.
Resultados esperados	Utilizador partilha o URL das promoções.
Resultados obtidos	**Tarefa executada com sucesso.**
Observações:	Sem observações relevantes.

Tabela 14 - Caso de teste #12: Partilhar “Promoções” no Facebook