

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



FEUP

Resultados de MCDTs Mobile

Nuno Miguel Estrada Pereira Gouveia

Mestrado Integrado em Engenharia Informática e Computação

Orientador: António Miguel Pimenta Monteiro (Doutor)

22 de Julho de 2012

Resultados de MCDTs Mobile

Nuno Miguel Estrada Pereira Gouveia

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Doutor (a) Gabriel de Sousa Torcato David, Prof. Associado da FEUP

Arguente: Doutor (a) Carlos Manuel Azevedo Costa, Prof. Auxiliar da Universidade de Aveiro

Vogal: Doutor (a) António Miguel Pontes Pimenta Monteiro, Prof. Auxiliar da FEUP

22 de Julho de 2012

Resumo

Atualmente, a influência das Tecnologias de Informação na área da saúde é cada vez mais evidente, sendo a sua presença e importância cada vez maior. Nas consultas efectuadas por um doente e para que se possa fazer um diagnóstico preciso, os profissionais de saúde necessitam não só de analisar os seus sintomas, mas também de analisar e requisitar meios complementares de diagnóstico e terapêutica - *MCDTs*. O problema que motiva esta dissertação passa pela visualização e manipulação da representação digital dos resultados de *MCDTs* dos doentes, alojados na base de dados da rede interna da unidade hospitalar, sendo o grande desafio presente a gestão da enorme diversidade de formatos de representação dos mesmos. As soluções existentes atualmente no mercado não estão preparadas para lidar com a dinâmica do processo de diagnóstico, sendo acedidas essencialmente em dispositivos estáticos num ambiente *desktop*.

Neste sentido, esta dissertação pretende adicionar portabilidade à operação sobre os resultados de *MCDTs*, produzindo uma aplicação capaz de ser executada nas duas mais importantes plataformas para *tablets* - *Android* e *iOS*.

Para desenhar a melhor abordagem para a resolução do problema, começou por ser feita uma revisão bibliográfica, onde foi analisada a já existente aplicação *eResults* e seus concorrentes no mercado. Analizaram-se também as tecnologias a utilizar, nomeadamente para o desenvolvimento de uma aplicação móvel multi-plataforma, assunto que foi alvo de forte investigação e prototipagem.

A solução especificada contemplou quatro grandes componentes, nomeadamente a autenticação, a pesquisa de doentes com recurso a *webservices*, a visualização multimédia e as notificações, não esquecendo outras características como a eficiência, consistência, segurança e usabilidade. A implementação foi feita através de uma abordagem *web*, permitindo a aplicação adaptar-se de forma inteligente ao ambiente em que se encontra, ou seja, detetando a execução num ambiente *mobile*, são feitas as alterações necessárias aos elementos de cada página para que estes mantenham o nível de usabilidade requerido, utilizando para tal a *framework jQuery Mobile*.

Foi desenvolvido um protótipo da aplicação, não contendo apenas a componente de notificações, mas respondendo àqueles que eram os objetivos prioritários desta aplicação, ou seja, permite que os médicos tenham o poder da consulta e manipulação dos resultados de *MCDTs* literalmente nas suas mãos. Em simultâneo com este objetivo, o desenvolvimento de aplicações móveis *web* sai beneficiado, uma vez que esta dissertação apresenta uma forma deste ser feito transparentemente, sem haver necessidade de preocupação com o ambiente de execução.

Abstract

Nowadays, the influence of the Information Technologies in the health care area is each time more evident, being its presence and importance each time greater. When consulting a patient and to be able to do a precise diagnosis, the doctors need not only to analyse their symptoms, but also to analyse and solicit Supplementary Means of Diagnosis and Therapeutics - SMDTs. The problem that motivates this dissertation, is the visualisation and manipulation of the digital representations of the SMDTs results of the patients, stored in the health care unit network's database system, being the great challenge present the management of the enormous diversity of formats they are represented in. The current market solutions aren't prepared to deal with the dynamics of the diagnosis process, being accessed through static devices in a desktop environment.

By these means, this dissertation aims to add portability to the operations over SMDTs results, producing an application capable of being executed in the two most important platforms for *tablet's* - *Android* and *IOS*.

To design the best approach to the problem's resolution, it was made a bibliographic revision, analysing the application *eResults* as well as its competitors in the market, but the main focus were the technologies to use, namely for the development of cross-platform applications, a matter which had a great deal of both investigation and prototyping.

The solution specified in the development of the problem, contemplated four big components, namely the authentication, the patient's search using webservice, the multimedia visualization and the notifications, not forgetting other characteristics as efficiency, consistency, security and usability. The implementation was made through a web approach, allowing the application to adapt itself in an intelligent way to the environment it is in, that is, detecting the mobile environment and making the necessary substitutions to maintain the required usability level, using the *jQuery Mobile* framework.

Concerning the obtained results, it was developed a prototype of the application, missing only the notifications component, but meeting those that were the priority objectives of the application, that is, allowing the doctors to have the power of the visualisation and manipulation of the SMDTs results literally in their hands. Simultaneously with this objective the development of mobile web applications gets benefited, once this dissertation presents a way of doing it transparently, without the need to worry about the execution environment.

Agradecimentos

Em primeiro lugar gostaria de agradecer à Glintt Healthcare Solutions por me ter acolhido nas suas instalações para a realização desta dissertação, facultando todas as condições para o desenvolvimento da mesma. Agradeço a todas as pessoas que colaboraram de alguma forma no projeto, nomeadamente ao Eng. Nuno Ribeiro pela supervisão e interesse constante no projeto e muito especialmente ao Eng. Francisco Correia, responsável pela orientação da dissertação na empresa, por ter sempre encontrado tempo, mesmo quando não o tinha, para orientar, sugerir e ajudar no desenvolvimento do projeto.

Agradeço ao Doutor António Miguel Pimenta Monteiro ter aceite o convite para ser orientador desta dissertação, pela sua disponibilidade, pelos conselhos, sugestões e esclarecimentos prestados ao longo de toda a dissertação.

A todos os meus colegas e amigos que me acompanharam ao longo do meu percurso académico, com quem partilhei longas horas e projetos, agradeço a forma como direta ou indiretamente contribuíram para o meu sucesso e a todos deixo um forte abraço.

Agradeço à Tuna de Engenharia da Universidade do Porto, enquanto grupo por ter contribuído para a minha formação enquanto futuro engenheiro e enquanto pessoa mas também individualmente a todos os grandes amigos que me acompanharam ao longo de todo este tempo.

Guardo o maior agradecimento para a minha família, em especial aos meus avós, à minha madrinha, aos meus pais e ao meu irmão, por me terem sempre dado força para conseguir chegar mais longe.

Nuno Miguel Estrada Pereira Gouveia

*“Failure is simply the opportunity to begin again,
this time more intelligently”*

Henry Ford

Conteúdo

1	Introdução	1
1.1	Contexto e Enquadramento	1
1.2	Motivação, Objetivos e Contribuições	2
2	Revisão Bibliográfica	5
2.1	Solução atual - <i>eResults</i>	6
2.2	Outras Soluções no mercado	9
2.2.1	SiiMA - First Solutions, SA	9
2.2.2	Software Appolo - Confidentia	10
2.2.3	wClinicas - New Company	11
2.3	Tecnologias	11
2.3.1	Aplicações Móveis	13
2.3.2	Acesso a dados	15
2.3.3	Ambientes de desenvolvimento multi-plataforma	20
3	Desenvolvimento	35
3.1	O problema	35
3.2	Requisitos e funcionalidades	36
3.2.1	Atores	36
3.2.2	Requisitos da Solução	37
3.2.3	Casos de Uso	38
3.3	Arquitetura	42
3.3.1	Componentes físicos necessários	42
3.3.2	Módulos de software	43
3.3.3	Modelo de dados	46
3.3.4	Ligações e interações	48
3.4	Implementação	49
3.4.1	Escolha das tecnologias de implementação	49
3.4.2	Deteção de ambiente mobile	51
3.4.3	Módulo de visualização de multimédia	52
3.4.4	Módulo de acesso a dados	54
3.4.5	Módulo de pesquisa de doentes	55
3.4.6	Módulo de notificações	56
4	Operação e resultados	59
4.1	Operação do sistema e papéis dos atores	59
4.1.1	Autenticação	60
4.1.2	Documentos	60

CONTEÚDO

4.2	Utilização e benefícios	66
4.3	Testes e resultados de funcionamento	67
4.3.1	Testes executados com sucesso	67
4.3.2	Testes executados sem sucesso ou não executados	68
5	Conclusões e Trabalho Futuro	71
5.1	Conclusões	71
5.1.1	Satisfação dos Objetivos	72
5.2	Trabalho Futuro	72
A	Tabelas de casos de uso	73
A.1	Módulo de Autenticação	73
A.2	Módulo de pesquisa de pacientes	75
A.3	Módulo de visualização de multimédia	77
A.4	Módulo de notificações	82
B	Mockups de Interface	87
B.1	Módulo de Autenticação	87
B.1.1	Login	87
B.1.2	Logout	88
B.2	Módulo de pesquisa de pacientes	89
B.2.1	Pesquisa Simples	89
B.2.2	Pesquisa Detalhada	90
B.2.3	Pesquisa de linha de histórico	91
B.3	Módulo de visualização de multimédia	92
B.3.1	Visualiza Pdf	92
B.3.2	Visualiza Pdf	93
B.3.3	Visualiza Vídeo	94
B.3.4	Manipula PDF	95
B.3.5	Manipula Imagens	96
B.4	Módulo de notificações	97
B.4.1	Adiciona pedido de notificação	97
B.4.2	Consulta notificações	98
B.4.3	Consulta pedidos de notificações	99
B.4.4	Edita pedidos de notificações	100
C	Tabelas de testes de aceitação	101
C.1	Módulo de Autenticação	101
C.2	Módulo de Documentos	103
C.3	Módulo de notificações	112
	Referências	117

Lista de Figuras

2.1	Arquitetura física do eResults	7
2.2	O módulo de documentos do eResults, com os resultados apresentados em formato miniatura	8
2.3	Aplicação nativa para iPad (iCal)	13
2.4	Aplicação web para iPad (yahoo)	14
2.5	Estabelecimento da interação entre fornecedor e requisitante de um serviço	16
2.6	Arquitetura do ambiente de desenvolvimento multi-plataforma Titanium	22
2.7	Protótipo de suporte a multimédia do titanium	24
2.8	Arquitetura da <i>framework Rhodes</i>	28
2.9	Prototipo da aplicação final desenvolvido em Sencha Touch 2.0	30
2.10	Comparação entre ambientes multi-plataforma	31
3.1	Diagrama de Atores	37
3.2	Modelo de casos de uso do módulo de autenticação	39
3.3	Modelo de casos de uso do módulo de pesquisa de doentes	40
3.4	Modelo de casos de uso do módulo de visualização de multimédia	41
3.5	Modelo de casos de uso do módulo de notificações	42
3.6	Organização dos componentes físicos da aplicação	43
3.7	Modulos e componentes de software da solução	44
3.8	Modelo de dados da componente de notificações	46
3.9	Modelo de dados das componenetes do <i>eResults</i> utilizadas na aplicação	47
3.10	Diagrama de fluxo e interações entre módulos da aplicação	48
3.11	Código utilizado para a deteção do ambiente de execução	52
3.12	Código utilizado para a chamada do glinttViewer	53
3.13	Função da chamada ao <i>webservice</i> na camada de controlo	55
3.14	Interação da camada de visualização com a camada de controlo	55
3.15	Definição de uma “combo-box” ao nível da <i>view</i>	56
3.16	Atribuição do <i>plugin</i> “glinttCombo” a um <i>div</i>	56
3.17	Criação do elemento <i>HTML</i> respeitante ao ambiente <i>mobile</i>	57
3.18	Combo-box no ambiente <i>desktop</i> (à esquerda) e no ambiente <i>mobile</i> à (direita) . .	58
4.1	Barra de pesquisa simples do módulo de documentos e os seus elementos	60
4.2	Barra de pesquisa detalhada do módulo de documentos e os seus elementos	61
4.3	Apresentação dos resultados em formato lista	62
4.4	Apresentação dos resultados em formato thumbs	63
4.5	Apresentação do visualizador de <i>pdfs</i> e os seus respetivos elementos	64
4.6	Apresentação do visualizador de imagens e os seus respetivos elementos	65
4.7	Apresentação do visualizador de imagens e os seus respetivos elementos	66

LISTA DE FIGURAS

B.1	Mockup do login, correspondente ao A.1 (CU01)	87
B.2	Mockup do logout, correspondente ao A.2 (CU02)	88
B.3	Mockup da pesquisa simples, correspondente ao A.3 (CU03)	89
B.4	Mockup da pesquisa detalhada, correspondente ao A.4 (CU04)	90
B.5	Mockup da pesquisa de linha de histórico, correspondente ao A.5 (CU05)	91
B.6	Mockup da visualização de pdf, correspondente ao A.6 (CU06)	92
B.7	Mockup da visualização de imagens, correspondente ao A.7 (CU07)	93
B.8	Mockup da visualização de vídeos, correspondente ao A.8 (CU08)	94
B.9	Mockup da manipulação pdfs, correspondente ao A.9 (CU09)	95
B.10	Mockup da manipulação imagens, correspondente ao A.10 (CU10)	96
B.11	Mockup do menu de configurações de uma nova notificação, correspondente ao A.11 (CU11)	97
B.12	Mockup da consulta de notificações, correspondente ao A.12 (CU12)	98
B.13	Mockup da consulta de pedidos notificações, correspondente ao A.13 (CU13)	99
B.14	Mockup da edição de pedidos notificações pendentes, correspondente ao A.14 (CU14)	100

Lista de Tabelas

A.1	CU01	73
A.2	CU02	74
A.3	CU03	75
A.4	CU04	76
A.5	CU05	76
A.6	CU06	77
A.7	CU07	78
A.8	CU08	79
A.9	CU09	80
A.10	CU10	81
A.11	CU11	82
A.12	CU12	83
A.13	CU13	84
A.14	CU14	85
A.15	CU15	86
C.1	TA01	101
C.2	TA02	102
C.3	TA03	102
C.4	TA04	103
C.5	TA05	104
C.6	TA06	105
C.7	TA07	106
C.8	TA08	107
C.9	TA09	107
C.10	TA10	108
C.11	TA11	109
C.12	TA12	110
C.13	TA13	111
C.14	TA14	112
C.15	TA15	113
C.16	TA16	114
C.17	TA17	115
C.18	TA18	116

LISTA DE TABELAS

Abreviaturas e Símbolos

Android	Sistema operativo para dispositivos móveis desenvolvido pela Google Inc.
API	Application Programming Interface
Browser	Aplicação que permite aos utilizadores a visualização de páginas web
CRUD	Create, Read, Update and Delete
eResults	Aplicação web de consulta e manipulação de MCDTs desenvolvida pela Glintt-HS
framework	Conjunto de Bibliotecas de código que funcionam como uma abstracção, permitindo ao programador realizar uma determinada tarefa
FTP	File Transfer Protocol
HD	High definition (Alta definição)
HTTP	Hypertext Transfer Protocol
iOS	Sistema operativo para dispositivos móveis desenvolvido pela Apple Inc.
LAN	Local Area Network (Rede de Área Local)
MCDT	Meios Complementares de Diagnóstico e Terapêutica
REST	Representational state transfer
RMI/IIOP	Java Remote Method Invocation (RMI) interface over the Internet Inter-Orb Protocol
RSS	Rich Site Summary
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
TI	Tecnologias da Informação
Tablet	Dispositivo móvel, maior que um smartphone, controlado através de um ecrã tátil
URI	Uniform Resource Identifier
URL	Uniform Resource Locator - endereço de um recurso, disponível numa rede
VM	Virtual Machine (Máquina Virtual)
W3C	World Wide Web Consortium
WWW	<i>World Wide Web</i>

Capítulo 1

Introdução

Este documento tem como principal objetivo a descrição do projeto “Resultados de *MCDTs* Mobile” - uma aplicação para *tablets* nas plataformas *Android* e *iOS* para visualização e manipulação de resultados de *MCDTs*.

1.1 Contexto e Enquadramento

Este projeto está inserido na área da Saúde, mais propriamente na disponibilização de soluções informáticas para o auxílio da actividade de prestação de cuidados de Saúde.

Nos dias de hoje, observamos cada vez mais uma forte dependência de muitos serviços nas TI, que é o culminar de um esforço que tem vindo a ser feito pelas instituições públicas e privadas para a informatização dos seus serviços, e a área da Saúde não é exceção. Nesta área, o objetivo deste esforço passa por conseguir uma melhor interligação entre os setores de uma unidade hospitalar, bem como um armazenamento eficiente da informação, que permita o rápido acesso e pesquisa da mesma. A quantidade de informação existente, relativa a um doente, já não é possível guardar sobre forma “em papel” utilizada anteriormente e torna-se necessária a utilização de um Processo Clínico Electrónico - PCE [Gli09].

Diariamente numa Unidade de Saúde, os médicos ao consultarem um doente e para que possam fazer um diagnóstico preciso, necessitam não só de analisar os seus sintomas, mas também de recorrer a dados do seu passado clínico e analisar ou requisitar Meios Complementares de Diagnóstico e Terapêutica - *MCDTs*.

O problema que motiva esta dissertação é a visualização e manipulação dos resultados dos *MCDTs*, por parte dos médicos. Estes resultados possuem uma enorme diversidade de representação podendo ser desde imagens como Raio-X, vídeos por exemplo resultantes de ultrassonografias, texto como o resultado de análises sanguíneas ou análises gráficas que permitam visualizar a evolução de um determinado parâmetro.

Este problema conhece já várias soluções, que falham essencialmente na portabilidade, entre as quais o *eResults* - uma aplicação web de apresentação de resultados de *MCDTs* sobre os formatos supra-referidos, desenvolvida pela *Glintt-HS*. A solução a desenvolver deverá ser uma aplicação para Tablets *Android* e *iOS* que funcionará como um complemento desta aplicação, adicionando portabilidade à mesma, fazendo com que esta se torne mais dinâmica e inteligente na interação com estes dispositivos.

Esta dissertação foi proposta e realizada na *Glintt-HS*, uma das 8 empresas que compõem o grupo Glintt, dedicada ao desenvolvimento de aplicações na área da Saúde. É uma empresa líder no setor, estando presente nos mais importantes hospitais e clínicas nacionais, contando também com alguma projeção em mercados internacionais.

1.2 Motivação, Objetivos e Contribuições

Com o advento dos dispositivos móveis, a portabilidade e disponibilidade da informação tornou-se uma realidade ao nosso alcance. O principal objetivo desta dissertação passa pela aplicação destas características inerentes aos dispositivos móveis, ao problema da consulta e manipulação dos resultados dos *MCDTs*, dotando os médicos de toda a informação necessária para que possam fazer um diagnóstico preciso a um doente, em qualquer parte do hospital. Esta aplicação deverá, entre outros, dar resposta aos seguintes pontos:

- Permitir a consulta de resultados de *MCDTs* de um determinado doente;
- Permitir a manipulação desses mesmos resultados;
- Requerer notificações da entrada de resultados de *MCDTs* no sistema, via *SMS*, *e-mail* ou na própria aplicação;
- Ser desenvolvida em simultâneo para *Android* e *iOS*;

Os utilizadores desta aplicação, sendo especialistas na área da Saúde, por norma não possuem grandes competências a nível informático, o que faz com que a usabilidade seja um ponto chave desta aplicação. A aplicação deverá ter uma interface e utilização apelativa e intuitiva de forma a tornar a curva de aprendizagem dos utilizadores praticamente inexistente, eliminando ao mesmo tempo os problemas inerentes à mudança de tecnologia nomeadamente o custo de aprendizagem da utilização de um novo sistema.

Para a concretização dos dois primeiros objetivos será necessário fazer um estudo das soluções existentes atualmente para resolver o problema entre as quais se encontra o *eResults*. Será também importante perceber como será possível utilizar as características dos dispositivos e aplicações móveis para tornar estas duas tarefas mais fáceis, eficientes e integradas no processo de diagnóstico.

Para dar resposta ao quarto objetivo, uma vez que não é possível no tempo disponível para realizar a dissertação, desenvolver uma aplicação nativa para cada sistema operativo, será necessário fazer um estudo de vários ambientes de desenvolvimento multi-plataforma existentes no mercado.

Introdução

Por último para concretizar o terceiro objetivo será necessário fazer um estudo da melhor forma de implementar este tipo de notificações, nomeadamente da tecnologia de notificações assíncronas, *push notifications*.

Este projeto apresenta diversas contribuições para a área da saúde, em especial para as TI aplicadas a esta área. A primeira e mais evidente é a capacidade de oferecer portabilidade aos dados (*MCDTs*) através da utilização de dispositivos móveis. Esta capacidade irá trazer muito mais celeridade, eficiência e rigor aos médicos no diagnóstico dos doentes. Os dispositivos móveis não irão conferir apenas portabilidade aos dados, mas também uma manipulação muito mais intuitiva e poderosa.

A gestão da enorme diversidade de conteúdos multimédia é um conteúdo inovador nomeadamente em dispositivos móveis, sendo uma tarefa que exige um comportamento inteligente por parte da aplicação. Tal como supra-referido, estes conteúdos não serão apenas imagens e vídeos, mas também texto e análises gráficas. Aqui é inserido ainda outro conteúdo inovador que passa não só pela importação de dados num formato pronto para gerar análises gráficas, mas também pela capacidade de gerar gráficos evolutivos de determinados conteúdos analíticos.

A forma como os dados serão armazenados e como serão requeridos pela aplicação é outro conteúdo inovador. Ao pedir a informação de um determinado doente, a aplicação não pode simplesmente carregar de imediato toda a informação do doente, uma vez que isso seria uma tarefa bastante morosa devido ao grande volume de dados inerente aos conteúdos multimédia. Assim, a aplicação tem que ser capaz de mostrar aos utilizadores a informação disponível e a sua pré-visualização através da “meta-informação” que carrega inicialmente, e apenas carregar realmente os dados quando são necessários ou seja, quando são requeridos pelo utilizador. No caso dos vídeos para evitar carregar grandes volumes de informação pode ser utilizada a técnica de streaming de dados.

O grande desafio desta dissertação passa pela combinação de técnicas e tecnologias de manipulação de dados que serão utilizadas para lidar com estes conteúdos.

Introdução

Capítulo 2

Revisão Bibliográfica

Um dos grandes conceitos por detrás do problema desta dissertação é o da apresentação de resultados ou *reporting*. Com a evolução tecnológica exponencial que temos assistido ao longo dos últimos anos e a informatização dos sistemas, a quantidade de informação que é gerada e armazenada em bases de dados é enorme. A maior parte do valor desta informação não é gerado quando é inserida nas bases de dados, mas quando é retirada da mesma e utilizada. O grande desafio da apresentação de resultados, que também pode ser designado de *Business intelligence*, é ser capaz de compreender a informação contida nas bases de dados, transformá-la e apresentá-la aos utilizadores finais num formato que estes entendam e que lhes forneça informação útil para que possam tomar decisões. [WW07] Aplicando este conceito ao problema em questão, é necessário que toda a informação dos resultados de *MCDTs* presente nas bases de dados das Unidades Hospitalares, nos diferentes formatos em que são armazenados, seja disponibilizada aos profissionais da saúde, em forma de conteúdos multimédia que lhes permitem analisar e retirar conclusões sobre os resultados de *MCDTs* e realizar um diagnóstico preciso.

Para abordar este problema, neste capítulo serão analisadas em primeiro lugar as soluções atuais para o problema da visualização e manipulação de *MCDTs* presentes no mercado, dando especial ênfase à solução desenvolvida na *Glintt-HS* - o *eResults*.

Seguidamente serão também analisadas as tecnologias a utilizar no desenvolvimento da aplicação. Serão analisados os conceitos de aplicação nativa e aplicação web (no contexto móvel) ao que se seguirá a análise dos mais relevantes ambientes de desenvolvimento multi-plataforma.

Por fim serão analisadas as possíveis formas de acesso a dados, estando entre estas o *WCF*.

De notar que, embora este facto não seja necessariamente de carácter restritivo ou impositivo, é sabido que a *Glintt-HS* tem preferência por tecnologias e plataformas *Microsoft*, possuindo um “know-how” muito alargado em termos de tecnologias *.NET* pelo que as alternativas que estiverem mais próximas deste domínio serão à partida privilegiadas.

2.1 Solução atual - *eResults*

De forma a melhor entender o problema a ser resolvido, é importante entender qual a solução atual para o problema, quais as suas virtudes e fraquezas, de forma a ser possível perceber quais dessas características devem ser mantidas, quais devem ser desenvolvidas e ainda quais as que devem ser acrescentadas.

Numa unidade de Saúde, existe um grande número de laboratórios que produzem resultados de *MCDTs*, desde Imunohematologia a Patologia Clínica entre muitos outros. Estes serviços possuem meios de realizar os exames necessários e sistemas que permitem visualizar o resultado desses exames. Este tipo de sistemas por norma são independentes entre si e possibilitam a visualização dos dados respetivos aos exames produzidos no próprio serviço. O seu maior problema passa pelo facto de não permitirem interoperabilidade entre si, tornando mais difícil a tarefa de disponibilizar os seus resultados a quem deles necessita, de uma forma eficaz e em tempo útil.

Neste âmbito, nasceu o *eResults*, uma aplicação web de apresentação de resultados de *MCDTs* sobre os formatos de imagem, vídeo, texto e análises gráficas. Esta aplicação permite não só a agregação destes resultados numa única plataforma como também uma consulta em tempo útil dos mesmos. [Mor08] Esta aplicação pode receber dados dos diversos sistemas que dão origem aos resultados de *MCDTs*, independentemente do desenvolvimento dos sistemas de onde provêm estar ou não à responsabilidade da *Glintt-HS*. Cada aplicação é responsável pelo envio dos resultados que produz para o *eResults*. [Cam08] Sendo uma aplicação web, o *eResults* está acessível a partir de qualquer máquina na unidade de Saúde, que tenha acesso à rede interna da mesma e que possua um *web browser*.

Na figura 2.1 é possível observar a arquitetura física do *eResults*.

Podemos observar que existem essencialmente três grandes componentes na arquitetura física:

- **Sistemas geradores de *MCDTs*** - Estes são sistemas existentes nos diferentes laboratórios, que permitem a realização de um determinado exame. Depois de gerarem o resultado do exame, reencaminham o mesmo para o servidor do *eResults*
- **Web service** - Aqui é feito o processamento da informação. A aplicação responde a qualquer pedido feito por uma máquina cliente, tratando da interação com a base de dados e mostrando ao cliente respetivo um resultado de *MCDT* ou um conjunto destes, conforme tenha sido requisitado.
- **Máquinas cliente** - Aqui é feita a visualização dos resultados de *MCDTs*. As máquinas cliente são terminais com acesso à rede interna do hospital, onde podem aceder à interface do *eResults*, via *browser*. Sempre que o utilizador deseja ter acesso a um determinado resultado de *MCDT* é enviado um pedido ao servidor que retorna o resultado do mesmo.

O *eResults* possui 8 módulos distintos, acessíveis na parte superior do ecrã:

- **Linha de histórico**- Permite uma visualização dos resultados de um determinado doente ao longo do tempo;

Revisão Bibliográfica

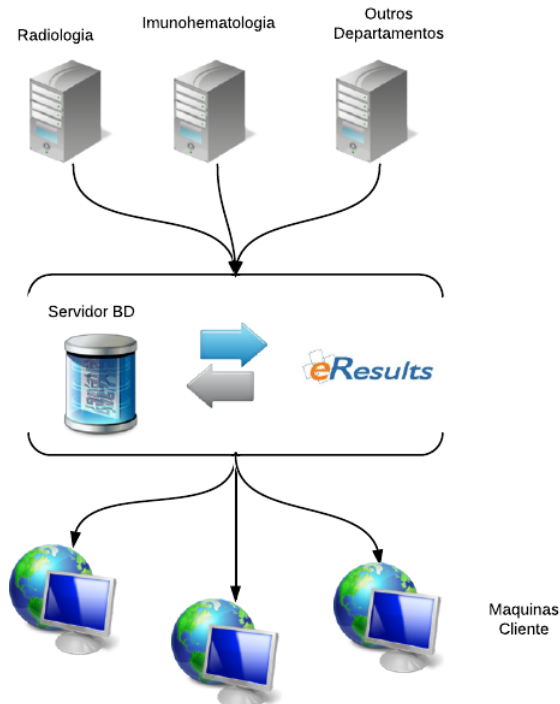


Figura 2.1: Arquitetura física do eResults

- **Documentos**- Permite a pesquisa e exploração dos resultados de exames de um determinado doente;
- **Resultados analíticos**- Permite a visualização de dados analíticos em formatos de tabelas ou gráficos;
- **Filtros**- Permite ao utilizador configurar filtros que podem ser usados em todas as pesquisas;
- **Arquivador**- Permite a indexação de documentos à informação relativa a um doente;
- **Indicadores**- Mostra indicadores de gestão sobre acessos, utilização e monitorização da informação;
- **Utilizador**- Permite editar a informação pessoal do utilizador da aplicação;
- **Administrador**- Para a configuração de acessos, permissões e utilizadores;

Destes módulos, interessa aprofundar um pouco a análise aos módulos de linha de histórico, documentos e resultados analíticos, que deverão ser implementados na aplicação móvel.

Linha de histórico

Este módulo permite ao técnico de saúde acompanhar o percurso clínico do doente de uma forma mais eficaz, dando-lhe acesso a todos os resultados de exames realizados pelo mesmo, numa ordem cronológica.

Os exames são dispostos num formato de lista, que pode ser condensada por episódio clínico ou simplesmente cada exame atómicamente. Cada item da lista fornece um link direto para o resultado em questão, abrindo a componente de visualização do mesmo. A visão da linha de histórico pode ser feita a nível anual, mensal, semanal ou até diário.

Documentos

Este módulo permite a visualização dos resultados disponíveis para cada doente, com excepção dos resultados analíticos.

Ao pesquisar um doente, são obtidos todos os seus resultados de exames, podendo ser apresentados no formato de lista ou miniaturas (uma pré-visualização do resultado), de tamanho manipulável, como é possível observar na figura 2.2.



Figura 2.2: O módulo de documentos do eResults, com os resultados apresentados em formato miniatura

Ao fazer duplo clique sobre um determinado elemento, é aberto o seu modo de visualização. Se o resultado for uma imagem, esta pode ser manipulável e nesse caso o utilizador pode rodar a imagem, aumentar e diminuir o seu tamanho. É possível a qualquer momento imprimir a imagem ou repor o seu estado original. Se o resultado for composto por várias imagens, todas são apresentadas e é possível navegar entre as mesmas.

Por último é importante referir que uma vez que o *eResults* é uma aplicação *web*, podem existir formatos de resultados que não são suportados pelo *browser*, sendo que nesse caso a aplicação pede ao utilizador que faça *download* do resultado e o consulte fora da aplicação.

Resultados Analíticos

Neste módulo podem ser visualizados os resultados analíticos. Estes resultados chegam ao *eResults* como um conjunto de dados sobre indicadores obtidos num determinado exame. Cada exame pode ser visto individualmente na forma tabular, que para além de mostrar o valor do indicador, mostra também o seu estado, ou seja se está dentro dos valores normais, abaixo ou acima dos mesmos.

Caso existam vários exames incidentes sobre o mesmo indicador, é possível fazer uma análise gráfica, onde é mostrada a evolução do indicador ao longo do tempo.

Desta análise, pode-se concluir que existem bastantes funcionalidades do *eResults* que podem ser aplicadas na aplicação para *tablets*. No entanto alguns pontos serão melhorados ou mesmo alterados, nomeadamente:

- O tamanho utilizado para os menus da aplicação deve ser repensado, para não desperdiçar área de ecrã. Deverá ser utilizado por exemplo um determinado gesto para mostrar os menus;
- A forma de apresentação da linha de histórico será repensada para poder aproveitar as características dos dispositivos, de forma a ser manipulável de uma forma mais intuitiva;
- Em relação à parte dos documentos, a sua apresentação em formato de lista deverá ser feita de uma forma, que o tamanho de cada elemento possibilite ao utilizador a sua escolha sem dificuldade. A forma privilegiada de apresentação dos resultados será em formato miniaturas, sendo o seu tamanho manipulável com um simples gesto de *zoom in* ou *zoom out*;

Outras alterações deverão ser feitas e algumas funcionalidades adicionadas, quer seja no tempo útil desta dissertação, ou após a mesma.

2.2 Outras Soluções no mercado

Para além do *eResults* existem outras soluções no mercado, com funcionalidades semelhantes, que serão apresentadas de seguida. A pesquisa restringiu-se ao mercado nacional, uma vez que é o principal mercado onde o *eResults* compete e é a realidade onde este projeto se insere.

2.2.1 SiiMA - First Solutions, SA

O *SiiMA* é uma *suite* de produtos concebida para a gestão de *MCDTs* desenvolvida pela *First Solutions, SA*. À semelhança do *eResults* é uma aplicação web que funciona como uma plataforma base que é apoiada por módulos verticais das diversas especialidades, nomeadamente:

- **RIS** - destina-se à gestão da informação de exames realizados nos serviços de Radiologia e Imagiologia;
- **Oftalmologia** - destina-se à gestão de todo o ciclo de operações relacionadas com o serviço de oftalmologia, não só dos exames realizados mas também de outros dados acerca das consultas;

- **Cardio** - destina-se à gestão da informação de exames realizados nos serviços de Cardiologia. Está dividido em cinco sub-módulos, nomeadamente: Electrocardiografia, Ecocardiografia, Electrofisiologia, Hemo e UCIC/Internamento;
- **Gastro** - destina-se à gestão da informação de exames realizados nos serviços de Gastroenterologia;
- **MN** - destina-se à gestão da informação de exames realizados nos serviços de Medicina Nuclear;
- **MFR** - destina-se à gestão da informação de exames e tratamentos realizados nos serviços de Medicina Física e reabilitação. Suporta todo o ciclo de um tratamento desde a prescrição de um fisiatra até à execução do tratamento por um terapeuta;
- **Neuro** - destina-se à gestão da informação de exames realizados nos serviços de Neurofisiologia;
- **Pneumo** - destina-se à gestão da informação de exames realizados nos serviços de Pneumologia;

Todos estes módulos estão integrados com os equipamentos nos serviços referidos. Esta aplicação foi desenvolvida para ter uma integração completa, não sendo capaz de se integrar com outras soluções. O *SiiMA* possui ainda um módulo de requisições centralizado de *MCDTs*. [FS11]

Não é conhecido nenhum tipo de integração específica com *tablets*, mas uma vez que é uma aplicação web pode ser utilizada nos mesmos, com o prejuízo de não existir a adaptação da organização da informação para as especificidades destes aparelhos.

2.2.2 Software Appolo - Confidentialia

O *Appolo* é um sistema de gestão global para laboratórios, desenvolvido pela *Confidentialia*. É uma ferramenta abrangente, permitindo gerir todas as actividades do laboratório num único sistema, desde a produção de resultados de exames até a gestão financeira. Tem a capacidade de se adaptar a diferentes necessidades que os diferentes laboratórios possam ter, permitindo também a integração com sistemas e equipamentos externos.

Esta solução funciona num ambiente distinto do *eResults*, mas o seu estudo deve-se ao facto de ser uma solução com uma presença muito forte no mercado das análises e possuir também formas de visualização dos resultados por si gerados.

O *Appolo* está dividido em vários módulos aplicativos, dos quais, para o âmbito deste estudo, se salientam os seguintes:

- **Visualização de Resultados**- Este módulo pode ser também designado como “central de dados”. Este é o local onde convergem todos os dados da aplicação. É possível consultar todo o historial de um determinado doente, verificar o estado de todos os pedidos, consultar o resultado dos mesmos, saber se já foram faturados, quem os realizou e a data e hora a

que foram realizados. Pode também ser feito um pedido de repetição de uma determinada colheita, ou inclusivamente a anexação de mensagens a um determinado resultado. Neste caso os resultados são sempre em formato de texto o que não causa a dificuldade presente no *eResults* de lidar com diferentes formatos de multimédia;

- **Mobilidade**- Em termos de interactividade com sistemas móveis, este sistema permite o envio de *sms* ou *email* de notificação da disponibilidade de resultados ou de repetição de colheitas. É possível também o envio de resultados completos, a pedido do doente ou do médico.
- **Digitalização de Documentos**- Qualquer documento em papel (do laboratório) pode ser digitalizado e anexado a uma determinada inscrição. Se o documento tiver um código de barras identificativo, o sistema é capaz de reconhecer de que doente se trata e indexar de forma inteligente o documento à ficha do doente respetivo; [Apl11]

2.2.3 wClinicas - New Company

O *wClínicas* é uma aplicação de gestão de consultórios, destinada a clínicas de pequena e média dimensão, desenvolvida pela *New Company*. É uma ferramenta integrada de gestão do consultório não se limitando à parte clínica, tendo também suporte para toda a faturação necessária ao bom funcionamento do mesmo, eliminando a necessidade de utilizar diferentes ferramentas.

O *wClínicas* permite gerir informação detalhada sobre cada doente, incluindo o seu historial clínico. Permite o armazenamento e consulta de relatórios médicos e exames laboratoriais, com a possibilidade de associação de imagens aos mesmos. É possível também uma integração de toda esta informação entre várias clínicas do mesmo grupo, aumentando a comodidade para os utilizadores do sistema e para os próprios doentes.

Este sistema possui ainda diferentes tipos de alertas via *SMS* ou *email* de consultas, aniversários e qualquer outro tipo desejado. Mais recentemente, foi adicionada a funcionalidade de prescrição electrónica de *MCDTs*. [NCSGdI11]

Esta ferramenta não é um concorrente direto do *eResults*, uma vez que não é direccionada para o mesmo mercado (o *eResults* procura atingir essencialmente os grandes hospitais e clínicas). Em relação ao principal objecto de estudo (consulta e manipulação de resultados *MCDTs*) é bastante limitada, não suportando a enorme diversidade de multimédia inerente aos resultados de *MCDTs*.

2.3 Tecnologias

Primariamente a serem analisadas as tecnologias envolvidas no desenvolvimento desta dissertação, importa analisar uma questão essencial: o que é um dispositivo móvel.

Um dispositivo ou sistema móvel, é um aparelho portátil com capacidade de processamento, capaz de nos ajudar a desempenhar tarefas importantes ou simplesmente de nos proporcionar entretenimento. Neste âmbito, podemos identificar como sendo dispositivos móveis, uma larga gama

de aparelhos nomeadamente telemóveis, computadores portáteis, consolas portáteis, sistemas de posicionamento global (*GPS*), leitores de áudio, *smartphones*, *tablets* entre outros. Os dois últimos são os que mais têm evoluído recentemente, devido a uma grande aposta das empresas da área nestes dispositivos, e estão na linha da frente da evolução tecnológica. Importa no entanto perceber o que é cada um deles e quais as suas diferenças.

Os *Smartphones* são a nova e mais avançada geração de telemóveis (3G+). Os primeiros *smartphones* surgiram da combinação de PDAs - *Personal digital Assistants*, com telemóveis. O seu objetivo é juntar num único dispositivo, várias funcionalidades desde as básicas dos telemóveis como chamadas e mensagens de texto, a reprodução e gravação de áudio e vídeo, consulta e manipulação de ficheiros, até jogos, sendo que a sua interface geralmente é através de um ecrã tátil e não um teclado. [Enc12b] A conectividade dos *smartphones* é um ponto muito forte dos mesmos, permitindo o acesso por exemplo a redes *Wi-Fi* e de banda larga móvel. Embora esta capacidade já estivesse disponível em alguns telemóveis 2G, o tamanho do ecrã relativamente maior dos *smartphones* (à volta das 3 polegadas), faz com que o acesso à *Internet* seja uma tarefa simples e agradável. Estes dispositivos possuem um sistema operativo altamente complexo e desenvolvido (em permanente atualização), que tem atraído cada vez mais programadores a desenvolver aplicações para estes dispositivos, aumentando ainda mais o valor potencial dos mesmos.

Os *tablets* são uma nova categoria de dispositivos ainda mais recente. Possuem sistemas operativos semelhantes aos *smartphones* e em termos de funcionalidades não acrescentam praticamente nada em relação aos mesmos. O que realmente difere é o tamanho destes dispositivos. O ecrã, bastante maior (na ordem das 7 a 10 polegadas que chega a ser três vezes superior), permite uma experiência de interação completamente diferente, como por exemplo o processamento de texto, que se torna muito mais confortável. O facto de ser um dispositivo maior, permite também melhorias em termos de *hardware*, sendo a diferença mais visível, uma capacidade de processamento muito superior. Estas diferenças abrem os horizontes em termos do tipo de aplicações que podem ser desenvolvidas para estes dispositivos, alargando o seu espectro. Esta área tem assistido e continuará a assistir a uma evolução tremenda, que permitirá o desenvolvimento de cada vez mais e melhores aplicações.

O que podemos concluir para o desenvolvimento desta dissertação é que embora tanto o *Android* como o *iOS* possuam o mesmo sistema operativo para os dois tipos de dispositivos, o desenvolvimento para ambos é bastante diferente. O desenvolvimento para *tablets*, que é o dispositivo alvo para esta dissertação, deve ser pensado de forma a maximizar as capacidades destes dispositivos em termos de aproveitamento da área do ecrã para criar uma interface e interação com o utilizador, de alta qualidade e em termos do processamento que estes dispositivos são capazes, fazendo com que o ganho em portabilidade em relação aos computadores, não signifique necessariamente uma perda de funcionalidades.

Serão discutidas agora as tecnologias envolvidas no desenvolvimento desta dissertação.

2.3.1 Aplicações Móveis

As aplicações móveis assumem cada vez mais um papel preponderante no mercado tecnológico. Este é um mercado simbiótico uma vez que quantas mais aplicações existirem para um determinado sistema operativo, mais apelativo este se torna para os utilizadores e quantos mais utilizadores utilizarem um determinado sistema operativo, mais apelativo este se torna para quem desenvolve aplicações para o mesmo, num ciclo de crescimento contínuo. Atualmente este mercado conta com mais de um milhão de aplicações [Mob12a]. Mas então o que é uma aplicação móvel? Não é mais do que uma aplicação desenvolvida para ser executada em um ou mais dispositivos móveis, quer sejam *smartphones*, *tablets* ou qualquer outro dispositivo móvel. [Enc12a] Essas aplicações podem ser nativas ou web (disponíveis online). Irão ser analisadas as diferenças entre ambas de seguida.

2.3.1.1 Aplicações Nativas

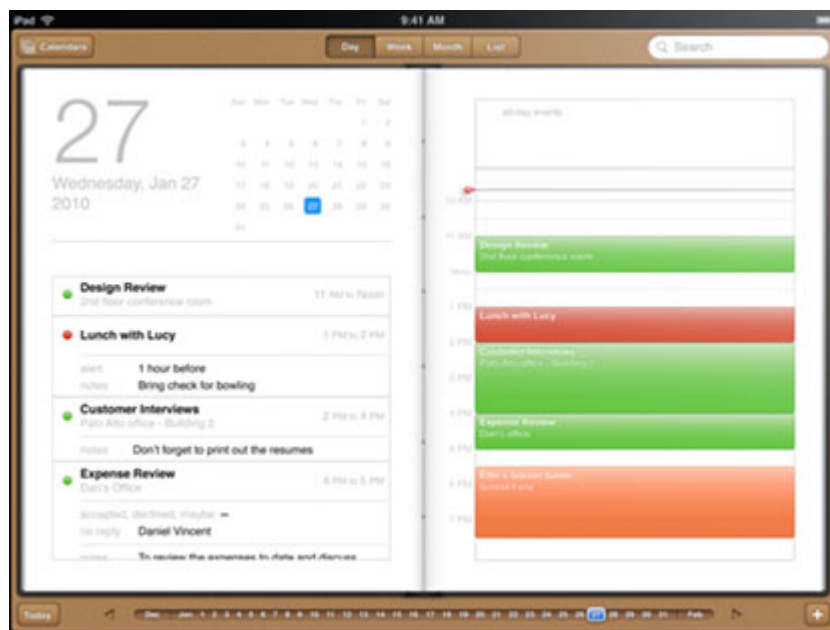


Figura 2.3: Aplicação nativa para iPad (iCal)

Uma aplicação nativa é desenvolvida para ser executada num sistema operativo específico, como por exemplo o *iOS* ou o *Android*. Estas aplicações tiram o máximo partido das capacidades dos dispositivos e das suas ferramentas, como por exemplo o acelerómetro, a câmara fotográfica ou a agenda telefónica. Pode-se considerar que a principal vantagem das aplicações nativas seja a sua eficiência, por serem executadas diretamente no aparelho. A sua maior popularidade faz com que os utilizadores estejam à espera que os controlos tenham um determinado aspeto (ex: botões e menus no *iOS* e *Android*), tendo essa vantagem em relação às aplicações web.

Por outro lado, a grande maioria das aplicações existentes atualmente, necessita de aceder a informação presente na internet, deixando as aplicações à mercê da latência da rede. Uma grande

desvantagem das aplicações nativas, do ponto de vista do desenvolvimento, surge quando uma aplicação pretende ter mais do que um sistema alvo, fazendo com que seja necessário desenvolver uma aplicação por cada sistema, ou recorrer a ambientes de desenvolvimento multi-plataforma, correndo o risco de ter perdas na eficiência da aplicação.

2.3.1.2 Aplicações Web

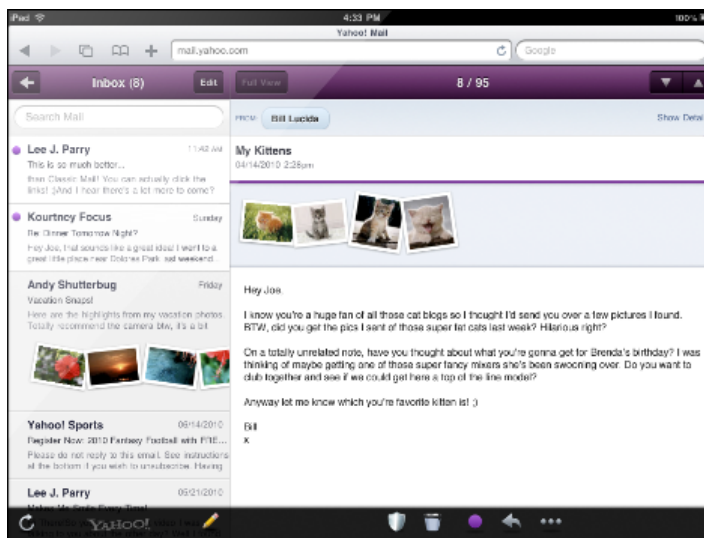


Figura 2.4: Aplicação web para iPad (yahoo)

Uma aplicação web, como o próprio nome indica, está alojada num qualquer servidor é acessível a partir da internet ou de uma rede interna a que o dispositivo esteja ligado. Estas aplicações apresentam como principal vantagem o facto de poderem ser executadas em qualquer sistema operativo ou dispositivo, desde que este possua um *web browser* e acesso à rede. Por esse motivo o desenvolvimento destas aplicações torna-se muito mais rápido, mas também devido à maior difusão das tecnologias (*HTML*, *CSS* e *Javascript* entre outras) e a existência de um numero de programadores com experiência nas mesmas, muito mais vasto do que em qualquer tecnologia nativa de um qualquer sistema operativo móvel. Embora o senso comum leve a crer que uma aplicação web móvel, por não estar alojada no dispositivo, não consiga aceder a capacidades nativas do mesmo, os recentes desenvolvimentos das tecnologias web (em especial o *HTML5*), contrariam essa conceção. Uma aplicação web consegue aceder à maioria das capacidades nativas dos dispositivos, utilizando uma *Browser Application Programming Language*, vulgarmente designada por *Browser API*.

Em geral, estas aplicações perdem em termos de eficiência em relação às aplicações nativas, muito embora esta barreira comece a ser quebrada devido à evolução das redes nomeadamente as redes *3G* e a recém chegada *4G*. Existe também, um número crescente de *frameworks* que permitem às aplicações web simular uma aparência nativa, como é o caso por exemplo do *jQuery Mobile*.

De uma forma geral é possível concluir que embora as aplicações nativas possuam vantagens claras e um domínio acentuado do mercado de aplicações móveis, esta é uma tendência que está a começar a mudar. As aplicações web móveis começam cada vez a aproximar-se mais dos dispositivos e a ganhar terreno em termos de popularidade, sendo que caminhamos provavelmente para uma situação de equilíbrio entre ambas. [mob12b]

2.3.2 Acesso a dados

Uma vez que a informação mais importante para a aplicação, os resultados de *MCDTs* dos doentes, está contida num servidor, é necessário aceder a essa informação, e esse acesso é feito através de *Web Services*.

Um *Web Service* é uma sistema feito para suportar a interoperabilidade entre duas máquinas, numa determinada rede. Esta interoperabilidade é feita entre um fornecedor de um serviço e um requisitante desse mesmo serviço. O fornecedor do serviço é uma entidade (pessoa ou uma empresa), que por motivos comerciais ou outros, disponibiliza um serviço através de um agente fornecedor. O requisitante do serviço é também uma entidade (pessoa ou organização) que necessita de uma ou mais funcionalidades fornecidas pelo serviço e comunica com o agente fornecedor através de um agente requisitante.

O “contrato” feito entre fornecedor e requisitante é constituído por duas partes - a descrição do serviço e a semântica. A descrição do serviço não é mais do que a definição da mecânica das mensagens trocadas, como o formato das mesmas, os tipos de dados, os protocolos de transporte e os formatos de serialização do transporte, documentada numa *Web service description* - *WSD*. Esta *WSD* é especificada na *Web service description language* - *WSDL* que é compreensível pelas máquinas que implementam os agentes fornecedor e requisitante. A semântica é um acordo entre as entidades sobre o intuito das mensagens trocadas, e o que é espectável que o serviço responda a determinadas mensagens, ou seja, a forma como os agentes devem interagir. Não tem que ser necessariamente escrito ou negociado, podendo simplesmente ser anunciado pelo fornecedor de serviço, de forma implícita ou explícita, oral ou escrita, compreensível pelas máquinas que implementam o serviço ou não. [Con04]

Para que estes *Web services* possam ser utilizados são necessários quatro passos distintos, ilustrados na figura 2.5

1. O fornecedor e o requisitante do serviço tomam conhecimento um do outro (ou apenas o requisitante);
2. As entidades acordam a semântica e o *WSD* (ou o requisitante simplesmente toma conhecimento dos mesmos);
3. As entidades implementam a semântica e o *WSD* nos agentes fornecedor e requisitante respetivamente;

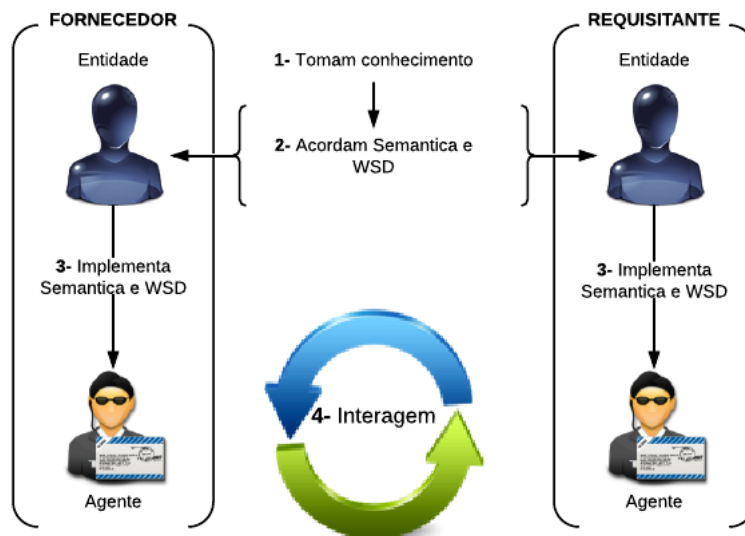


Figura 2.5: Estabelecimento da interação entre fornecedor e requisitante de um serviço

4. Os agentes interagem entre si.

SOAP SOAP é um protocolo para a troca de informação estruturada, através de mensagens *XML*. Está projetado para ser utilizado num ambiente descentralizado e distribuído. O tipo de mensagens utilizado pode ser enviado utilizando um de vários protocolos de rede nomeadamente *HTTP* (que é o mais comum), mas também *SMTP*, *FTP*, *RMI/IIOP* ou até um protocolo definido por terceiros. A sua *framework* foi feita de forma a ser completamente independente do modelo e semântica da implementação utilizados.

Os principais objetivos passam pela sua simplicidade e extensibilidade que são atingidos essencialmente ocultando as especificidades da *framework* de troca de mensagens como a segurança, o roteamento, a confiabilidade e os padrões de troca de mensagens. [Con07a]

Outro dos objetivos passa pela definição de mensagens que possibilitem a invocação de processos remotos, em linguagens de programação comuns (Remote Process Call - RPC). [Con07b]

REST REST, acrónimo de *Representational State Transfer* define um conjunto de restrições/orientações de arquitetura de software, destinado ao desenvolvimento de sistemas distribuídos dependentes de recursos, num padrão cliente-servidor. Como o próprio autor do conceito o define:

REST is a coordinated set of architectural constraints that attempts to minimize latency and network communication, while at the same time maximizing the independence and scalability of component implementations. [Fie02]

A própria *World Wide Web* está implementada segundo uma arquitetura REST. Para compreender este conceito vejamos o que é necessário para que um *Web service* seja compatível com o mesmo, ou seja, *RESTful*:

- Utilizar métodos *HTTP* explicitamente, de uma forma consistente com a definição do protocolo. Assim é mantida a uniformidade das operações possíveis, sem correr o risco de existirem operações acidentalmente indesejáveis. Assim o *REST* estabelece uma relação directa entre as operações *CRUD* e os métodos *HTTP* da seguinte forma:
 - *Create* com *POST*;
 - *Read* com *GET*;
 - *Update* com *PUT*;
 - *Delete* com *DELETE*;
- Ser independente do estado. Um cliente, quando envia um pedido, deve incluir todos os dados necessário para que o servidor execute o pedido. Desta forma o servidor não precisa de saber qual o estado ou contexto em que cada cliente está. Isto faz com que a implementação do serviço no servidor seja muito mais simples e com que seja escalável, ou seja, um servidor pode recorrer a outros para desempenhar mais rapidamente uma tarefa, aumentando a performance.
- Utilizar *URIs* estruturados hierarquicamente. Desta forma o acesso a um determinado recurso torna-se intuitivo e organizado. Os *URIs* também devem ser estáticos para que caso o recurso ou a implementação do serviço mudem, o caminho seja o mesmo. Ao mesmo tempo é importante que a forma como os recursos estão relacionados na codificação do *URI* não reflita a forma como estão armazenados, para que apenas o serviço saiba como aceder aos mesmos.
- Transferir os estados em *XML* ou *JSON* uma vez que são formatos uniformizados e simples o que permite que o mesmo serviço possa ser utilizado por vários clientes, em diferentes linguagens, em diferentes plataformas ou dispositivos. [Rod08]

Embora o *REST* seja visto como o substituto do *SOAP*, ambas as formas de implementação de *Web services* são largamente utilizadas e possuem vantagens e desvantagens em determinados cenários.

2.3.2.1 WCF

O *WCF - Windows Communication Foundation* é uma *framework* desenvolvida pela *Microsoft*, como parte integrante da *.NET framework 4.5*, que permite a criação e implementação de serviços distribuídos, ou seja, *Web Services*. Esta solução surgiu como um agregador das funcionalidades das várias tecnologias anteriores a si, nomeadamente *ASMX*, *.NET Remouting*, *Enterprise Services*, *WSE*, *System. Messaging* e *System. Net*. [DCA07]

A informação é enviada de forma assíncrona de um *endpoint* de serviço para um ou vários *endpoints* clientes. O *endpoint* de serviço pode ser um servidor de qualquer tipo não sendo necessário ser *Microsoft*.

As principais características e funcionalidades do WCF, com especial interesse no âmbito desta dissertação são as seguintes:

- **Orientado a serviços**, permitindo facilmente a extensibilidade da aplicação para a inclusão de novos tipos de clientes;
- **Interoperabilidade** com outros tipos de *Web services*;
- **Vários padrões de mensagem**, desde pedido/resposta a sentido único entre outros, possibilitando a escolha do mais adequado;
- **Segurança**, através da possibilidade de encriptação das mensagens trocadas;
- **Durabilidade das mensagens**, assegurada guardando a mensagem em base de dados enquanto esta não é enviada (para prevenir perda de mensagens no caso de existirem problemas de comunicação);
- **Vários protocolos e transporte**, sendo o *SOAP* através de *HTTP* a forma privilegiada. É possível também que as mensagens sejam enviadas apenas em *XML* seguindo a filosofia REST; [Mic12]

Um serviço *WCF* é composto por três elementos distintos, que são a classe do serviço, geralmente implementada em *C#* ou *Visual Basic* com um ou mais métodos, o servidor ou *host* que aloja o processo e um ou mais *endpoints*, que são pontos de acesso ao serviço, através dos quais o serviço é acedido pelos clientes.

Os pontos de acesso são constituídos por um endereço que representa o local onde o serviço pode ser acedido (*URL*), uma configuração que define que protocolo (ou combinação destes) deve ser utilizado para ser acedido e um contrato que especifica o tipo de mensagem que é utilizado na comunicação. [DCA07]

2.3.2.2 Java EE 6

O *Java Enterprise Edition 6* é uma plataforma de desenvolvimento de aplicações empresariais, que utiliza o servidor *GlassFish Open Source edition*. Esta plataforma foi desenvolvida através do *Java Community Process (JCP)*, que é responsável pelo desenvolvimento de todas as tecnologias *Java*.

As aplicações desenvolvidas com auxílio desta plataforma, são aplicações multi-camada, possuindo por norma três camadas, dispersas por três localizações distintas: a máquina do cliente, a máquina do servidor *Java EE* e a base de dados. Desta forma estas aplicações estendem a versão *standard* das aplicações cliente-servidor que possuem apenas duas camadas, colocando o servidor *Java EE* entre o cliente e o servidor da base de dados.

A arquitetura das aplicações *Java EE* é constituída por **componentes**. Um componente é uma unidade funcional e independente que é agregada na aplicação *Java EE*, que comunica com outros componentes. Existem vários tipos de componentes nomeadamente os **componentes de**

cliente como os clientes aplicacionais e as *applets*, os **componentes web** como os *java servlets*, entre outros e os **componentes de servidor** como é o caso dos *Enterprise Java Beans (EJB)*. Os componentes são desenvolvidos na linguagem de programação *Java*, sendo compilados da mesma forma que qualquer classe *Java*, tendo apenas que respeitar as normas de formação de componentes *Java EE*, uma vez que são agregados, executados e geridos pelo servidor *Java EE*. Os **contentores** são a interface entre os componentes e as funcionalidades de baixo nível específicas e cada plataforma que os suporta. Para que possam ser executados, os componentes devem ser compilados e implementados no seu respetivo contentor.

As aplicações cliente podem ser desenvolvidas na linguagem *Java*, com ajuda de *APIs* próprias para o efeito, sendo obviamente esta a forma privilegiada para tal. No entanto as aplicações podem ser desenvolvidas noutra qualquer linguagem possuindo da mesma forma acesso ao servidor *Java EE*, o intermediário com o servidor da base de dados, o que faz do *Java EE* uma plataforma interoperável. A sua arquitetura de componentes, e independência de plataforma torna as aplicações *Java EE* fáceis de desenvolver, uma vez que a lógica de negócio está organizada em componentes que podem ser reutilizados. As aplicações *Java EE* seguem um modelo de programação simplificado, substituindo a utilização de descritores *XML* por anotações que podem ser inseridas no próprio ficheiro *Java*. Com as anotações é possível colocar a especificação da informação diretamente no código, ao lado do elemento que é diretamente afetado pela mesma. Existem também mecanismos pré definidos de controlo de acesso, como por exemplo o login de utilizadores, poupando assim aos programadores o desenvolvimento deste tipo de funcionalidades básicas.

Para a implementação de serviços através do protocolo **SOAP**, a plataforma *Java EE* disponibiliza a funcionalidade designada de *JAX-WS*, tipicamente utilizado para a implementação de “grandes” *webservices*.

Para a implementação de serviços através do protocolo **REST** a plataforma *Java EE* disponibiliza a funcionalidade *JAX-RS*, utilizada preferencialmente em ambientes de integração. [Cor11]

Notas finais

É ainda importante referir que para além das duas tecnologias referidas (*WCF* e *Java EE*) cada plataforma móvel possui os seus mecanismos próprios para a o acesso a *webservices*.

Ressalva-se ainda o facto de que em qualquer destes cenários, a utilização do protocolo **REST** ou **SOAP**, depende bastante do tipo de problema a resolver. Nenhum dos protocolos é consistentemente superior ao outro em todas as situações. No âmbito da aplicação a desenvolver, o tipo de protocolo/tecnologia a utilizar dependerá essencialmente da forma como o webservice está já implementado, evitando assim ao máximo a sua alteração, permitindo a sua utilização sob a forma de “caixa-negra”.

2.3.3 Ambientes de desenvolvimento multi-plataforma

O crescimento das tecnologias móveis, nomeadamente o aumento da procura no mercado dos *smartphones* levou ao aparecimento de novas plataformas, e ao constante desenvolvimento das mesmas, para poderem acompanhar o ritmo da inovação. Com a proliferação das plataformas, a oferta para os consumidores torna-se cada vez mais competitiva e atraente mas dificulta a vida a quem desenvolve aplicações para estas plataformas. O desafio de disponibilizar uma aplicação para todas as plataformas é praticamente inexequível em tempo útil se for necessário replicar completamente o código nas diferentes tecnologias de desenvolvimento para cada plataforma. Se a isto juntarmos a variedade de formatos dos dispositivos (tamanho de ecrã, formas de entrada de dados, etc) e a velocidade com que as plataformas são atualizadas, confirmamos que esta é uma tarefa muito penosa e extremamente difícil de concretizar.

Para solucionar este problema, nos últimos anos têm proliferado e evoluído ambientes de desenvolvimento multi-plataforma que seguem a filosofia de codificar numa única tecnologia e executar nas diferentes plataformas.

Existem várias formas de alcançar o objetivo de uma aplicação ser capaz de executar em diferentes plataformas, que são as seguintes:

- **Cross compilation**- Esta técnica separa completamente o ambiente de desenvolvimento, do ambiente de execução. O desenvolvimento processa-se da seguinte forma: a plataforma fornece uma *API* independente de qualquer plataforma utilizando uma linguagem de programação bem difundida (como *Javascript* ou *Ruby*). Esta *API* é utilizada para desenvolver a aplicação nomeadamente a interface com o utilizador, a persistência de dados e a lógica de negócio. O código é processado por um compilador multi-plataforma que transforma o código em aplicações nativas de cada plataforma. A principal vantagem desta técnica é que é capaz de gerar aplicações nativas que se comportam de uma forma extremamente semelhante às desenvolvidas exclusivamente para uma plataforma, permitindo uma execução eficiente e uma experiência de utilização consistente. A desvantagem desta técnica é que estes compiladores são extremamente complexos e podem facilmente ficar obsoletos se não forem atualizados regularmente;
- **Máquina Virtual**- Esta técnica é uma variação da anterior. A codificação é feita da mesma forma que na técnica anterior, mas é fornecida também uma máquina virtual que é utilizada para executar as aplicações abstraindo a plataforma alvo do código que está a ser executado. Esta técnica partilha a maioria das vantagens da anterior, uma vez que gera um ambiente nativo, mas as aplicações tendem a ser executadas de uma forma ligeiramente mais lenta devido à latência introduzida pela execução da máquina virtual na tradução de instruções para a plataforma subjacente;
- **Aplicações web**- Outra abordagem com uma popularidade crescente passa pelo desenvolvimento de aplicações *web* que podem ser executadas no browser do dispositivo ou embutidas numa aplicação. Esta técnica utiliza tecnologias *web* como *HTML*, *CSS* e *Javascript* e tenta

simular um ambiente nativo. São utilizadas as capacidades inerentes ao *HTML5* e *CSS3* para permitir o armazenamento de dados, animações, reprodução de áudio e vídeo, encurtando as distâncias entre a aplicação e o dispositivo. Esta técnica pode ser extremamente eficiente para aplicações com uma forte dependência da Internet e pouca necessidade de processamento como clientes email, leitores de *feeds* ou notícias e interação social mas não pode ser encarada como solução para aplicações com grande necessidade de processamento como por exemplo jogos. A principal vantagem passa pela facilidade de distribuição destas aplicações e pela fácil adaptação às diferentes plataformas (uma vez que todas elas possuem um *browser*), no entanto, na maioria dos casos é bastante difícil gerar um ambiente nativo nos dispositivos, estando apenas disponíveis elementos visuais *web* e não os específicos de cada plataforma;

- **Widgets**- Esta solução já existia para ambientes *desktop* mas foi adaptada para dispositivos móveis. Um *widget* é uma ferramenta interativa normalmente com apenas uma funcionalidade simples como uma calculadora, um calendário ou por exemplo mostrar as últimas notícias ou o estado do tempo atual. As *widgets* são executadas por um motor específico que lhes permite o acesso a funcionalidades básicas dos dispositivos. O desenvolvimento é feito utilizando tecnologias *web* como *HTML*, *CSS* e *Javascript*. Esta solução serve apenas para aplicações simples e não sendo uma alternativa viável para aplicações mais elaboradas; [HSD11]

Este problema toma uma importância vital nesta dissertação, uma vez que é necessário desenvolver uma aplicação capaz de executar nas plataformas *Android* e *iOS*. Assim, serão agora analisadas as soluções mais robustas existentes no mercado, por forma a escolher a que mais se adequa ao problema em questão.

2.3.3.1 Appcelerator Titanium

O *Titanium* é um ambiente de desenvolvimento multi-plataforma desenvolvido pela *Appcelerator Inc.* tendo duas versões: *Titanium Mobile* e *Titanium Desktop*. O último permite o desenvolvimento em simultâneo para *Windows*, *Mac OS* e *Linux* com uma percentagem de reutilização de código a rondar os 100 % [Inc11c]. A solução que irá ser agora analisada em detalhe será o *Titanium Mobile* que é a que se adequa à necessidade desta dissertação.

O *Titanium Mobile* permite o desenvolvimento em simultâneo para *iPhone*, *iPad* e dispositivos *Android*. Tem ainda em fase *beta* a possibilidade de desenvolver para *blackberry*.

Com o *Titanium* é possível aceder a diversas capacidades nativas dos dispositivos:

- Vibração;
- GPS e mapas;
- Acelerómetro;
- Reprodução de multimédia;

- Orientação do dispositivo;
- Galeria de fotografias (para visualização ou armazenamento);
- Câmara com captura de imagens e gravação de vídeo;
- Detectar que o utilizador agitou o dispositivo;
- Captura de ecrã;
- Alarmes de proximidade (pontos de interesse);
- *Push Notifications*;

Para além destas funcionalidades que podem ser utilizadas de uma forma independente ao dispositivo, o *Titanium* tem ainda formas integradas de fornecer acesso a redes sociais (*Facebook* e *Twitter*), *RSS* e *APIs SOAP*. [AGL10]

Este ambiente de desenvolvimento é baseado num conjunto de *APIs Javascript* independentes das plataformas. Utiliza a técnica de *cross compilation*, estando a sua arquitetura organizada da seguinte forma: Numa primeira camada temos o código da aplicação, desenvolvido, como supra-referido, utilizando *APIs Javascript*. É possível desenvolver a interface da aplicação utilizando *HTML* e *CSS* mas a melhor forma, para obter os elementos gráficos nativos é utilizar a mesma *API Javascript*. Imediatamente abaixo encontra-se uma camada onde podem ser desenvolvidos módulos nativos de cada plataforma que eventualmente sejam necessários. A parte mais importante da arquitetura é a *Bridge*. Esta camada é um motor *Javascript* responsável por “traduzir” o código das *APIs Javascript* em código nativo de cada plataforma. [Inc11a] A figura 2.6 ilustra esta arquitetura.



Figura 2.6: Arquitetura do ambiente de desenvolvimento multi-plataforma Titanium

O *Titanium Mobile SDK* trabalha em conjunto com os *SDKs* das plataformas para transformar o código das *API Javascript* em código nativo de cada plataforma com uma eficiência próxima das aplicações nativas puras, que pode ser testado num emulador de cada dispositivo ou no dispositivo em si. Para que isto seja possível é necessário que os *SDKs* de cada plataforma estejam devidamente instalados, o que implica que se se desejar fazer desenvolvimento em simultâneo para

Android e *iOS*, este seja feito num computador com *Mac OS X*, uma vez que o *iOS SDK* apenas está disponível para esta plataforma. No caso de ser necessário testar a aplicação em dispositivos *Apple* é necessário também ter a licença paga, como referido na secção dedicada à referida plataforma.

Embora seja possível utilizar o *Titanium Mobile SDK* com qualquer *IDE* e emulador de dispositivo à escolha, existe uma solução integrada desenvolvida pela *Appcelerator Inc.* que é o *Titanium Studio*. É um *IDE* completo baseado em *eclipse* que permite gerir todas as tarefas relacionadas com um projeto multi-plataforma *Titanium*, desde edição de código em *HTML5*, *CSS3*, *JavaScript*, *Ruby*, *Rails*, *PHP* e *Python* com *code completion*, à depuração do mesmo, teste em simuladores dos dispositivos e preparação dos pacotes de instalação da aplicação. Possui ainda integração com o repositório de código *Git*. [Inc11d]

Para utilizar o serviço de *Push Notifications* é necessário utilizar um serviço intermediário designado de *Urban Airship*. Este serviço está encarregue de lidar com os diferentes serviços de notificações das diferentes plataformas. [Inc11b]

Protótipos Foram desenvolvidos vários protótipos nesta tecnologia para averiguar a sua capacidade essencialmente para a reprodução de multimédia e acesso a webservices. Em relação à reprodução multimédia concluiu-se que esta plataforma consegue reproduzir bem a maioria dos formatos necessários, embora a manipulação de imagens (zoom e rotação) seja mais complexa, devido ao facto de alguns eventos *multitouch* não existirem nesta *framework* (como é caso do *rotate*). Por outro lado a reprodução e manipulação de *pdfs* e *docs* é feita com recurso a uma *webview*, que apesar de ser uma solução satisfatória, não é a preferível. A figura 2.7 mostra alguns testes realizados a este protótipo. Em relação ao acesso a webservices, foram tentadas várias alternativas. As tentativas iniciais vizavam a utilização de ferramentas como o *sudzc*, *wsdl2js* e *suds.js* para a geração do código javascript necessário à chamada do *webservice*, através do *wsdl* do mesmo. Estas ferramentas, provaram-se ineficientes, essencialmente porque embora funcionem bem para *webservices* simples, para um nível superior de complexidade quando os argumentos das chamadas, são objectos com vários atributos, o funcionamento fica comprometido. Assim a solução encontrada passava pela construção do envelope *SOAP* para cada pedido, bem como a decodificação da resposta recebida. Para piorar ainda a situação, para lidar com o problema do *crossreference* entre domínios, era necessária a criação de um serviço intermédio que fizesse a ponte entre o *eResults* e a aplicação, transformando as respostas em formato *JSON* para uma mais fácil manipulação da mesma ao nível da aplicação. Esta solução implicaria não só o desenvolvimento de uma interface para cada uma das funções necessárias à aplicação, neste serviço intermédio, como também a criação de uma camada de acesso a *webservices* na aplicação que lidasse com construção dos envelopes *SOAP* e decodificação das respostas *JSON*. Por último, foi constatado que em termos de organização do projeto, o *Titanium* não apresenta soluções sólidas, o que se torna preocupante, quando a dimensão dos projetos aumenta.

Pode então ser concluído que o *Titanium* é um ambiente de desenvolvimento relativamente sólido que permite utilizar conhecimentos de tecnologias web para ser rapidamente produtivo no

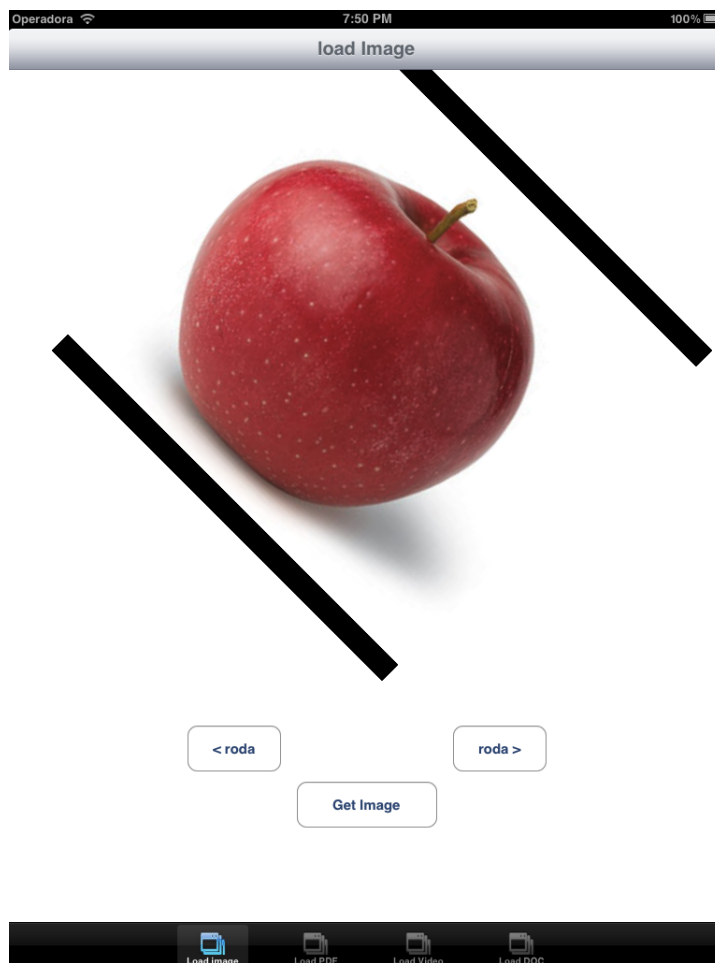


Figura 2.7: Protótipo de suporte a multimédia do titanium

desenvolvimento de aplicações para dispositivos móveis. A sua principal vantagem passa por ser capaz de gerar código nativo para cada plataforma. No entanto a documentação é desorganizada e pouco útil, sendo esta falha ligeiramente colmatada pela extensa e activa comunidade no forum de discussão.

2.3.3.2 MonoTouch/Mono for Android

O *MonoTouch/Mono for Android*, desenvolvido pela *Xamarin* emprega uma filosofia bastante diferente da observada no *Titanium*. O seu principal objetivo é que seja possível desenvolver aplicações para *iOS* e *Android* sem ter necessidade de conhecer *Objective C* e *Java*, utilizando em vez disso *C#* e a *.Net Base Class Library - BCL* que são tecnologias vastamente utilizadas. O desenvolvimento não é feito em simultâneo. O ambiente principal, o *MonoTouch* permite desenvolver aplicações para *iPhone* e *iPad* enquanto o ambiente *Mono for Android* permite desenvolver para dispositivos *Android*. Embora o desenvolvimento seja feito neste ambientes separadamente, a grande maioria do código pode ser reutilizado graças ao facto de ser desenvolvido numa tecnologia que abstrai as tecnologias nativas. [Xam11] Será feita uma análise apenas ao *MonoTouch*, que

é o ambiente original e principal, uma vez que o *Mono for Android* possui a mesma arquitetura mas transposta para a tecnologia *Android*.

O *MonoTouch* fornece um *IDE* específico designado *MonoDevelop*, extremamente semelhante ao popular *IDE Visual Studio* e está disponível para *Windows*, *MAC OS* e *Linux*. Permite edição de código e com a ajuda de funcionalidades como *code highlighting* e *code completion*, gerir o projeto de forma integrada, testar as aplicações em simuladores dos dispositivos, utilizar o *interface builder* para desenvolver as interfaces, entre outras funcionalidades.

A suportar o *MonoTouch* está a *framework Mono* que não é mais do que uma versão *Open Source* da *framework .Net* que pode ser utilizada em qualquer plataforma. É utilizada uma versão ligeiramente adaptada para que possa ser feito o desenvolvimento nesta tecnologia e para que depois o código possa ser “linkado” para a tecnologia nativa de cada dispositivo.

O *MonoTouch* tem quatro componentes fundamentais:

- O *MonoTouch dll*, um *C# assembly* que é uma *API* de ligação às *APIs* nativas do *iOS*;
- O *mtouch* que é um *SDK* de linha de comandos que faz a compilação do *C#* e do código intermédio (*CIL*- Common Intermediate Language). Este código compilado é capaz de correr num simulador de *iOS* ou num dispositivo;
- Um aditivo para o *MonoDevelop* que permite a integração com o *mtouch*, o *Interface Builder* e o *iPhone Simulator*. Também permite posteriormente a expedição da aplicação para dispositivos e a sua distribuição na *App Store*
- Uma licença comercial do *Mono runtime* que permite a ligação do código desenvolvido com o mesmo. [MBD⁺10]

Para além de mais poderoso em comparação com o javascript, o *C#* permite também uma melhor estruturação do código/projeto, uma vez que organiza o código segundo o modelo MVC. No entanto uma desvantagem da abordagem do *MonoTouch* é que as aplicações geradas neste ambiente ocupam mais espaço em disco do que se fossem desenvolvidas em *Objective-C* e isso deve-se ao facto de cada aplicação ser obrigada a ter uma cópia da *framework Mono*.

O que se pode concluir é que este é um ambiente também sólido, que utiliza uma linguagem de programação muito poderosa e bastante difundida, que oferece uma abstração que nos dá a capacidade de gerar resultados mais rapidamente do que utilizando as tecnologias nativas mas que utiliza uma filosofia um pouco diferente da que é procurada no âmbito desta dissertação.

2.3.3.3 PhoneGap

O *PhoneGap* é um ambiente de desenvolvimento multi-plataforma *open source*, desenvolvido na sua grande maioria pela *Nitobi* recentemente adquirida pela *Adobe*. De entre as plataformas analisadas, é a que cobre a maior variedade de plataformas nomeadamente *iOS*, *Android*, *Windows Mobile*, *Blackberry*, *Web OS*, *Symbian* e *Bada*. [Nit11a]

Este ambiente permite desenvolvimento de aplicações nativas das plataformas supra-referidas sendo que a codificação é feita em tecnologias *web* nomeadamente *HTML* (com suporte para *HTML5*), *CSS* (com suporte para *CSS3*) para a interface e *Javascript* para a lógica de negócio, sendo o acesso às funcionalidades dos dispositivos feito através de uma *API* de *Javascript* própria. Através desta *API* é possível aceder a funcionalidades como *GPS*, a lista de contactos, vibração, sensores dos dispositivos (acelerómetro e giroscópio), câmara, reprodução de som, informação do dispositivo e *Click to talk* [AGL10]

A arquitetura seguida é um modelo *web* híbrido possuindo *wrappers* aplicacionais para as plataformas móveis mencionadas. As aplicações correm num *browser* incorporado que renderiza a interface e faz a ligação com as funcionalidades nativas dos dispositivos. Este é o fator que faz com que o *PhoneGap* esteja presente em tantas plataformas e ao mesmo tempo cria a sua maior desvantagem, que é o facto de não ser capaz de gerar de uma forma consistente a interface nativa de cada plataforma.

O *PhoneGap* é indicado para o desenvolvimento de aplicações com uma forte componente *web*, especialmente para a adaptação de aplicações *web* para dispositivos móveis. Uma das suas principais vantagens é que ao adaptar uma aplicação *web* já existente, a maioria do código já desenvolvido pode ser reutilizado para a aplicação móvel. [HSD11] Outra grande vantagem passa pela possibilidade de utilizar um vasto conjunto de *frameworks* largamente difundidas disponibilizadas pela comunidade como por exemplo o *jQuery mobile*, *XUI*, *Sencha Touch*, *Dojo Mobile*, entre muitas outras, que permitem realizar tarefas desde a construção da interface até à lógica de negócio. [Nit11c]

Para o desenvolvimento de aplicações não existe nenhuma ferramenta específica, bastando somente um editor de texto. No entanto existem *plugins* do *PhoneGap* para *IDEs* como o o *Eclipse* ou o *XCode*. Sendo utilizada esta via e a compilação do código sendo feita localmente na máquina do utilizador, o *PhoneGap* torna-se um pouco limitativo porque apenas permite compilar o código para cada plataforma separadamente, sendo necessário instalar o *SDK* respetivo e respeitar os requisitos do mesmo.

O meio realmente privilegiado para compilar o código deste ambiente é o *PhoneGap Build*. Com esta ferramenta basta fazer *upload* do código uma única vez e a compilação é feita na *cloud*, gerando aplicações prontas para distribuir, nas diferentes plataformas. [Nit11b]

Pode ser concluído que esta plataforma pretende cobrir um tipo de aplicações bem definido e apresenta grandes vantagens para o desenvolvimento das mesmas. Embora o *PhoneGap Build* permita compilar o código para diversas plataformas em simultâneo, não é uma solução muito agradável para testar as aplicações, fazendo falta um *IDE* próprio para este ambiente.

2.3.3.4 Rhodes

O *Rhodes* é uma *framework Open Source* do ambiente de desenvolvimento multi-plataforma desenvolvido pela *Rhomobile Inc.* fazendo parte do grupo *Motorola Solutions*. [Inc11f]

Esta *framework* permite desenvolver aplicações nativas em simultâneo para as plataformas *iOS*, *Android*, *Windows Mobile*, *Symbian*, and *BlackBerry*, dando cobertura às principais plataformas do mercado.

As aplicações são desenvolvidas segundo o padrão de desenvolvimento *MVC* de forma semelhante à *framework Rails* sendo a lógica de negócio desenvolvida em *ruby*, e a interface em *HTML* (ficheiros *erb* que são *templates* de *HTML* com *ruby* embutido), *CSS* e *Javascript*. Uma grande vantagem desta *framework* é a inclusão de uma ferramenta de gestão objeto-relacional, designado *Rhom*. Esta ferramenta permite criar uma abstração da base de dados utilizada por cada plataforma, tendo uma abordagem alto-nível, semelhante ao *Active Record* em *Rails*.

É possível aceder às funcionalidades de cada dispositivo, nomeadamente aos contactos e calendário, *GPS* e mapas, câmara, *push notifications*, reconhecimento de assinaturas e *bluetooth*.

Na figura 2.8 é possível observar a arquitetura do *Rhodes* cujos componentes são:

- **Model View Controller**- a camada superior onde se encontra o código da aplicação, estruturado segundo o padrão *MVC*;
- **Rhom**- esta camada é responsável pela gestão objeto-relacional, ou seja, pela interação com as bases de dados dos dispositivos, e respetiva tradução para objetos que o modelo consiga entender;
- **Máquina Virtual Ruby**- esta máquina virtual executa em todas as plataformas suportadas e é aqui que o código *ruby* é executado;
- **APIs do Dispositivo**- aqui encontram-se as *APIs* que permitem aceder às funcionalidades dos dispositivos;
- **Cliente RhoConnect**- esta camada é responsável por sincronizar a informação proveniente do servidor *RhoConnect*, a ser abordado de seguida; [Nal11]

A arquitetura supra-referida é contemplada apenas para o desenvolvimento de aplicações locais, sem necessidade de sincronização com servidores aplicativos. Para o caso de aplicações mais complexas que precisem constantemente de ser “alimentadas” por dados provenientes de serviços *web*, existe a ferramenta *RhoConnect* (versão 3.0 do *RhoSync*). Esta ferramenta é automaticamente sincronizada com os serviços *web* que desejar (*REST* ou *SOAP*) e posteriormente interage com a camada cliente *RhoConnect* do *Rhodes* utilizando os serviços de comunicação assíncrona com os dispositivos como é o caso do *C2DM* do *Android* por exemplo. Permite também a disponibilização dos dados *offline* para que possam ser consultados mesmo sem ter uma ligação à rede. Caso o programador deseje pode, fazer ele próprio a chamada aos serviços e a sincronização dos dados, mas esta ferramenta permite facilitar e abstrair essa tarefa. Esta ferramenta é ideal para aplicações móveis empresariais que necessitem de ser alimentadas constantemente por dados vindos por exemplo de um *ERP*, no entanto, não é gratuita e necessita de um plano empresarial pago.



Figura 2.8: Arquitetura da *framework Rhodes*

O desenvolvimento pode ser feito de duas formas. O *Rhostudio* é um *plugin* para *eclipse* que permite o desenvolvimento em *ruby* com as funcionalidades naturais do *eclipse*. Esta solução necessita no entanto de que sejam instalados na máquina os *SDKs* respetivos de cada plataforma, para as quais se deseja desenvolver, e respeitar os seus respetivos requisitos.

A segunda forma, que é privilegiada pela *Rhomobile* e considerado uma das principais vantagens deste ambiente é o *RhoHub*. É uma aplicação web que permite a edição de código online, integração com um repositório *git*, uma instalação do servidor *RhoSync* e a compilação de código para todos os dispositivos suportados sem ter a necessidade nenhum *SDK*.

Recentemente passou a estar disponível também a ferramenta *RhoGallery*, que permite a distribuição e gestão de aplicações num ambiente empresarial. [Inc11e] [Nal11]

Podemos concluir que esta é uma solução bastante completa que permite uma abordagem diferente, mais estruturada que as restantes. Permite a programadores *web*, especialmente se tiverem conhecimento de *Ruby on Rails*, começar a desenvolver aplicações em muito pouco tempo.

2.3.3.5 Sencha Touch

O *Sencha Touch* é uma *framework* de desenvolvimento de aplicações móveis web, desenvolvida em *HTML5*, pela empresa *Sencha Touch Inc.*. A sua versão atual é o *Sencha Touch 2.0*, dando suporte a vários sistemas operativos nomeadamente *iOS*, *Android*, *BlackBerry* e *Kindle OS*. Não possui nenhum ambiente de desenvolvimento integrado, deixando ao critério de cada um a escolha do ambiente que mais lhe agrada, podendo ser desde o *Visual Studio* até ao *notepad* ou *Vim*.

Apesar de ser essencialmente talhado para o desenvolvimento de aplicações móveis web, vem com uma ferramenta, designada de *sencha touch tools*, que permite não só a criação automática da maioria dos ficheiros necessários a um novo projeto, como também o “empacotamento” de uma aplicação web para uma aplicação nativa de *iOS* ou *Android*. Esta característica, bem como a sua

elevada performance em termos de renderização, animações e *scrolling*, encurtam a fronteira entre aplicações web e nativas.

O desenvolvimento assenta sobre componentes da sua *API Javascript*, que derivam do sistema de classes *Ext Js 4*, disponibilizando mais de cinquenta componentes diferentes. Para além das configurações que podem ser dadas a cada componente é ainda possível alterar os mesmos através de CSS, ou até criar páginas ou elementos em *HTML e CSS*. O desenvolvimento segue um padrão *MVC*, que torna a sua organização muito mais simples, mesmo com o aumento da dimensão do projeto. [Inc12b]

Protótipos À semelhança do *Appcelerator Titanium*, foram desenvolvidos protótipos nesta tecnologia, procurando testar a sua capacidade para a reprodução e manipulação de multimédia e acesso a webservices. No campo da reprodução multimédia foi relativamente simples perceber que o sencha suporta abertura de vários formatos de imagem, no entanto a sua manipulação é relativamente mais complexa, tendo sido necessário recorrer a um *plugin* externo para desempenhar essa tarefa. A reprodução de vídeo é feita de forma simples, fazendo streaming de um vídeo, após a recepção do seu *url*. Em relação à reprodução de *pdfs* e *docs* a primeira solução encontrada foi a criação de uma *iframe*, que renderizava estes documentos recorrendo ao serviço *google docs*. Esta solução foi cedo abandonada, devido ao facto de depender da internet (a aplicação pode ser executado num ambiente em que tem apenas disponível uma rede privada - *intranet*), depender também de um serviço de terceiros e devido ao facto das *iframes* terem um desempenho irregular nas plataformas móveis. Foram exploradas várias alternativas, nomeadamente o *flexpaper*, o *Vuzit*, o *pdfjs*, o *jquery Document Viewer* e *Ux.Media Component* (todas testadas *in loco*), tendo sido chegada a uma solução que não envolve nenhuma das alternativas exploradas (embora utilize algumas ideias observadas no *plugin adaptive UI* do *flexpaper*), desenvolvida em *HTML e Javascript/jQuery* que será explorada em maior detalhe na secção 3.4.3.1. Em relação ao acesso a webservices, a solução encontrada foi semelhante à descrita na secção 2.3.3.1, envolvendo uma chamada *Ajax* ao serviço intermédio, supra-referido. Esta solução envolveria da mesma forma um esforço adicional para o desenvolvimento da interface para cada chamada, bem como a preparação do envelope para cada chamada e o processamento da resposta em formato *JSON*. Foi realizado ainda um protótipo final para testar como seria a interface da aplicação final se esta fosse desenvolvida nesta tecnologia, que é apresentada na figura 2.9.

Em geral é possível concluir que esta tecnologia tem uma curva de aprendizagem bastante curta e possibilita um rápido desenvolvimento de aplicações, auxiliadas por uma documentação extremamente bem organizada e uma comunidade bastante activa no forum. [Inc12a] A sua organização segundo o padrão *MVC* é uma enorme vantagem e ajuda a manter a coerência e escalabilidade do projeto. No entanto uma das principais conclusões é que esta tecnologia se comporta bastante bem para projetos de dimensão pequena/média, sem necessidade de realizar tarefas muito complexas, mas perde alguma robustez, quando a complexidade aumenta, nomeadamente no caso dos webservices.

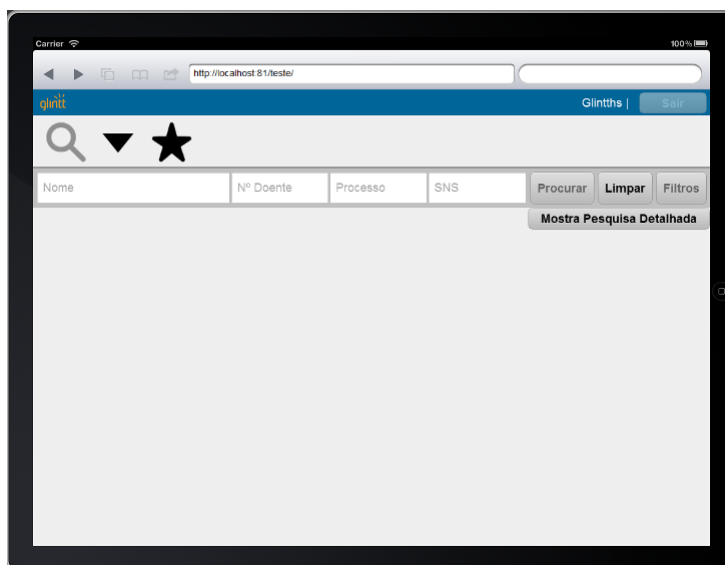


Figura 2.9: Protótipo da aplicação final desenvolvido em Sencha Touch 2.0

2.3.3.6 jQuery Mobile

O *jQuery Mobile* é uma *framework* de desenvolvimento de aplicações móveis web, desenvolvida em *HTML5*, pela *The jQuery Foundation*. Suporta um vasto conjunto de sistemas operativos, nomeadamente *iOS*, *Android*, *Blackberry*, *bada*, *WindowsPhone*, *Palm Web OS*, *Symbian* e *Meego*.

Esta *framework* é uma extensão da popular *framework jQuery*, possuindo por isso uma curva de aprendizagem extremamente rápida para programadores *web* com experiência na mesma. O desenvolvimento de uma aplicação com recurso a esta tecnologia é extremamente semelhante ao desenvolvimento de uma aplicação web normal utilizando *jQuery*, tendo apenas a diferença de ser necessário criar um *viewport*, ao qual serão adicionados os elementos com aspecto nativo (ou antes *mobile*), que são representados como elementos *HTML* normais, sendo identificados por um atributo designado de *data-role*, que identifica o seu papel na aplicação (botão, lista, link, etc).

A sua principal vantagem é que esta tecnologia pode ser integrada facilmente em qualquer aplicação desenvolvida com tecnologias web uma vez que é apenas uma extensão da biblioteca *jQuery*, que não é mais do que código *Javascript*. Este facto tem também um importante peso na quantidade de plataformas em que estas aplicações podem ser executadas. Para que possa ser executado como uma aplicação nativa, é necessário que seja utilizado em conjunto com outra ferramenta, como seja o caso do Phonegap, estando assim restrito às plataformas que este suporta.

Protótipo Foi realizado apenas um protótipo com esta tecnologia, que serviu para ter alguma ambientação e experiência com a tecnologia, e conhecer um pouco da sua organização e componentes. Não foi testada a reprodução de conteúdos multimédia, uma vez que esta tecnologia suporta qualquer conteúdo que seja suportado por *HTML5*, o que inclui imagens, vídeo, áudio e *pdfs* (desde que o plugin do *acrobat reader* esteja instalado no browser) estando apenas dependente do suporte desses conteúdos/formatos, pelos dispositivos em que a aplicação for executada.

Podemos concluir que esta é uma alternativa muito mais leve do que qualquer outra analisada até aqui. Apenas com a inclusão de dois ficheiros está pronta a utilizar não sendo sequer estritamente necessário o seu download (pode ser utilizada a versão *CDN Hosted*). No entanto esta solução praticamente só possibilita a criação de aplicações móveis web, negligenciando a parte nativa.

2.3.3.7 Comparação entre ambientes

Para que se possa perceber as diferenças entre os ambientes de desenvolvimento multi-plataforma, estes são postos lado a lado, sendo comparadas as principais características, na figura 2.10

	 Titanium	 Rhodes	 PhoneGap	 MonoTouch*	 SenchaTouch	 jQuery Mobile
PLATAFORMAS						
Android	Sim	Sim	Sim	Sim	Sim	Sim
iOS	Sim	Sim	Sim	Sim	Sim	Sim
Blackberry	Sim	Sim	Sim	Não	Sim	Sim
Windows Mobile	Não	Sim	Sim	Não	Não	Sim
Outras	Não	Sim	Sim	Não	Sim	Sim
ESPECIFICIDADES						
Linguagem de Programação	API Javascript, HTML, CSS	Ruby, HTML, CSS	HTML, CSS, Javascript	.NET Framework, C#	HTML, CSS, Javascript	HTML, CSS, Javascript(jQuery)
Padrão Desenvolvimento	-	MVC	-	-	MVC	-
Modo de Compilação	Cross Compilation	Máquina Virtual	Modelo Web Híbrido	"Linkagem" para a API Nativa	Web	Web
Elementos UI Nativos	Sim	Sim	Não	Sim	Simulação	Simulação
Suporte para Tablets	Sim	Sim	Sim	Sim	Sim	Sim

Figura 2.10: Comparação entre ambientes multi-plataforma

A primeira coisa que pode ser observada é que todos os ambientes possibilitam o desenvolvimento para as duas plataformas necessárias *Android* e *iOS* (embora o *MonoTouch* seja com recurso ao *Mono for Android*). Neste campo o *Rhodes*, o *Phonegap*, e o *jQuery Mobile* destacam-se como sendo os ambientes que suportam mais plataformas, o que no âmbito desta dissertação pode parecer irrelevante no entanto pode ser importante para trabalhos futuros, e suporte a mais plataformas que não o *Android* e *iOS*.

Em termos de linguagens de desenvolvimento, todas as plataformas giram à volta de tecnologias web, sendo o *MonoTouch* quem mais se destaca permitindo o uso de *C#* e da *.NET Framework* (a *framework Mono* que é uma versão *open source* da mesma). No campo do desenvolvimento

o *Rhodes* e o *Sencha Touch* destacam-se das outras, sendo as únicas que restringem o desenvolvimento segundo o padrão *MVC*. Esta restrição é encarada como um fator positivo, uma vez que garante que são seguidas boas práticas de programação.

Comparando o modo de compilação de cada uma, quem sai em vantagem é o *Titanium*, que utiliza a técnica de *cross compilation*. Desta forma permite gerar aplicações com um desempenho extremamente semelhante a aplicações nativas. O *Rhodes* não fica muito atrás uma vez que a técnica da Máquina Virtual consegue atingir desempenhos semelhantes. Já o *Phonegap* tem a pior abordagem uma vez que executa as suas aplicações sobre um motor de *browser* o que prejudica o desempenho das mesmas. Já o *MonoTouch* faz uma “linkagem” para as *APIs* nativas mas as suas aplicações ocupam bastante mais espaço em disco devido à grande extensão da sua *framework Mono* da qual cada aplicação deve guardar uma cópia. O *Sencha Touch* e o *jQuery Mobile* são considerados à parte uma vez que as suas aplicação são aplicações web e comportam-se como tal.

O *Phonegap*, *Sencha Touch* e *jQuery mobile* ficam em ligeira desvantagem por não suportarem elementos *UI* nativos. No entanto estas plataformas possuem a sua própria *UI*, que simula um ambiente mobile e não perde em termos de usabilidade.

Todas as plataformas suportam o desenvolvimento para *Tablets*, nos sistemas operativos móveis que suportam, por isso esta característica não é distintora.

O que pode ser concluído é que as melhores alternativas analisadas são o *Titanium* e o *Rhodes* em termos de aplicações móveis nativas e o *Sencha Touch* e o *jQuery Mobile* em termos de aplicações móveis web, praticamente em pé de igualdade. O *MonoTouch* poderia ser uma excelente alternativa uma vez que a *Glintt* é bastante adepta de tecnologias microsoft como é o caso do *C#* e da *.NET Framework*, mas a sua filosofia de reutilização de código, em deterioramento de codificar uma vez e executar nas diferentes plataformas faz com que seja preterida.

A decisão sobre a tecnologia escolhida para o desenvolvimento desta dissertação foi complexa e demorada, devido à enorme variedade da oferta e ao número de requisitos que estas tinham que cumprir. A explicação sobre a decisão final será apresentada na secção 3.4.1, quando for abordado o tema da implementação.

2.3.3.8 HTML5

Esta tecnologia foi deixada para último lugar e analisada à parte, não por ser menos importante, mas porque na realidade, por si só não se adequa ao problema da dissertação (mesmo que seja desenvolvida uma aplicação web, esta deverá ter um aspecto mobile e não o de uma aplicação web normal). É um tema extremamente atual, que não podia deixar de ser analisado.

O *HTML* tem acompanhado a *Internet* praticamente desde a sua génese, tendo a sua primeira versão sido publicada em 1993. Esta tecnologia foi criada com a web 1.0 em mente e era simplesmente uma linguagem de *markup*. Foi evoluindo ao longo dos anos, tendo a sua quarta versão sido publicada em 1999. Desde essa altura a *Internet* sofreu uma evolução tremenda e o *HTML* não foi capaz de se adaptar de uma forma consistente à *web 2.0*.

Com o objetivo de levar esta tecnologia a um nível completamente diferente, surgiu em 2004 o *Web Hypertext Application Working Group (WHATWG)* que criou a especificação do *HTML5*.

Em 2006 o W3C tomou conta da situação o que fez com que em 2008 estivesse pronta a primeira versão funcional do *HTML5* ainda em rascunho.

Devido ao facto de resolver problemas bastante práticos, as empresas produtoras dos *browsers* têm feito um grande esforço para incorporar o suporte ao *HTML5*, impulsionando a sua evolução. Assim podemos esperar que esta tecnologia resolva não só os problemas relacionados com a falta de uniformização em relação à *web 2.0* mas também da *web 3.0* ou *web* semântica. [LAS10]

Diferenças O *HTML5* introduziu bastantes novidades em relação à sua versão anterior como novas *tags*, uma forma uniformizada de validação de formulários, suporte de conteúdos multimédia, e novas funcionalidades que aproximam as aplicações web dos dispositivos onde são executadas. As novas funcionalidades que serão agora analisadas são as que mais interessam no contexto desta dissertação.

Em primeiro lugar destaca-se o suporte de conteúdos multimédia. Com a introdução das *tags* `<audio>` e `<video>`, passou a ser feito um suporte nativo destes formatos, deixando de estar a reprodução destes ficheiros dependente de *plugins* como o *adobe flash* ou o *quicktime*. Isto permite aos programadores integrar estes conteúdos mais facilmente nas suas aplicações utilizando CSS e ao mesmo tempo garante que estes conteúdos podem ser executados em qualquer plataforma, incluído qualquer *tablet*. A *tag* `<canvas>`, oferece uma superfície onde podem ser desenhadas figuras, criadas animações, entre outras funcionalidades gráficas. Isto permite que aplicações gráficas ou jogos possam também passar a ser suportadas nativamente sem a necessidade de recorrer a tecnologias de terceiros.

Outro destaque são três funcionalidades que permitem a aproximação da aplicação *web* com o dispositivo em que estão a ser executadas. A primeira é a possibilidade de utilizar o ***drag and drop*** para mover ficheiros, anexar documentos e manipular interfaces, eliminando a necessidade de utilizar complexos menus para desempenhar essas tarefas que apenas criam uma ideia de afastamento da aplicação em relação ao dispositivo. Em segundo lugar temos a **localização geográfica** que permite, utilizando as capacidades dos dispositivos para saber a localização dos utilizadores, adequar as informações fornecidas ao mesmo a esse conhecimento, como por exemplo recomendar a localização do restaurante ou bomba de gasolina mais próxima. Por último, a mais importante destas funcionalidades é o **armazenamento local e execução offline**. Esta funcionalidade é oferecida através de uma *API* própria que permite às aplicações continuarem a ser executadas *offline* fazendo um *caching* local da informação da aplicação. Quando a conexão é reestabelecida, a aplicação sincroniza a informação alterada e atualiza conteúdos que sejam necessários. Desta forma é possível ter uma experiência de trabalho ininterrupta semelhante à presente em dispositivos *offline*. [ND11]

O *HTML5* possibilita ainda, uma melhor interação com *smartphones* e *tablets* permitindo o reconhecimento de gestos. Com esta tecnologia, uma aplicação *web* consegue lidar com eventos especiais, lançados por estas plataformas como gestos pre-definidos (por exemplo *zoom in* e *zoom out*) ou mesmo a alteração da orientação do dispositivo o que permite uma grande adaptação da aplicação ao dispositivo, quase como se esta estivesse a ser executada no próprio. [LAS10]

Revisão Bibliográfica

Apesar de todas estas inovações, e novas portas que o *HTML5* abre, esperam-se ainda desenvolvimentos tremendos nesta área, que encurtarão cada vez mais as distâncias entre *web* e *nativo*, tornando as diferenças entre estas praticamente imperceptíveis.

Capítulo 3

Desenvolvimento

Neste terceiro capítulo é apresentado o desenvolvimento do projeto. É feito em primeiro lugar um aprofundamento do problema em questão, identificando as áreas onde a aplicação deverá atuar. Em seguida são descritos os requisitos e funcionalidades a implementar. Para perceber a forma como a aplicação foi projetada, é feita uma análise da arquitetura da mesma e por último, é demonstrada a implementação da solução.

3.1 O problema

Tal como já referido na introdução desta dissertação, na secção 1.1, o principal problema que motiva esta dissertação é a visualização dos resultados de *MCDTs* por parte dos médicos. Como também já foi referido, a *Glintt HS* possui já uma solução para desempenhar esta tarefa, designada de *eResults*, analisada na secção 2.1 do segundo capítulo.

Uma problemática relacionada com a consulta de resultados *MCDTs* trata-se da sua portabilidade. Um médico pode necessitar de consultar determinados resultados num ambiente que não seja no seu consultório, onde tem acesso ao seu computador pessoal, como seja por exemplo, numa sala de tratamentos, numa sala de operações ou até numa reunião com outros colegas para partilhar opiniões. Tendo estas situações em atenção, foi chegado à conclusão que o ideal seria desenvolver uma aplicação para dispositivos móveis (neste caso *tablets*) que permitisse realizar as mesmas tarefas já existentes no *eResults*, adicionando portabilidade à equação.

Apesar de o *eResults* ser uma aplicação web, a solução em cima da mesa para este problema nunca poderia ser a simples abertura da sua versão atual num *browser* de um dispositivo móvel. A principal tecnologia utilizada no seu desenvolvimento foi o *Microsoft Silverlight*, que não é propriamente uma tecnologia indicada para dispositivos móveis, ao mesmo tempo que as dimensões dos componentes da aplicação não proporcionariam uma usabilidade aceitável para os utilizadores. O ideal seria então o desenvolvimento de uma aplicação nativa para estes dispositivos, ou uma

aplicação móvel adaptada a *mobile*, sem ter que desenvolver uma aplicação por cada dispositivo alvo.

Os principais problemas a solucionar pela aplicação serão:

Reprodução multimédia Os resultados a serem consultados, possuem uma enorme variedade de formatos multimédia, sendo os principais vídeo, imagem, e *pdf*. A aplicação deverá ser capaz de lidar com todos estes formatos de uma forma transparente para o utilizador, sem recurso a ferramentas externas, ou seja, completamente “embebida” na aplicação. Embora os *tablets* estejam vocacionados para lidar com multimédia, estes ficheiros podem ter várias extensões diferentes, sendo que os vídeos presentes na aplicação são essencialmente *mp4* e *mov* e as imagens *jpg*, *png* e *tiff*. De certa forma a aplicação fica refém da capacidade de cada dispositivo de reproduzir estas extensões ou não.

Acesso a webservices Para que possam ser consultados os resultados, estes têm que ser acedidos na base de dados onde estão armazenados. Este processo é feito através do recurso a *webservices*. Esta funcionalidade é extremamente importante, uma vez que todos os dados relevantes para a aplicação não se encontram no dispositivo, mas sim nas bases de dados alojadas em servidores da rede interna de cada hospital. Do ponto de vista do utilizador, é importante que este acesso seja completamente transparente e veloz, como se os dados estivessem disponíveis no próprio dispositivo. Do ponto de vista do desenvolvimento é importante que o acesso aos serviços seja feito de forma simples e direta, com o mínimo de código possível, ou seja, sem necessidade de tratamento de baixo nível na chamada e receção da resposta do servidor (construção de envelopes da chamada, decomposição da resposta).

Notificações Uma funcionalidade extra que se pretende adicionar à aplicação são as notificações. Recorrendo a alguma forma de notificação, seja síncrona ou assíncrona (*push notifications*), pretende-se que um médico possa ser notificado da existência de um novo resultado na base de dados que seja do seu interesse.

3.2 Requisitos e funcionalidades

3.2.1 Atores

Durante a fase de análise do problema foi identificado apenas um grupo de utilizadores, que foi designado como “utilizador”.

Utilizador A nível operacional, esta aplicação será utilizada por médicos, sendo estes os únicos atores do sistema, por “utilizadores”. Este grupo de utilizadores efetua todo o tipo de ações no sistema, nomeadamente a pesquisa de resultados, a consulta dos mesmos e o registo de notificações.



Figura 3.1: Diagrama de Atores

A administração do sistema, nomeadamente tarefas como a inserção de novos utilizadores e a monitorização do sistema, são feitas numa camada externa à aplicação e por esse motivo, esse tipo de ator não é apresentado com ator do sistema.

3.2.2 Requisitos da Solução

São agora apresentados os requisitos da solução, tanto a nível funcional, como a nível não-funcional, provenientes da fase de análise e especificação de requisitos do projeto.

3.2.2.1 Requisitos Funcionais

Os requisitos funcionais definem as ações que o sistema deve permitir aos utilizadores realizar, ou seja, todas as funcionalidades que o sistema possui. O sistema deverá implementar as seguintes funcionalidades:

- Autenticar do utilizador, com base nos seus dados de autenticação do eResults;
- Permitir a pesquisa dos resultados de um ou vários doentes, através de uma barra de pesquisa simples, ou detalhada;
- Visualizar os resultados de um doente, independentemente do seu formato ser uma imagem, vídeo, ou *pdf*;
- Manipular os mesmos resultados, especialmente a sua ampliação (imagens) para permitir a observação com maior detalhe;
- Inserir pedidos de notificações;
- Receber notificações dos novos resultados pedidos;

3.2.2.2 Requisitos Não-Funcionais

Os requisitos não funcionais, não dependem de uma funcionalidade específica mas dizem respeito ao funcionamento do sistema em geral.

Requisitos de qualidade São definidos os seguintes requisitos referentes à qualidade do sistema:

- **Eficiência** - O sistema deve ser rápido e fluído nas transições entre módulos/separadores e deverá ser capaz de processar os pedidos ao servidor (*webservices*) rapidamente mantendo o tempo de espera para o utilizador em níveis mínimos e toleráveis;
- **Consistência** - O sistema deve assegurar a integridade da informação enviada para o servidor;
- **Segurança** - O sistema deve, através de uma proteção por *password*, assegurar que apenas pessoas autorizadas têm acesso à informação presente no servidor;
- **Usabilidade** - É essencial que o sistema seja simples e fácil de operar, de forma a minimizar a curva de aprendizagem dos utilizadores, que sendo profissionais da área da saúde, por norma não reagirão bem a uma nova ferramenta que seja demasiado complexa. Por outro lado é importante que a interação com este sistema não seja muito dissonante do *eResults*, para evitar ao máximo a resistência à mudança;

Requisitos de interface São definidos os seguintes requisitos referentes à interface do sistema:

- **Interface com o utilizador** - Como foi supra-referido é necessário que o sistema seja simples e fácil de utilizar. Essa característica está essencialmente dependente da interface que deverá ser intuitiva. O sistema de navegação por separadores permitirá ao utilizador a qualquer momento aceder facilmente a qualquer funcionalidade e apresenta uma divisão coerente das funcionalidades. Em termos de design e esquema de cores seguirá o design desenvolvido para a renovação do *eResults*, com ligeiras alterações que permitam ter o nível de usabilidade desejado nos *tablets*;
- **Interface de Hardware** - Para poder executar a aplicação será necessário um *tablet* que suporte *Android* ou *IOS*;
- **Interface de Comunicação** - Regra geral o sistema terá acesso à intranet da unidade hospitalar. Isto significa que o sistema não poderá estar dependente de serviços existentes na internet, mas apenas dos que poderão ser disponibilizados na rede local, como é o caso das bases de dados que serão acedidas via *webservices*;

3.2.3 Casos de Uso

São agora apresentados os casos de uso da solução, identificados durante a fase de análise e especificação de requisitos do projeto. A sua apresentação é feita num formato textual e divididos pelos módulos correspondentes na solução. Embora nesta secção sejam apresentados todos os casos de uso identificados, de forma semelhante, deve-se notar que cada caso de uso tem uma importância relativa para a solução final e em termos de relevância de conteúdo para a dissertação, o que leva a que sejam cotados com diferentes níveis de prioridade. Assim, e tal como foram

definidas na secção 3.1 de apresentação do problema, as grandes problemáticas, e os casos de uso relacionados com as mesmas serão endereçadas com maior prioridade.

Uma vez que toda a aplicação gira à volta dos resultados de *MCDTs*, os principais casos de uso estão relacionados então com a visualização de conteúdos multimédia, sendo que esta problemática será a primeira a ser abordada. Com igual importância estão também os casos de uso relacionados com a pesquisa de doentes que abordam a problemática de acesso a dados. Temos também a questão das notificações, que será abordada após estas duas e apresenta também um elevado grau de prioridade, ainda que inferior aos dois já mencionados. Estas problemáticas são abordadas nos módulos 3.2.3.2 (pesquisa de doente), 3.2.3.3 (visualização multimédia) e 3.2.3.4 (notificações). Ainda dentro de cada módulo existem também diferentes níveis de prioridade, que podem ser analisados com mais detalhe na apresentação dos casos de uso em formato tabular, no anexo A. Cada caso de uso apresenta uma ligação para a sua respectiva tabela.

3.2.3.1 Módulo de autenticação

São agora apresentados os casos de uso respeitantes ao módulo de autenticação. Na figura 3.2 é possível observar o modelo de casos de uso deste módulo.

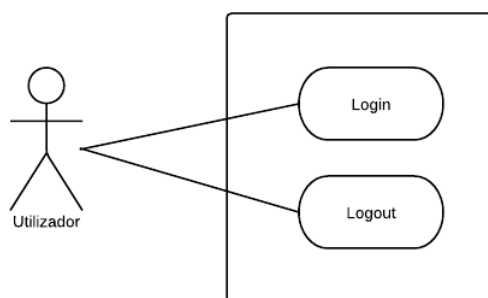


Figura 3.2: Modelo de casos de uso do módulo de autenticação

CU01 - Login O utilizador deve ser capaz de se autenticar no sistema, com recurso às suas credenciais (“Username” e “Password”) do *eResults*. Tabela: A.1

CU02 - Logout Depois de se autenticar no sistema, o utilizador deve ser capaz de em qualquer momento (independentemente do separador em que se encontra) desautenticar-se ou fazer “logout”. Tabela: A.2

3.2.3.2 Módulo de pesquisa de doentes

São agora apresentados os casos de uso respeitantes ao módulo de pesquisa de doentes. Na figura 3.3 é possível observar o modelo de casos de uso deste módulo.

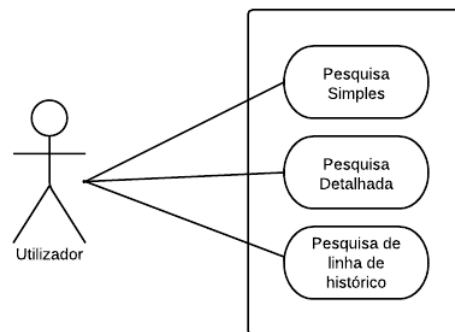


Figura 3.3: Modelo de casos de uso do módulo de pesquisa de doentes

CU03 - Pesquisa simples Acedendo ao separador “Documentos” o utilizador deve ser capaz de realizar uma pesquisa simples para encontrar os resultados de um determinado doente. Esta pesquisa é realizada através de uma barra de pesquisa simples com quatro campos, dos quais é necessário preencher pelo menos um. Tabela: [A.3](#)

CU04 - Pesquisa detalhada Acedendo ao separador “Documentos” o utilizador deve ser capaz de realizar uma pesquisa detalhada para encontrar os resultados de um determinado doente. Esta pesquisa é realizada através de uma barra de pesquisa detalhada que é uma extensão da barra de pesquisa simples, revelada pressionando no botão “mais” presente na mesma. Tabela: [A.4](#)

CU05 - Pesquisa de linha de histórico Acedendo ao separador “Histórico” o utilizador deve ser capaz de realizar uma pesquisa para encontrar a linha de histórico de um determinado doente. Esta pesquisa pode ser feita utilizando apenas a barra de pesquisa simples, ou também a barra de pesquisa detalhada. Tabela: [A.4](#)

3.2.3.3 Módulo de visualização de multimédia

São agora apresentados os casos de uso respeitantes ao módulo de visualização de multimédia. Na figura [3.4](#) é possível observar o modelo de casos de uso deste módulo. Tabela: [A.5](#)

CU06 - Visualiza PDF Se ao abrir um determinado resultado, este for de formato *pdf*, o utilizador deve ser capaz de visualizar o documento, sendo apresentado numa janela própria para o efeito e tendo uma apresentação semelhante à que teria utilizando a aplicação *Acrobat Reader*. Tabela: [A.6](#)

CU07 - Visualiza Imagem Se ao abrir um determinado resultado, este for de formato imagem, o utilizador deve ser capaz de visualizar o documento, sendo apresentado numa janela própria para o efeito. Tabela: [A.7](#)

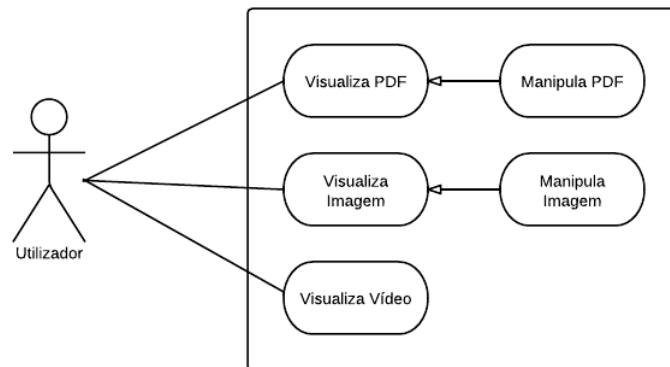


Figura 3.4: Modelo de casos de uso do módulo de visualização de multimédia

CU08 - Visualiza Vídeo Se ao abrir um determinado resultado, este for de formato vídeo, o utilizador deve ser capaz de visualizar o documento, sendo apresentado numa janela própria para o efeito, permitindo a interação natural com uma aplicação de reprodução de video, sendo o carregamento do vídeo feito em simultâneo com a sua reprodução, para diminuir a latência desta ação (para que o utilizador não ter que esperar que o vídeo esteja completamente carregado para poder começar a visualização do mesmo). Tabela: [A.8](#)

CU09 - Manipula PDF O utilizador deve ser capaz de manipular um resultado em formato *pdf* de uma forma semelhante ao que faria na aplicação *Acrobat Reader*, nomeadamente, mudando de página com botões específicos para o efeito e ampliando a imagem com recurso a botões ou gestos *multitouch*. Tabela: [A.9](#)

CU10 - Manipula Imagem O utilizador deve ser capaz de manipular um resultado em formato imagem, sendo capaz de ampliar e rodar a imagem utilizando gestos *multitouch*. Tabela: [A.10](#)

3.2.3.4 Módulo de notificações

São agora apresentados os casos de uso respeitantes ao módulo de notificações. Na figura [3.5](#) é possível observar o modelo de casos de uso deste módulo.

CU11 - Adiciona pedido de notificação O utilizador deve ser capaz de adicionar pedidos de notificação, “subscrevendo” os resultados de um determinado doente (ou apenas determinados tipos de resultados), aquando da pesquisa dos resultados de um doente (no separador “Documentos” ou “Histórico”). Tabela [A.11](#)

CU12 - Consulta notificações Quando existe uma nova notificação, o utilizador deve ser avisado através de um sinal numérico sobre o separador “Notificações” (que identifica quantas notificações novas tem) ou de outras formas que o utilizador tenha definido (*email*, ou *sms*), podendo consultar as suas notificações não lidas no separador “Documentos”. Tabela [A.12](#)

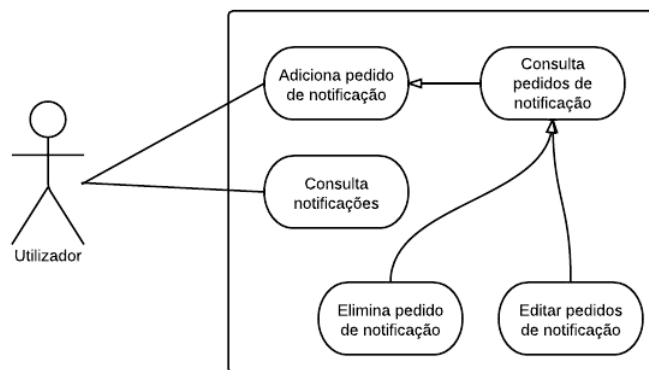


Figura 3.5: Modelo de casos de uso do módulo de notificações

CU13 - Consulta pedidos de notificação O utilizador deve ser capaz de consultar todos os seus pedidos de notificações pendentes, no separador “Documentos” pressionando o botão “Ver pedidos”. Tabela [A.13](#)

CU14 - Edita pedido de notificação Ao consultar os seus pedidos de notificação pendentes, o utilizador deve ser capaz de editar os dados referentes a esse pedido exceto o doente a que está afecto. Tabela [A.14](#)

CU15 - Elimina pedido de notificação Ao consultar os seus pedidos de notificação pendentes, o utilizador deve ser capaz de eliminar um determinado pedido, através de um botão “eliminar” sobre o pedido. Tabela [A.15](#)

3.3 Arquitetura

3.3.1 Componentes físicos necessários

Os componentes físicos da aplicação estão dispersos em vários locais distintos, fazendo com que a aplicação seja distribuída. A organização destes componentes é apresentada na figura [3.6](#).

Os diferentes componentes são:

- **Servidor de base de dados** - É o componente onde se encontra a maioria da informação da aplicação. Está alojado num servidor acessível pela rede local (*LAN*);
- **Servidor aplicacional** - Esta é a componente central da aplicação. Aqui é feito todo o processamento necessário ao funcionamento da aplicação. Como foi já descrito anteriormente, este componente é responsável por processar todos os pedidos vindos das máquinas cliente, tratando da interação necessária com a base de dados para apresentação dos resultados, tratamento e envio da resposta novamente para as máquinas cliente.

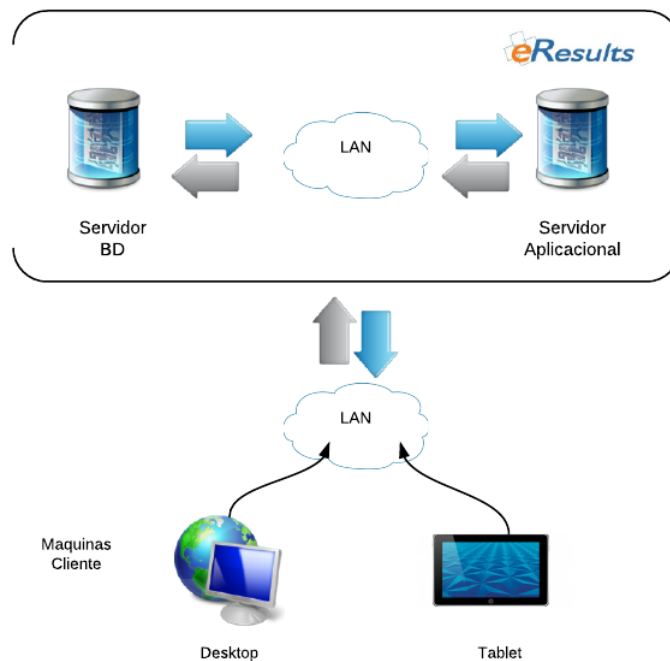


Figura 3.6: Organização dos componentes físicos da aplicação

- **Maquinas cliente** - Este é o único componente visível para o utilizador. É responsável pela interação com o mesmo, recebendo o seu *input* e transformando o mesmo em pedidos para o servidor aplicacional via *webservices* sob a rede local.

Tal como é demonstrado na figura 3.6, todos os componentes da aplicação estão acessíveis através da rede local, sem dependência da *internet*.

Por último, é importante analisar uma situação que está relacionada com a disponibilização dos resultados dos *MCDTs*. Quando é feita uma pesquisa dos resultados de um determinado doente, a aplicação é informada da localização de cada um dos resultados, através dos endereços *URL* onde cada um se encontra. Estes endereços não têm que estar localizados necessariamente na base de dados, podendo estar em outros servidores de ficheiros da rede local ou até em servidores externos.

3.3.2 Módulos de software

Esta aplicação deverá, como já referido anteriormente, disponibilizar ao utilizador a maioria das funcionalidades presentes no *eResults*. Desta forma alguns dos módulos de software em que a mesma é dividida, são herdados do *eResults*, tendo alguns sido suprimidos e um novo acrescentado. Os módulos da solução são apresentados na figura 3.7 e são os seguintes:

Documentos

O módulo de “Documentos” é um dos elementos centrais da aplicação, incluindo os componentes mais importantes da mesma.

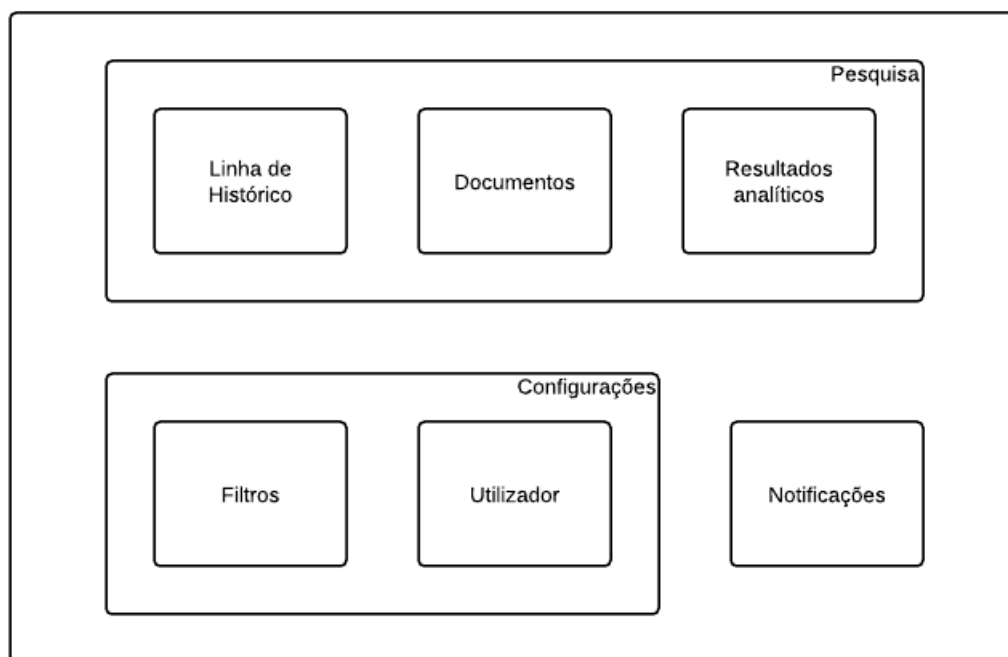


Figura 3.7: Módulos e componentes de software da solução

Neste é possível fazer a pesquisa de um doente, inserindo os dados do mesmo nas barras de pesquisa disponibilizadas para o efeito. Uma das barras apresenta um conjunto mais reduzido de campos, sendo designada “barra de pesquisa simples”. Esta barra disponibiliza um botão que permite fazer a sua expansão, mostrando uma barra com um conjunto mais alargado de campos, designada “barra de pesquisa detalhada”. Depois de feita a pesquisa do doente, são apresentados os seus resultados de *MCDTs*, em formato de lista ou “minitaturas”. Por último, escolhendo um dos resultados, é possível fazer a visualização do mesmo, independentemente do seu formato.

Linha de histórico

Este módulo é extremamente semelhante ao módulo de documentos, permitindo os mesmos tipos de pesquisa (simples e detalhada) e também a visualização dos resultados de *MCDTs*. O que distingue este módulo do anterior é a forma como a lista dos resultados de um doente é apresentada. Esta apresentação é feita por ordem cronológica, existindo várias opções de agrupamento dos resultados (por mês, ano, entre outras), permitindo ao utilizador ter uma perceção do histórico clínico do doente em questão.

Resultados analíticos

O módulo de “Resultados analíticos” permite ao utilizador a consulta de resultados, que chegam à aplicação sob a forma de um conjunto de indicadores. A apresentação de cada resultado é feito em formato tabular, mostrando não só o valor de cada indicador, como também se este se encontra ou não dentro dos valores normais para o mesmo. Na presença de mais do que um exame

do mesmo tipo é possível também fazer uma análise gráfica da evolução dos valores dos indicadores ao longo do tempo, permitindo ao utilizador tirar outro tipo de conclusões de uma forma muito mais simples do que se necessitasse de analisar os valores um a um.

Filtros

O módulo de “Filtros” permite ao utilizador a definição de um conjunto de filtros que podem ser aplicados a todas as pesquisas.

Utilizador

Este módulo permite ao utilizador editar os seus dados na aplicação, através de um simples formulário

Notificações

O módulo de “Notificações”, é um módulo completamente novo, que não existia ainda no eResults. Este módulo permite que um utilizador subscreva os resultados de determinados doentes, de forma a ser informado, assim que novos resultados estejam disponíveis. Esta funcionalidade permite ao utilizador fazer um acompanhamento muito mais eficaz dos seus doentes, aumentando a celeridade de muitos episódios clínicos. Existe um separador específico para que o utilizador possa consultar os novos resultados relacionados com as suas “subscrições”, bem como a edição ou remoção de pedidos de notificações que estejam ainda pendentes. No entanto, para fazer uma subscrição a um determinado doente o utilizador deve fazer uma pesquisa pelo doente no separador de “Documentos” ou “linha de Histórico”, pressionando posteriormente no botão “subscrever” após encontrar o doente em questão. Esta subscrição pode ser de qualquer resultado referente ao doente ou pode ser configurado apenas para resultados específicos. É também possível escolher o tipo de notificações que se pretende receber, sendo possível para além das notificações no próprio separador, receber um *email* ou *sms*.

Como é possível perceber pela descrição dos módulos da aplicação, bem como pela apresentação dos casos de uso na secção [3.2.3](#), alguns destes módulos possuem maior importância para a solução final, e também um maior grau de complexidade. O módulo central e prioritário é o módulo de “Documentos”, uma vez que concentra em si as funcionalidades mais importantes da aplicação, nomeadamente a pesquisa de doentes e a apresentação dos resultados e manipulação dos seus formatos multimédia. Num nível inferior, temos o módulo de “Notificações”, pela sua inovação e novas funcionalidades para o utilizador. De seguida, com menos prioridade temos o módulo de “Linha de histórico” e o de “Resultados Analíticos” e por último os módulos de configurações dos “Filtros” e “Utilizador”.

3.3.3 Modelo de dados

É agora analisado o modelo de dados da solução a desenvolver. Este modelo tem como principal foco as componentes referentes ao módulo de notificações, uma vez que estas serão as novas entidades e relação que têm que ser implementadas, mas também a forma como as entidades relevantes para a solução (que são acedidas via webservices) estão representadas na base de dados.

O modelo referente às notificações é apresentado na figura 3.8, e é composto pelas seguintes entidades:

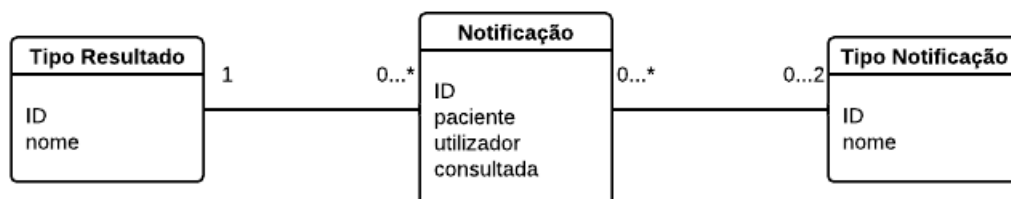


Figura 3.8: Modelo de dados da componente de notificações

- **Tipo de Resultado** - Representa um tipo de resultado, que pode ser por exemplo “Imagiologia” ou “Cardiologia” entre outras. O seu papel é o de sinalizar se um pedido de notificação deve ser restrito a um determinado tipo de resultado. Tem uma relação “muitos para muitos” com a entidade “Notificações”, o que significa que um tipo de resultado pode dizer respeito a muitas notificações, bem como uma notificação pode ter muitos tipos de resultados;
- **Tipo de Notificação** - Representa um tipo de notificação, sendo os seus valores possíveis “sms” ou “email”. O seu papel é o de sinalizar se um pedido de notificação deve ser enviado pelos meios descritos ou não. Tem uma relação “muitos para muitos” com a entidade “Notificação” (do lado do tipo de notificação é no máximo dois), o que significa que um tipo de notificação pode dizer respeito a muitas notificações, e que uma notificação pode ter muitos tipos de notificações (dois);
- **Notificação** - Representa uma notificação. É a entidade central deste modelo, agrupando as outras entidades. Uma notificação diz respeito a um determinado doente, uma vez que é uma subscrição dos novos resultados do mesmo. Outro atributo da notificação é o utilizador que fez o seu pedido, para que seja possível enviar as notificações para o o seu requisitante. Estes dois atributos presentes no diagrama representam referências para estas entidades na base de dados do *eResults*. O atributo “consultada” indica se essa notificação já foi ou não consultada pelo utilizador. A associação com a entidade “Tipo de resultado” permite fazer uma restrição das notificações. Caso exista algum tipo de resultado associado a uma notificação, serão enviadas notificações apenas se os novos resultados forem desse(s) tipo(s), caso contrário serão enviadas notificações bastando que os novos resultados sejam referentes ao doente em questão. A associação com a entidade “Tipo de notificação” permite adicionar

novos tipos de notificações, para além da notificação “standard” que o utilizador recebe na aplicação, no separador “Notificações”. Esses tipos de notificações podem ser *sms* ou *email*;

As restantes entidades utilizadas na aplicação, são referentes à base de dados do *eResults*, que foi mantida inalterada, tendo sido utilizada como “caixa-negra”, através de chamadas aos *webservices*. Esta base de dados é bastante extensa e complexa, e por esse motivo, serão apenas apresentadas as entidades com as quais existiu interação no projeto, que podem ser observadas na figura 3.9. As entidades são as seguintes:

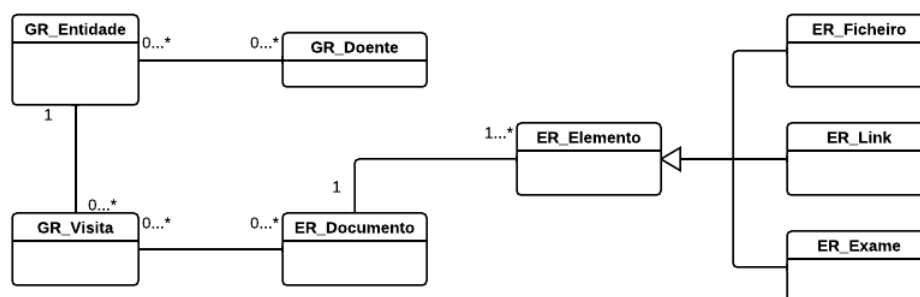


Figura 3.9: Modelo de dados das componentes do *eResults* utilizadas na aplicação

- **GR_Entidade** - Representa uma pessoa/utilizador. Esta entidade apresenta apenas os dados genéricos de uma pessoa, os restantes dados mais específicos de cada tipo (médico, doente, etc) estão presentes em outras tabelas associadas a esta. Tem uma relação “muitos para muitos” com a entidade “GR_Doente” e uma relação “um para muitos” com a entidade “GR_Visita”;
- **GR_Doente** - Representa um doente. Tem uma relação “muitos para muitos” com a entidade “GR_Entidade”, completando assim a sua informação;
- **GR_Visita** - Representa uma visita de um utilizador à unidade hospitalar. Tem uma relação “um para muitos” com a entidade “GR_Entidade”, uma vez que uma pessoa pode realizar um número indeterminado de visitas. Tem uma relação “muitos para muitos” com a entidade “ER_Documento”, uma vez que numa visita podem ser produzidos muitos resultados, e um documento pode dizer respeito a mais do que uma visita;
- **ER_Documento** - Representa um documento. Tem uma relação “muitos para muitos” com a entidade “GR_Visita” como já explicado e uma relação “um para muitos” com a entidade “ER_elemento”, uma vez que um documento pode conter vários elementos (por exemplo, várias imagens);
- **ER_Elemento** - Representa um elemento. Diz respeito a um documento, tal como indica a relação “um para muitos” já apresentada com essa entidade. Tem uma relação de generalização com as entidades “ER_Ficheiro”, “ER_Link” e “ER_Exame”, ou seja pode ser uma destas três entidades.

- **ER_Ficheiro** - Representa um ficheiro. É um dos tipos de elementos que um documento pode ter.
- **ER_Link** - Representa um *link* para um ficheiro externo. É um dos tipos de elementos que um documento pode ter.
- **ER_Exame** - Representa um resultado analítico. É um dos tipos de elementos que um documento pode ter.

3.3.4 Ligações e interações

São agora explicadas as ligações entre os vários componentes e módulos de software, de forma a que seja mais perceptível o fluxo de ações processadas na aplicação. Uma visão geral destas interações é apresentada na figura 3.10

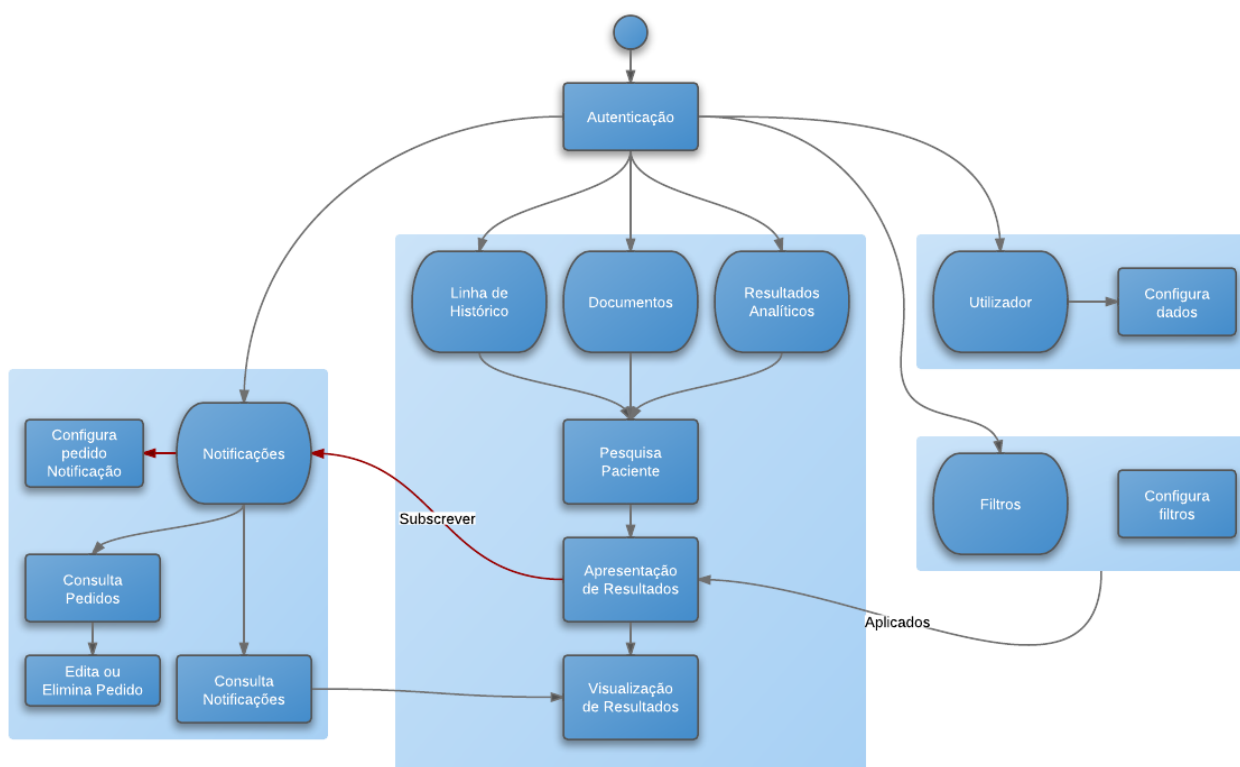


Figura 3.10: Diagrama de fluxo e interações entre módulos da aplicação

Ao iniciar a aplicação o utilizador necessita de se autenticar para poder prosseguir. Após estar autenticado e de ser mostrado o menu de boas vindas, o utilizador pode escolher um dos seis separadores, cada um representando um dos módulos da aplicação. Como é possível observar na figura 3.10, o módulo central é o módulo “Documentos”, cujos componentes são partilhados pelos módulos de “Linha de histórico” e “Resultados Analíticos” (nesta secção o conjunto destes três módulos será designado como módulo “Central”), e são a “pesquisa de doente”, a “apresentação de resultados” e “visualização de resultados” (multimédia). O módulo “Notificações” é também bastante

importante tendo os componentes “configura pedido de notificação”, “consulta de notificações” e “consulta/edita/elimina pedidos de notificações”. Os restantes módulos são menos importantes, sendo que o módulo “Filtros” apenas tem o componente “configura filtro” e o módulo “Utilizador” apenas tem o componente de “configura dados”.

Com exceção do módulo “Utilizador”, todos os módulos interagem com outros módulos, nomeadamente, os módulos “Notificação” e “Filtros” interagem com o “módulo central”. A interação do módulo “Central” com o módulo “Notificações” é feita em dois pontos: quando são apresentados os resultados de um doente, é possível fazer a subscrição dos resultados do mesmo, abrindo o módulo de notificações, e quando o utilizador consulta as suas notificações pode visualizar o(s) resultado(s) referentes à notificação abrindo a componente de visualização de resultados do módulo “Central”. A interação com o módulo filtros é feita ao nível da apresentação de resultados, uma vez que os filtros que são definidos atuam a este nível.

3.4 Implementação

Nesta secção é feita em primeiro lugar uma análise das tecnologias exploradas na fase inicial do projeto e em segundo lugar a apresentação dos detalhes da implementação dos vários módulos da aplicação.

A implementação é abordada, tomando como módulos de implementação as grandes áreas de intervenção na aplicação em deterioramento dos módulos em que a aplicação se encontra dividida (como referido na secção 3.3.2).

3.4.1 Escolha das tecnologias de implementação

Para escolher as tecnologias mais indicadas para a implementação da solução, foi necessário explorar várias alternativas, por forma a encontrar as que melhor satisfizessem as necessidades da solução, ao nível das principais áreas da aplicação, nomeadamente o acesso aos *webservices*, a reprodução de multimédia e as notificações. É de notar que esta escolha é sempre limitada pelas próprias restrições da empresa, seja a nível de preferência ou monetário, uma vez que existem já várias tecnologias licenciadas pela empresa e qualquer investimento extra deverá ser sempre ponderado.

Para o acesso a *webservices* a tecnologia escolhida foi o *WCF*, uma vez que existia já uma estrutura montada nesta tecnologia o que fez com que o trabalho fosse extremamente reduzido, faltando apenas definir a forma como a sua chamada seria feita do lado do cliente.

A escolha de tecnologia mais importante passou pela definição da tecnologia de desenvolvimento multi-plataforma. Na fase inicial do projeto, a exploração destas tecnologias foi feita tendo em mente o desenvolvimento de uma aplicação nativa, para *iOS* e *Android* e nesse sentido a primeira a ser explorada foi o *Appcelerator Titanium*. A sua escolha em relação às outras *frameworks* (por exemplo o *Rhodes*) prendeu-se pelo facto de esta ser uma *framework* bastante robusta e já existir experiência na empresa do desenvolvimento de aplicações com a mesma e o licenciamento da sua versão comercial. Foram desenvolvidos vários protótipos com resultados

satisfatórios em relação à reprodução de multimédia, mas com bastantes problemas ao nível do acesso aos serviços. Foram analisadas várias estratégias com recurso a ferramentas externas, já mencionadas no segundo capítulo, na secção 2.3.3.1 (por exemplo *sudzc* ou *wsdl2js*, entre outros) e que envolviam também a necessidade de criar um serviço intermédio que funcionava como uma interface com o serviço *WCF*. Para agravar a situação, esta tecnologia não oferecia um método *standard* para a estruturação do código no projeto, o que fazia com que a entropia do mesmo aumentasse exponencialmente com a complexidade do projeto. Por estes motivos esta solução teve de ser abandonada.

Foram feitos ligeiros testes com a plataforma de desenvolvimento *MonoTouch/Mono for Android*, que rapidamente foi abandonada devido ao facto de ser necessária a replicação de código entre plataformas e devido ao elevado custo da versão comercial plataforma.

Na procura de uma solução mais estruturada, e mais próxima das tecnologias *web* surgiu a *framework Sencha Touch*. Esta *framework* permite uma maior organização de código devido ao facto de seguir uma estrutura *MVC* e revela uma curva de aprendizagem extremamente rápida. Apesar das aplicações desenvolvidas nesta tecnologia serem essencialmente aplicações *web*, estas podem ser facilmente “empacotadas” para aplicações nativas *iOS* ou *Android* através da ferramenta *sencha touch tools*. Foram feitos também vários protótipos para perceber se esta *framework* era capaz de desempenhar as principais tarefas requeridas para a aplicação, e os resultados embora minimamente satisfatórios, não foram tão bons como seria expectável. Na reprodução multimédia foram encontrados problemas essencialmente ao nível da visualização de *pdf's* e a forma de acesso a *webservices* é feita segundo a mesma estrutura apresentada no *Appcelerator Titanium*.

Nesta fase e também devido à proximidade que a exploração estava a ter das tecnologias *web* a empresa começou a estudar a possibilidade de enquadrar esta solução com a renovação do *eResults* (aplicação *web* em ambiente *desktop*), desenvolvendo uma solução única que fosse capaz de juntar o melhor dos dois mundos e executar em ambiente *desktop* e *mobile* adaptando-se às necessidades específicas de cada ambiente. Foi estudada a possibilidade do desenvolvimento dessa solução ser feita utilizando a *framework Sencha Touch*, sendo a parte *desktop* seria feita com recurso à *API Ext JS 4.0*. Esta hipótese esbarrou no elevado preço da versão comercial dessa *API* e devido à incerteza se a *framework Sencha Touch* continuaria a ser disponibilizada gratuitamente.

A solução encontrada para resolver todos estes problemas, que foi a tecnologia escolhida para o desenvolvimento da solução, passou pela utilização da *framework ASP.NET MVC* (que a empresa já possui licenciada) para o desenvolvimento de toda a lógica de negócio, acesso a dados e apresentação no ambiente *desktop*, combinada com a utilização da *framework jQueryMobile* (*open source*), para a apresentação no ambiente *mobile* (*front-end*) conferindo-lhe o grau de usabilidade necessário para este ambiente.

A *framework ASP.NET MVC*, é propriedade da empresa *Microsoft* e a sua versão atual é a *ASP.NET MVC 3*. Esta *framework* permite aliar a arquitetura *MVC* à popular tecnologia de desenvolvimento *ASP.NET*, incluindo todas as funcionalidades aliadas à mesma. O resultado é uma *framework* leve mas extremamente robusta que oferece garantias sólidas a nível da sua manutenção. Outra das características e funcionalidades apresentadas por esta *framework* passa pela disponi-

bilização do motor de visualização *Razor view engine*, que permite a fácil injeção de código *C#* ou *Visual Basic* nas *Views* (utilizando ficheiros *.cshtml* ou *.vbhtml* respectivamente), aumentado a fluidez da codificação e tornando as *Views* mais dinâmicas. [Pop12]

A escolha da *framework jQueryMobile* para a adaptação ao ambiente *mobile* deveu-se à sua facilidade de integração com qualquer tecnologia *web* (não sendo o *ASP.NET MVC* uma exceção) e devido à robustez associada ao *jQuery* que é uma das tecnologias mais utilizadas no desenvolvimento *web*, nos dias de hoje.

Uma das problemáticas que motivou mais exploração e também algum tempo de desenvolvimento foi a reprodução de *pdf's*. Esta componente é de importância extrema para a aplicação, uma vez que a grande maioria dos resultados presentes na base de dados do *eResults* são deste tipo. Como referido no capítulo da revisão bibliográfica na secção 2.3.3.5 a solução pretendida pela empresa tinha duas grandes restrições: não poderia depender da *internet*, uma vez que aplicação será executada apenas com acesso uma rede local ou *intranet*, e não deveria depender da abertura de aplicações externas, devendo ser completamente “embebida” na aplicação. Depois de exploradas as alternativas na secção supra-referida, como por exemplo o *flexpaper*, o *Vuzit*, o *pdfjs* entre outros, foi desenvolvido o componente de visualização de *pdf's* (“pdfViewer”), cujo funcionamento e implementação é explicado na descrição do módulo de visualização de multimédia na secção 3.4.3.1.

3.4.2 Detecção de ambiente mobile

Para que a aplicação seja capaz de distinguir se está a ser executada em ambiente *mobile* ou *desktop*, é necessário utilizar alguma ferramenta que faça essa interpretação, que pode ser extremamente subjetiva, dado a variedade de dispositivos móveis existentes e aos diferentes tamanhos de ecrã e resoluções que estes possuem. Para o efeito foi utilizada uma ferramenta que vem incluída no *ASP.NET*, na biblioteca *System.web*. Esta ferramenta analisa o pedido *HTTP* enviado pelo *browser*, conseguindo saber se a versão do *browser* é *mobile* ou *desktop*. Assim, esta ferramenta é chamada no ponto de entrada da aplicação (o ficheiro *_viewstart.cshtml*), sendo utilizada para escolher qual ficheiro de *layout* carregar e ao mesmo tempo definir uma variável global *javascript* com esse valor, que estará disponível em todos os scripts que necessitarem de saber em que ambiente a aplicação se encontra, como se pode observar na figura 3.11.

Note-se que esta distinção de ambientes não irá criar duas aplicações diferentes, nem sequer duas camadas de visualização diferentes, mas sim uma adaptação da camada de visualização para que os seus elementos tenham a usabilidade pretendida em ambiente *mobile*, substituindo os componentes da versão *desktop*, por componentes *jQueryMobile*, nas situações que forem necessários. Assim, no ambiente *mobile* serão apenas incluídos os scripts e folhas de estilo necessárias à utilização do *jQuerymobile*.

```

@{
    if(!Request.Browser.IsMobileDevice)
    {
        Layout = "~/Views/Shared/_Layout.cshtml";
        @: <script>var isMobile = false;</script>
    }
    else
    {
        Layout = "~/Views/Shared/_LayoutMobile.cshtml";
        @: <script>var isMobile = true;</script>
    }
}

```

Figura 3.11: Código utilizado para a deteção do ambiente de execução

3.4.3 Módulo de visualização de multimédia

O módulo de visualização multimédia é responsável pelo tratamento dos resultados de *MCDTs* presentes na base de dados, recebendo uma ligação para a localização de um resultado e o seu formato, tratando da sua reprodução da forma indicada.

A atividade deste módulo começa quando o utilizador “clica” ou “pressiona” (conforme o ambiente) sobre um resultado na lista de resultados de um doente (como pode ser observado na figura 3.10 na secção das ligações e interações). Nesse momento é chamada a função “OpenDocument” do controlo da página, designado “DocumentViewerController” que obtém mais detalhes sobre o resultado em questão, disponibilizando os mesmos via *Json*. Do lado cliente, quando é recebida uma resposta e despoletado o método “onsuccess” da chamada ajax, é chamada a função “openDocument” que irá então reproduzir o ficheiro. Esta função seleciona uma *div* que já se encontrava na *view* (“DocumentPresentation”), mas em modo *hidden*, e atribui-lhe o *plugin* “glinttViewer”, como é possível ver na figura 3.12. Esta *div* tem um tamanho fixo no ecrã, estando enquadrada com os restantes elementos da aplicação.

Como é possível observar esta chamada é completamente independente do tipo de documento que será reproduzido, ou seja, essa informação é transparente para a *view*. O tratamento do tipo de documento é feito apenas ao nível do *plugin* “glinttViewer”, que irá verificar qual o tipo de documento em questão e chamar o *plugin* respectivo, nomeadamente o “pdfViewer”, o “imageViewer” ou o “videoViewer”.

Esta solução permite um elevado nível de modularidade à aplicação, fazendo com que esta seja expansível, ou seja, se no futuro for necessário suportar novos tipos de multimédia, bastará adicionar um novo *plugin*, sem ser necessário alterar o código base da aplicação.

3.4.3.1 pdfViewer

O pdfviewer foi sem dúvida o *plugin* mais trabalhoso de implementar. A solução encontrada, foi necessária devido às restrições impostas pela empresa, tal como supra-referido. Esta é total-

```

<script type="text/javascript">
    function openDocument ajaxContext {

        $('#documentviewer resultsfilter area').hide();

        NavigationManager.push(
            function () {
                $('#documentviewer resultsfilter area').show();
                $('#DocumentPresentation').hide();
            }
        );

        if (ajaxContext.hasOwnProperty('Render') &&
            ajaxContext.hasOwnProperty('Type') && ajaxContext.hasOwnProperty('Args')) {
            $('#DocumentPresentation').glinttViewer({
                type: ajaxContext.Type,
                args: ajaxContext.Args,
            });
        }
    }
</script>

```

Figura 3.12: Código utilizado para a chamada do glinttViewer

mente desenvolvida em *javascript*, *HTML* e *CSS*, podendo por isso ser utilizada tanto em ambiente *mobile* como *desktop* e procura ter um aspeto e funcionamento semelhante ao encontrado no *Acrobat Reader*.

A implementação desta solução teve como base o facto de qualquer dispositivo móvel com os sistemas operativos alvo (*iOS* e *Android*) conseguir reproduzir imagens. Assim a ideia para desenvolver um visualizador que funcionasse em qualquer dispositivo passou pela transformação de um ficheiro *pdf* num conjunto de imagens, sendo feito o seu tratamento de forma a simular o ambiente supra-referido.

Ao ser aberto um documento em formato *pdf* é desencadeado um processo do lado servidor que utiliza a ferramenta de linha de comandos *SWFTools*. Em primeiro lugar é utilizado o comando *pdf2swf* para converter o documento de *pdf* para *swf* e de seguida o comando *swfrender* para converter cada página do ficheiro *swf* para uma imagem em formato *png*. Uma vez que o formato *png* pode oferecer alguma perda de qualidade quando as imagens são ampliadas (nomeadamente as letras ficarem distorcidas), existe uma alternativa, um pouco mais morosa, que envolve a conversão para o formato vectorial (*svg*). Neste caso é utilizada a ferramenta *pdftk* para fazer uma divisão do ficheiro *pdf* por páginas e posteriormente a ferramenta *inkscape* para converter cada página num ficheiro vectorial (é necessário dividir o ficheiro *pdf* por páginas uma vez que o *inkscape* apenas converte a primeira de cada ficheiro *pdf*)

Apenas depois deste processo é feita a chamada ao *plugin*, tendo como argumentos o formato em que as imagens se encontram e um *array* com a localização de todas as imagens que compõem o documento. Através de um sistema de *divs* e botões o *plugin* implementa as seguintes

funcionalidades:

- Mudança de página através dos botões “anterior” e “seguinte”;
- Indicação da página atual, numa caixa de texto, calculada através da diferença entre a posição inicial e a deslocação do *scroll* da página;
- Mudança para uma página qualquer (dentro dos limites do documento), alterando o valor numérico na caixa de texto que indica o número de página atual;
- A apresentação do modo “split” (apresentação das páginas em formato duplo), através de um botão com o mesmo nome;

3.4.3.2 imgViewer

O *imgViewer* está dividido em duas iterações. Numa fase inicial a sua ação passa apenas por receber como argumento a localização do documento e colocar o mesmo num elemento *HTML* de imagem (“”), ocupando toda a área disponível. Nesta fase será apenas possível visualizar a imagem e não manipulá-la. Este facto deve-se com a maior prioridade que a visualização tem sobre a manipulação. Numa fase posterior, serão criadas duas barras, uma superior e outra lateral que permitirão a manipulação da imagem em termos de ampliação e rotação. Estes tipos de manipulação também serão implementados em movimentos *multitouch*, com recurso aos eventos do mesmo tipo, suportados pelo *jQuery Mobile*.

3.4.3.3 videoViewer

Este é sem dúvida o *plugin* de mais fácil implementação. O “*videoViewer*”, à semelhança “*imgViewer*”, recebe como argumento a localização do documento, e basta-lhe colocar o mesmo num elemento *HTML* de vídeo. Desta forma, graças às inovações do *HTML5* o documento (vídeo) é reproduzido, no *player* do browser dando ao utilizador todas as capacidade em termos de visualização e manipulação do mesmo. O vídeo é descarregado progressivamente enquanto vai sendo reproduzido, não sendo necessário esperar que o vídeo seja todo descarregado antes de começar a visualização. Em termos de manipulação, existem os comandos de *play* e *pause* e , bem como avançar ou recuar no vídeo deslocando a barra de progresso.

3.4.4 Módulo de acesso a dados

O módulo de acesso a dados é responsável pela interação com o *webservice* do *eResults*. Este *webservice* já existia antes do desenvolvimento desta aplicação, estando implementado em *WCF*.

Ao nível da aplicação foi apenas necessário desenvolver uma interface, que é bastante simplificada devido à utilização da tecnologia *ASP.NET MVC* e foi desenvolvida para a versão *desktop* da aplicação e utilizada na versão *mobile*. Esta interface é implementada na camada de “controlo” da aplicação, da qual é apresentado um cabeçalho na figura 3.13.

```
[HttpPost]
public ActionResult LoadDocuments(string EntityIds, string viewType, int
page)
```

Figura 3.13: Função da chamada ao *webservice* na camada de controlo

O fator mais importante, no que diz respeito ao âmbito da dissertação, está relacionado com a capacidade de chamar esta função na camada de visualização, o que permite a sua manipulação utilizando javascript. A figura 3.14 mostra como esta interação entre a vista e o controlo é feita, injetando código *C#* na mesma.

```
@Ajax.ImageActionLink(
    @doc.ThumbImgSource,
    "OpenDocument",
    new { id = @doc.UniqueId },
    new AjaxOptions {
        HttpMethod = "Post",
        InsertionMode = InsertionMode.Replace,
        UpdateTargetId = "DocumentPresentation",
        OnBegin = "openDocument(" + @doc.UniqueId + ")",
        OnComplete = "openDocumentEnded",
        OnSuccess = "openDocument"
    })
```

Figura 3.14: Interação da camada de visualização com a camada de controlo

Como é possível observar, ao ser disparado o evento “onsuccess”, é chamada a função “openDocument” que é uma função javascript. Esta função por sua vez consegue aceder ao resultado da chamada ao serviço, porque o controlo envia os resultados em formato *JSON*. Assim, para aceder a um atributo do resultado, a função *javascript* utiliza a instrução “ajaxContext.hasOwnProperty(‘X’)”, sendo “X” o nome do atributo enviado.

3.4.5 Módulo de pesquisa de doentes

Este módulo é responsável pela pesquisa dos doentes. Para tal, existem duas barras de pesquisa, uma que está sempre visível que é a barra de pesquisa “Simples” e outra que pode ser revelada por um botão existente na barra de pesquisa “Simples” que é a barra de pesquisa “Detalhada”. A intervenção feita neste módulo está relacionada com os componentes de formulário que necessitam de adaptações para que possam ter o nível de usabilidade requerido para a versão *mobile*.

A barra de pesquisa simples é constituída apenas por 4 caixas de texto e quatro botões. Em nenhum destes elementos foi necessária intervenção, uma vez que possuem dimensões razoáveis para ser pressionados confortavelmente num dispositivo móvel (as caixas de texto abrem o teclado virtual automaticamente) e se encontram já estilizados com o *design* da aplicação.

No caso da barra de pesquisa detalhada, a intervenção foi praticamente total. Apenas as caixas de texto se mantiveram intactas e todos os outros elementos foram tratados, nomeadamente as “combo-box”, os “date-picker” e as “checkbox”. Para exemplificar a forma como este tratamento se processa será utilizado como exemplo a “combo-box”.

Adaptação Mobile O processo de adaptação da aplicação ao ambiente *mobile*, parte do princípio que, ao ser desenvolvido código na camada de visualização, esse desenvolvimento é completamente independente do ambiente, e a adaptação que ocorre é completamente transparente para quem desenvolve. Passemos então ao exemplo da “combo-box” para a demonstração do conceito, utilizando para tal uma combo-box de escolha do género de um doente. A nível da camada de visualização é colocada apenas uma *div*, devidamente identificada como pode ser observado na figura 3.15.

```
<div style="height: 33%" id="text_genderContainer" />
```

Figura 3.15: Definição de uma “combo-box” ao nível da *view*

De seguida, ao nível do *script (javascript)* de configuração da página, é necessário fazer uma selecção desta *div* pelo seu identificador e atribuir-lhe o *plugin* “glinttCombo”, como é ilustrado na figura 3.16. No caso da “combo-box”, é necessário também posteriormente “perguntar” ao servidor quais as opções que deverão ser colocadas na mesma, desencadeando o processo descrito na secção de acesso a dados (3.4.4) e chamando a função “setDataSource” da “glinttCombo”. Para quem está a desenvolver a camada de visualização, termina aqui o processo de criação de uma “combo-box”, sendo o restante trabalho desempenhado pelo *plugin*.

```
$('#text_genderContainer').GlinttCombo({name:"#text_Gender"});  
$('#text_genderContainer').data('GlinttCombo').setDataSource(options);
```

Figura 3.16: Atribuição do *plugin* “glinttCombo” a um *div*

O *plugin* começa a sua ação nesta fase. Apenas a este nível é feita a distinção entre ambientes, com recurso à variável global definida no ponto de entrada da aplicação. No caso de o ambiente de execução ser *mobile* é então criado um elemento *HTML* “select”, com as devidas configurações e adicionado à *div*, caso contrário é adicionado o elemento normal do ambiente *desktop*. A criação do elemento da versão *mobile* pode ser observada na figura 3.17.

O resultado final para cada um dos ambientes é apresentado na figura 3.18.

3.4.6 Módulo de notificações

Este módulo é responsável pela notificação dos médicos da chegada de novos resultados que tenham subscrito. Numa fase inicial, quando estava planeado o desenvolvimento de uma aplicação

```

if (isMobile) {

    plugin.selectElement = $(document.createElement('select'));
    plugin.selectElement.attr("id", plugin.name);
    plugin.selectElement.attr("data-native-menu", "false");

    $element.append(plugin.selectElement).trigger('create');
}

```

Figura 3.17: Criação do elemento *HTML* respeitante ao ambiente *mobile*

nativa, este módulo foi idealizado com vista a utilização das *push notifications*. Com a alteração de planos e mudança para uma aplicação *web*, foi alterada essa premissa.

Assim, a estratégia a adoptar será uma estratégia síncrona e não assíncrona como seria com a tecnologia supra-referida. É executado um *script* periodicamente na base de dados (diariamente por exemplo), que irá ter o seguinte comportamento:

- Para cada médico, seleccionar os doentes que este tem subscritos;
- Para cada um desses doentes seleccionar os resultados que este tem associados;
- Para cada resultado, é feita a comparação do *timestamp* da sua inserção, com o *timestamp* da ultima execução do *script* sendo seleccionados os resultados com *timestamp* superior;
- No caso da subscrição ter alguma restrição ao nível do tipo de resultado, é feita a seleção apenas dos resultados requeridos;
- É feita a inserção dos resultados obtidos na tabela de notificações

Ao executar a aplicação, é feita uma contagem dos novos resultados, e apresentado um indicador sobre o menu das notificações, avisando o numero de novas notificações. O utilizador pode aceder ao respectivo menu e consultar os novos resultados. Ao consultar um dos resultados, as novas notificações são atualizadas, tanto a nível da aplicação, como a nível da base de dados.



Figura 3.18: Combo-box no ambiente *desktop* (à esquerda) e no ambiente *mobile* à (direita)

Capítulo 4

Operação e resultados

Neste quarto capítulo é apresentada a operação e resultados do sistema. Em primeiro lugar será feita uma explicação da operação do sistema e a forma como os utilizadores interagem com o mesmo. De seguida será feita uma reflexão sobre as situações em que o sistema pode ser utilizado e quais os benefícios que este apresenta. Por último serão apresentados os testes realizados ao sistema e o seu respetivo resultado.

4.1 Operação do sistema e papéis dos atores

A operação do sistema assenta sobre a informação apresentada no diagrama da figura 3.10, no capítulo de desenvolvimento, que explica a forma como os diferentes módulos e componentes do sistema interagem entre si. Uma vez que o sistema ainda se encontra em desenvolvimento, será apresentada a operação das funcionalidades do protótipo desenvolvido até ao momento.

Devido à extensão do projecto, a sua implementação foi dividida pelos diferentes módulos. A ordem escolhida para a implementação não foi deixada ao acaso, sendo que o módulo de documentos foi escolhido como o ponto de partida do desenvolvimento, uma vez que contém as funcionalidades consideradas básicas para um protótipo funcional do sistema. De momento, em conjunto com a autenticação na aplicação, este é o único módulo que se encontra já desenvolvido, cobrindo no entanto a grande maioria das mais importantes funcionalidades do sistema. Assim, a apresentação da operação do protótipo do sistema será constituída pela autenticação no mesmo e operação do módulo de documentos.

O sistema é operado apenas por um tipo de ator - o médico, tal como referido na secção 3.2.1, a apresentação do sistema será feita apenas da perspectiva desse mesmo ator, designado como **utilizador**. A descrição da operação do protótipo do sistema será feita em forma narrativa, articulando ações do utilizador com as atividades desempenhadas pelo sistema.

Embora o sistema seja funcional tanto em ambiente *desktop* como *mobile*, a operação apresentada, será apenas da perspectiva do ambiente *mobile*, uma vez que é este que se enquadra no âmbito desta dissertação.

Início do Sistema Para iniciar o sistema o utilizador deve começar por abrir um *web browser* no seu dispositivo, e aceder ao *url* interno da rede onde a aplicação é disponibilizada. Este *url* é diferente consoante a unidade hospitalar em que a aplicação está implementada, no entanto seguirá sempre a métrica “*http://localizacao_na_rede/Glintths.Shell*”.

4.1.1 Autenticação

Ao aceder ao *url* correto, será apresentado ao utilizador a página de autenticação, na qual o utilizador é solicitado a inserir o seu *username* e *password*. Caso alguma das credenciais seja inserida incorretamente, é mostrar uma mensagem de erro na própria página, caso contrário o utilizador é autenticado na aplicação e será aberta a página inicial da aplicação, que contém apenas uma mensagem de boas vindas e a barra de menu, que permite o acesso a qualquer um dos separadores da aplicação que disponibilizam os diferentes módulos da aplicação.

4.1.2 Documentos

O módulo de Documentos é iniciado quando o utilizador pressiona sobre o separador com o mesmo nome, no menu superior, quer seja a partir da página inicial, ou de qualquer outro separador.

Ao ser iniciado, é apresentada uma barra de pesquisa simples, ilustrada na figura 4.1. Esta barra possui quatro campos de texto e cinco botões, cujas funções são apresentadas na lista seguinte:



Figura 4.1: Barra de pesquisa simples do módulo de documentos e os seus elementos

- **C1** - Campo de texto para inserção do nome do doente;
- **C2** - Campo de texto para inserção do número de doente, um identificador interno da unidade hospitalar;
- **C3** - Campo de texto para inserção do número de processo do doente, outro dos identificadores internos da unidade hospitalar;
- **C4** - Campo de texto para inserção do número de SNS (Sistema Nacional de Saúde) do doente;

Operação e resultados

- **B1** - Botão “pesquisar”. Ao pressionar este botão, os dados na barra de pesquisa são utilizados pelo sistema para encontrar o doente;
- **B2** - Botão “apagar”. Ao pressionar este botão, os dados inseridos nos campos C1 - C4 são apagados;
- **B3** - Botão “filtros”. Este botão apenas simboliza se os filtros estão ou não ativos, não ocorrendo nenhuma ação caso este seja pressionado;
- **B4** - Botão “mais”. Ao pressionar este botão é mostrada a barra de pesquisa detalhada;
- **B5** - Botão “voltar”. Este botão só está visível depois de ser feita uma pesquisa. Ao pressionar neste botão é mostrada a vista anterior.

Para pesquisar um doente, o utilizador apenas necessita de preencher um ou mais dos campos apresentados e pressionar no botão de pesquisa. Caso os campos de pesquisa apresentados na barra de pesquisa simples não sejam suficientes, o utilizador pode pressionar no botão B4, sendo apresentada a barra de pesquisa detalhada. Esta barra pode ser observada na figura 4.2 e possui os seguintes campos:

A1

Figura 4.2: Barra de pesquisa detalhada do módulo de documentos e os seus elementos

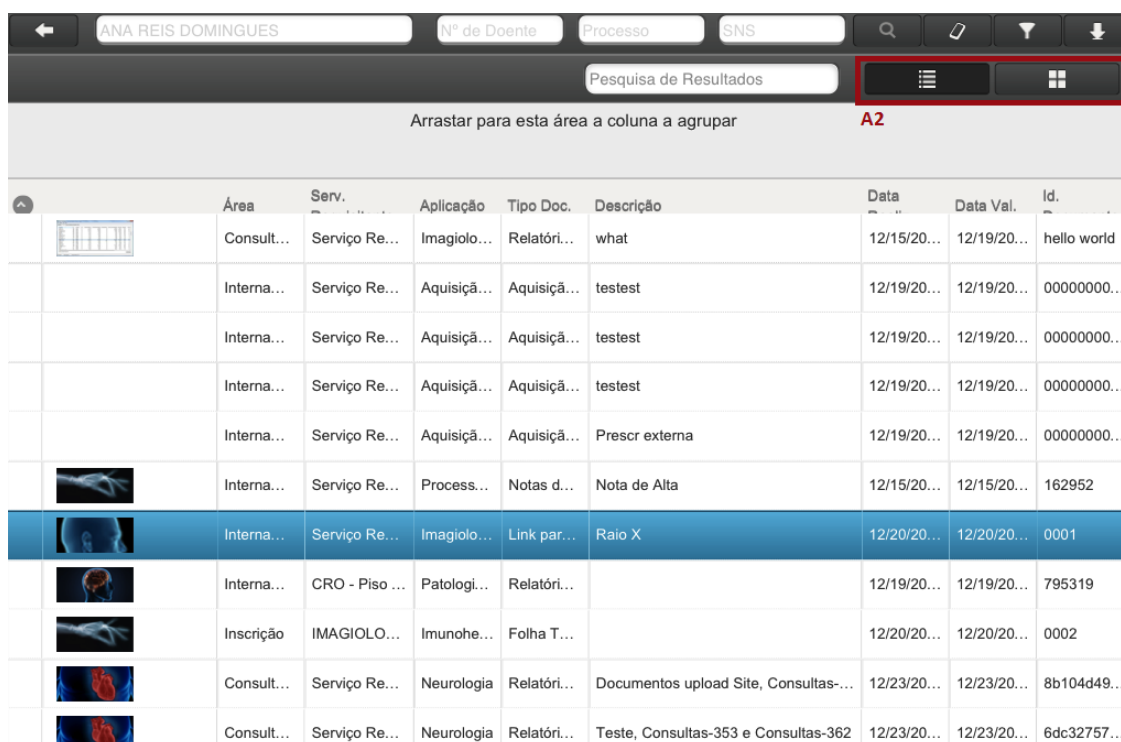
- **DP1** - Campo de data para inserção da data de nascimento do doente. A escolha é feita com recurso a um *datepicker*, com um formato comum em ambiente *mobile*;
- **CB1** - Campo de seleção para inserção do tipo de episódio do resultado. A escolha é feita com recurso a uma *combo-box*, com um formato comum em ambiente *mobile*;
- **CB2** - Campo de seleção para inserção do tipo aplicação do resultado. A escolha é feita com recurso a uma *combo-box*, com um formato comum em ambiente *mobile*;
- **CB3** - Campo de seleção para a inserção do género do doente. A escolha é feita com recurso a uma *combo-box*, com um formato comum em ambiente *mobile*;
- **C5** - Campo de texto para inserção do identificador do episódio;
- **CB4** - Campo de seleção para a inserção do tipo de documento. Os valores deste campo dependem do valor escolhido no campo CB2. A escolha é feita com recurso a uma *combo-box*, com um formato comum em ambiente *mobile*;

Operação e resultados

- **C6** - Campo de texto para inserção do identificador interno do documento;
- **C7** - Campo de texto para inserção do identificador externo do documento;
- **A1** - Área de inserção de um intervalo de datas. É escolhida uma ou mais das *checkboxes* CH1, CH2 ou CH3 que representam a data de execução, coiso1 e coiso2 respetivamente e utilizados os campos de data DP2 e DP3 para inserir o intervalo de datas.

Com a barra de pesquisa detalhada, o utilizador precisa apenas de preencher um ou mais campos do conjunto das duas barras e pressionar no botão de pesquisa para pesquisar um doente.

Depois de feita uma pesquisa, independentemente da utilização apenas de uma barra ou das duas, são apresentados os resultados da mesma. No caso de algum dado ter sido mal inserido ou do doente não ter nenhum resultado inserido na base de dados, é mostrada uma mensagem indicando que não foram encontrados resultados para a pesquisa feita. Caso contrário são apresentados os resultados, em formato lista, ou “thumbs”, podendo ser feita a escolha entre os formatos na zona A2, como é possível observar nas figuras 4.3 e 4.4, que mostram os dois formatos de apresentação dos resultados.



	Área	Serv.	Aplicação	Tipo Doc.	Descrição	Data	Data Val.	Id.
	Consult...	Serviço Re...	Imagiolo...	Relatóri...	what	12/15/20...	12/19/20...	hello world
	Interna...	Serviço Re...	Aquisiçã...	Aquisiçã...	testest	12/19/20...	12/19/20...	00000000...
	Interna...	Serviço Re...	Aquisiçã...	Aquisiçã...	testest	12/19/20...	12/19/20...	00000000...
	Interna...	Serviço Re...	Aquisiçã...	Aquisiçã...	testest	12/19/20...	12/19/20...	00000000...
	Interna...	Serviço Re...	Aquisiçã...	Aquisiçã...	Prescr externa	12/19/20...	12/19/20...	00000000...
	Interna...	Serviço Re...	Process...	Notas d...	Nota de Alta	12/15/20...	12/15/20...	162952
	Interna...	Serviço Re...	Imagiolo...	Link par...	Raio X	12/20/20...	12/20/20...	0001
	Interna...	CRO - Piso ...	Patologi...	Relatóri...		12/19/20...	12/19/20...	795319
	Inscrição	IMAGIOLO...	Imunohe...	Folha T...		12/20/20...	12/20/20...	0002
	Consult...	Serviço Re...	Neurologia	Relatóri...	Documentos upload Site, Consultas-...	12/23/20...	12/23/20...	8b104d49...
	Consult...	Serviço Re...	Neurologia	Relatóri...	Teste, Consultas-353 e Consultas-362	12/23/20...	12/23/20...	6dc32757...

Figura 4.3: Apresentação dos resultados em formato lista

O passo seguinte passa pela escolha de um resultado para ser visualizado. O utilizador escolhe um dos resultados na lista dos resultados do doente, abrindo assim a componente de visualização. A componente de visualização ocupa uma zona específica da página que lhe é concedida, independentemente do tipo de resultado que seja escolhido, fazendo com que visualização de qualquer tipo de resultado seja completamente embebida na página. Se a lista de resultados for apresentada

Operação e resultados

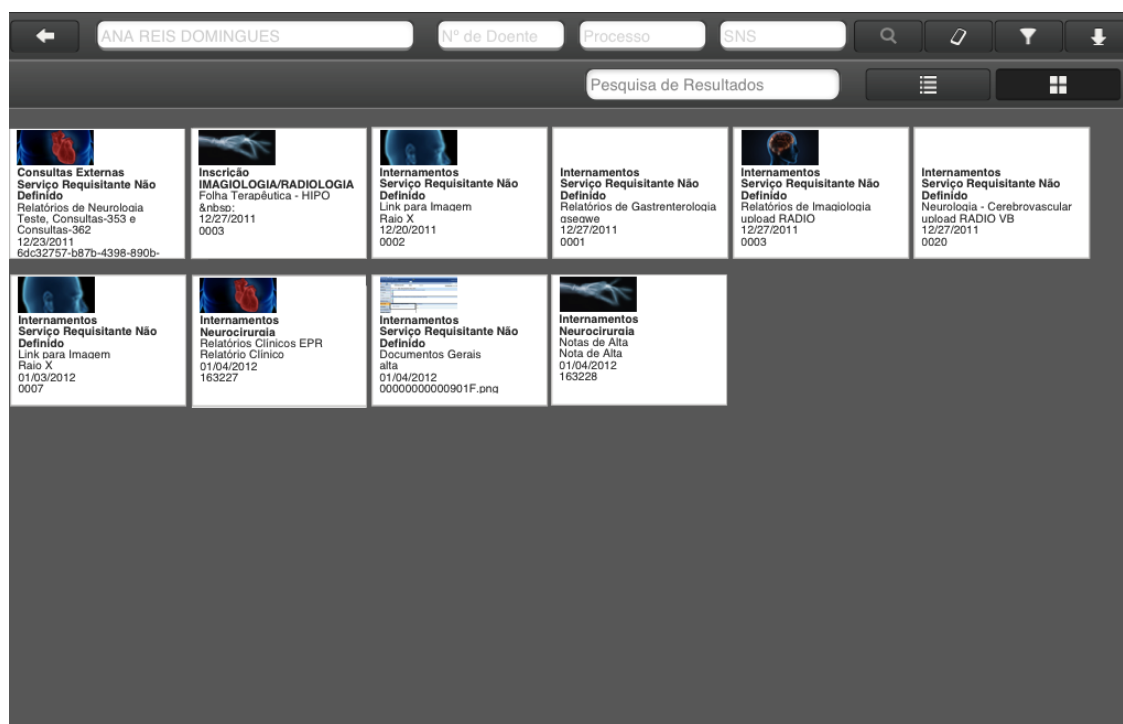


Figura 4.4: Apresentação dos resultados em formato thumbs

sob a forma de miniaturas, quando é feita a visualização de um resultado, as miniaturas estarão disponíveis para acesso rápido na parte esquerda do ecrã, bastando pressionar numa destas para abrir um novo resultado sem ter de abrir novamente o visualizador. No caso de a apresentação ser em formato lista, estas estarão na parte de trás da área de visualização, não permitindo o acesso rápido. Em qualquer um dos cenários é sempre possível pressionar sobre o botão “retrocede” na barra de pesquisa e voltar a trás, quer seja para voltar a ver os resultados da pesquisa ou para fazer uma pesquisa nova.

A visualização do resultado depende do seu tipo. São agora apresentados os três tipos de resultados suportados atualmente pela aplicação e a sua operação.

Pdf Ao abrir um resultado do tipo *pdf*, é executado o *plugin pdfviewer*. O resultado abre com a primeira página do *pdf* visível. O utilizador pode fazer *scroll* (arrastando a página para baixo com o dedo) para avançar para a página seguinte, ou ver a parte inferior da página, se esta não estiver visível inicialmente. Na figura 4.5 podemos observar como é feita a representação do *pdf*, e os elementos presentes na barra superior, que permitem manipular o pdf. Estes elementos são:

- **B6** - Botão “anterior”. Ao pressionar este botão, é mostrada a página anterior do documento, no topo da mesma. Só é possível pressionar este botão se a página atual não for a primeira;
- **B7** - Botão “seguinte”. Ao pressionar este botão, é mostrada a página seguinte do documento, no topo da mesma. Só é possível pressionar este botão se a página atual não for a última;

Operação e resultados

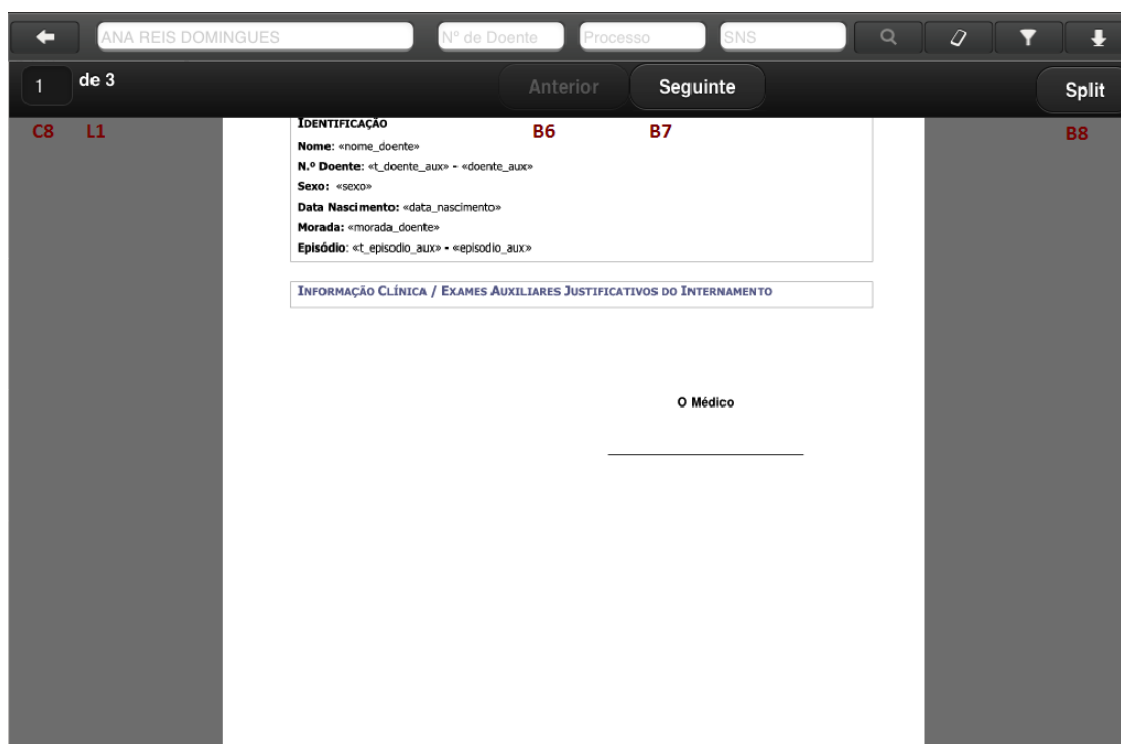


Figura 4.5: Apresentação do visualizador de *pdfs* e os seus respetivos elementos

- **C8** - Campo de texto indicador da página atual. Este campo é editável e ao inserir um número de página dentro dos limites do documento, é mostrada a página inserida, no topo da mesma. Caso seja inserido um caracter não numérico, ou um número fora dos limites do documento, é mostrada uma mensagem de aviso (informando que o número inserido é incorreto) e o documento mantém-se na posição atual.
- **L1** - Etiqueta que representa o número total de páginas do documento
- **B8** - Botão “split”. Ao pressionar este botão, passam a ser mostradas duas páginas de cada vez, uma ao lado da outra, estando as páginas ímpar do lado esquerdo e as páginas par do lado direito. Carregando novamente no mesmo botão faz com que se volte para o modo de visualização normal. Apenas é possível pressionar este botão para documentos com mais do que uma página.

Imagem Ao abrir um resultado do tipo imagem, é executado o *plugin imgViewer*. Um resultado do tipo imagem pode ser constituído por uma ou várias imagens, sendo que o resultado é sempre aberto na primeira destas. Nesta fase é apenas permitido ainda a visualização da imagem e não a sua manipulação pelo que a única ação disponível ao utilizador é a alteração da imagem atual para outra imagem do resultado. Os elementos da página são apresentados na figura 4.6 e as suas respetivas funções são:

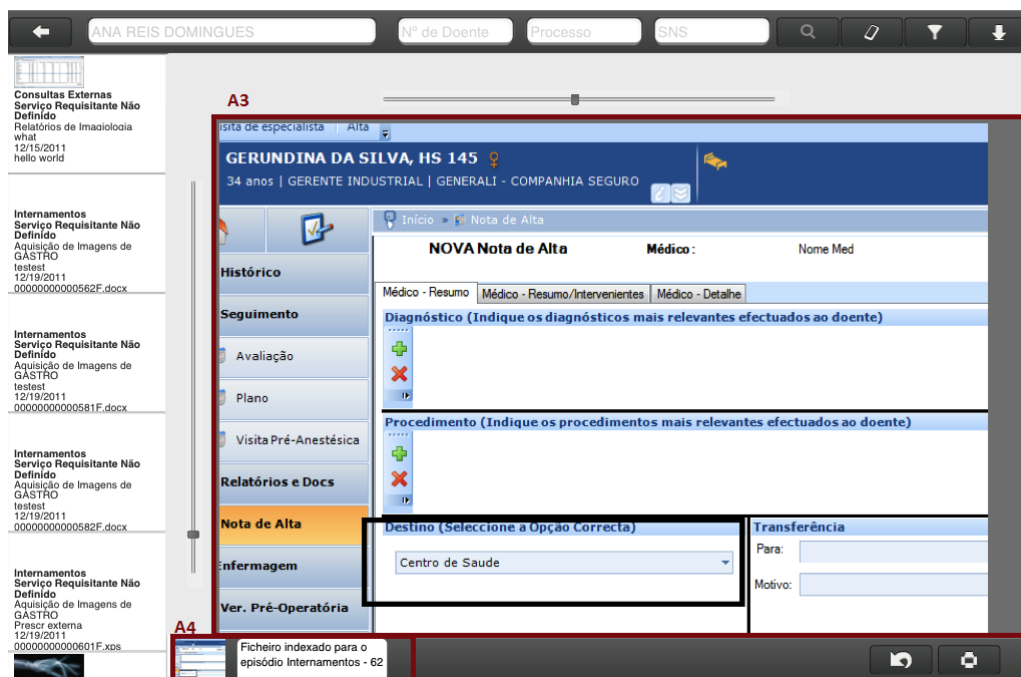


Figura 4.6: Apresentação do visualizador de imagens e os seus respetivos elementos

- **A3** - Área de visualização. Nesta área é possível observar a imagem atual do resultado escolhido;
- **A4** - Área de escolha da imagem. Caso o resultado seja constituído por mais do que uma imagem, nesta área é possível escolher qual das imagens é apresentada na área A3.

Vídeo Ao abrir um resultado do tipo vídeo, é executado o *plugin videoViewer*. Assim que o visualizador é aberto, o vídeo começa a ser carregado e pode ser reproduzido de imediato até ao local onde foi carregado. Uma vez que o *plugin* utiliza um reprodutor de *HTML5*, este permite que cada dispositivo reproduza o vídeo com o seu próprio reprodutor (estando no entanto completamente embebido no sistema). A figura 4.7 ilustra o funcionamento do visualizador num *iPad* e os seus respetivos elementos que são responsáveis pela manipulação do vídeo são os seguintes:

- **A5** - Área de visualização. Nesta área é possível observar a reprodução do vídeo.
- **B9** - Botão de *play/pause*. Pressionando sobre este botão faz a reprodução do vídeo parar ou continuar, dependendo do estado anterior.
- **B10** - Botão de *fullscreen*. Pressionando sobre este botão faz com que a reprodução do vídeo ocupe toda a área disponível no ecrã. Pressionando num botão semelhante no modo de ecrã inteiro faz com que se volte ao modo inicial.
- **A6** - Área da barra de estado. Esta barra indica a posição temporal do vídeo, e caso seja arrastado permite alterar esse mesmo valor. Na mesma barra existe ainda a indicação de até que local foi feito o carregamento do vídeo (preenchimento com maior opacidade).

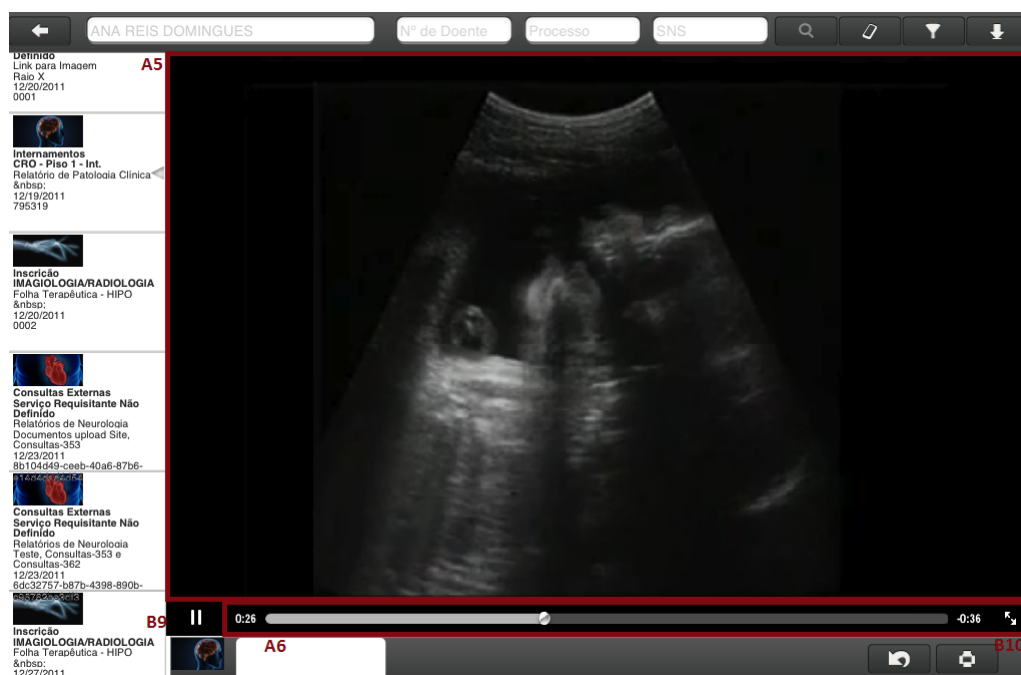


Figura 4.7: Apresentação do visualizador de imagens e os seus respetivos elementos

4.2 Utilização e benefícios

Uma vez que se encontra ainda em estado prototipal, a utilização do sistema cinge-se ao ambiente de testes interno da empresa. Apenas na sua versão final, poderá esta ser colocada em modo de produção e implementado em unidades hospitalares. Numa fase inicial será utilizada uma unidade hospitalar de pequena dimensão como implementação “piloto”, onde será implementado o sistema em conjunto com a atualização da versão *desktop*, para que sejam levados a cabo testes de utilização do sistema e de performance, garantindo que o sistema está de facto pronto para ser distribuído. Após completar esta fase, poderá então começar a fase de implementação do sistema, nas diversas unidades hospitalares onde a empresa marca presença.

Apesar de não ser ainda a versão final, o protótipo desenvolvido apresenta já um conjunto de benefícios visíveis. Em primeiro lugar temos o benefício inerente à disponibilização dos resultados nos dispositivos móveis que é a portabilidade. Os resultados de *MCDTs* deixam de estar presos ao ambiente *desktop* e aos computadores pessoais, e adquirem mobilidade, sem no entanto perder qualidade de visualização. Esta característica irá permitir que os médicos possam beneficiar da informação disponibilizada pelos resultados de *MCDTs* em qualquer parte da unidade hospitalar (com acesso à rede interna), nas inúmeras situações em que estes podem ser úteis, em vez de apenas no seu gabinete, no seu computador pessoal.

Os restantes benefícios ocorrem ao nível do desenvolvimento futuro, tanto da própria aplicação, como de qualquer aplicação móvel por parte da empresa.

A componente de visualização de multimédia foi desenvolvida sob o formato de *plugin*, sendo que este é responsável por delegar a reprodução dos diferentes formatos de multimédia para os

diferentes “*sub-plugins*”. Esta estratégia confere um grau de modularidade muito grande a esta componente, permitindo o suporte futuro de novos formatos de multimédia sem necessidade de alterar o código base do sistema, mas apenas acrescentar um novo plugin ou alterar um dos existentes. Esta componente é desenvolvida com recurso a tecnologias puramente web, nomeadamente *HTML*, *CSS* e *Javascript*, não estando por isso presa à atual tecnologia dos sistema - o *ASP.NET MVC*, e podendo ser integrada com qualquer outra tecnologia web.

O desenvolvimento ao nível da camada de visualização da aplicação, é feito de forma independente do ambiente em que este será apresentado. Para tal são utilizados controlos específicos, de que são exemplos o “*glinttCombo*” e o “*GlinttDatePicker*”, implementados sob a forma de extensões a elementos *HTML*. Estes controlos são responsáveis pela deteção do ambiente de execução e adequação dos elementos adicionados à página a esse mesmo ambiente, tornando esta tarefa perfeitamente transparente para quem está a fazer um desenvolvimento tanto neste sistema, como em qualquer outra aplicação futura da empresa com componente móvel.

4.3 Testes e resultados de funcionamento

São agora apresentados os testes realizados ao protótipo do sistema, baseados nos casos de utilização definidos na secção 3.2.3 do capítulo do desenvolvimento. Da mesma forma que os casos de uso possuem diferentes graus de prioridade e importância, essa categorização também está presente nos testes de aceitação. Assim, dos testes realizados, os que foram executados com sucesso foram os de maior prioridade, que são os relativos às funcionalidades do protótipo de sistema realizado, ou seja, essencialmente sobre o módulo de documentos.

Cada teste apresentado possui uma ligação para a sua respetiva tabela, com informação mais detalhada, presente no anexo C, onde pode ser consultada a lista completa dos testes de aceitação do sistema.

4.3.1 Testes executados com sucesso

4.3.1.1 Módulo de Autenticação

TA01 e TA02 - Login Este conjunto de testes tem como intuito verificar se a função de *login* se encontra a funcionar correctamente. Com as credenciais de autenticação de um utilizador verifica-se que é possível aceder ao sistema e se o sistema bloqueia o acesso em caso de erro em alguma das credenciais. Este teste foi executado com sucesso. Tabelas: C.1 e C.2.

TA03 - Logout Neste teste é verificado se é possível sair da aplicação, sendo feita a “desautenticação” do utilizador e apresentada a página de login. Este teste foi executado com sucesso. Tabela: C.3.

4.3.1.2 Módulo de Documentos

TA04 e TA05 - Pesquisa Simples Este conjunto de testes tem como objetivo verificar se a pesquisa de doentes com recurso à barra de pesquisa simples funciona correctamente, ou seja, inserindo dados corretos é possível encontrar os resultados de um doente, e inserindo dados errados é apresentada uma mensagem informando que não foram encontrados resultados. Tabelas: [C.4](#) e [C.5](#).

TA06 e TA07 - Pesquisa Detalhada Este conjunto de testes tem como objetivo verificar se a pesquisa de doentes com recurso à barra de pesquisa detalhada funciona correctamente, ou seja, inserindo dados corretos é possível encontrar os resultados de um doente, e inserindo dados errados é apresentada uma mensagem informando que não foram encontrados resultados. Tabelas: [C.6](#) e [C.7](#).

TA09 - Visualiza PDF Este teste tem como objetivo verificar se é possível abrir um resultado do tipo *pdf*. Após serem encontrados os resultados de um doente, é escolhido um que seja do tipo *pdf* e verificado se é possível visualizar o mesmo. Tabela [C.9](#).

TA10 - Visualiza Imagem Este teste tem como objetivo verificar se é possível abrir um resultado do tipo imagem. Após serem encontrados os resultados de um doente, é escolhido um que seja do tipo imagem e verificado se é possível visualizar o mesmo. Tabela [C.10](#).

TA11 - Visualiza Vídeo Este teste tem como objetivo verificar se é possível abrir um resultado do tipo vídeo. Após serem encontrados os resultados de um doente, é escolhido um que seja do tipo vídeo e verificado se é possível reproduzir o mesmo. Tabela [C.11](#).

TA12 - Manipula PDF Este teste tem como objetivo verificar se é possível manipular um resultado do tipo *pdf*. Após ser aberto um resultado deste tipo, é verificado se é possível realizar operações como mudar de página, fazer *scroll* pelo documento e utilizar a função de “split”. Tabela [C.12](#).

TA13 - Manipula Imagem Este teste tem como objetivo verificar se é possível manipular um resultado do tipo imagem. Após ser aberto um resultado deste tipo, é verificado se é possível ampliar e rodar a imagem. Tabela [C.13](#).

4.3.2 Testes executados sem sucesso ou não executados

4.3.2.1 Módulo de Notificações

TA14 - Adiciona Pedido Notificação Este teste tem como objetivo verificar se é possível adicionar um pedido de notificação, ou subscrição aos resultados de um determinado doente, quando é feita a pesquisa do mesmo, no módulo de documentos ou de linha de histórico. Tabela: [C.14](#).

TA15 - Consulta notificações Este teste tem como objetivo verificar se é possível consultar as notificações de um utilizador, no caso de existirem resultados novos que este tenha subscrito. Tabela: [C.15](#).

TA16 - Consulta pedidos de notificação Este teste tem como objetivo verificar se é possível verificar quais são os pedidos de notificação de um utilizador que estão ativos, ou seja, dos quais espera receber novas notificações. Tabela: [C.16](#).

TA17 - Edita pedido de notificação Este teste tem como objetivo verificar se é possível editar um pedido de notificação que um utilizador tenha inserido. Tabela: [C.17](#).

TA18 - Elimina pedido de notificação Este teste tem como objetivo verificar se é possível eliminar um pedido de notificação, ou seja retirar a subscrição aos resultados de um doente. Tabela: [C.17](#).

Operação e resultados

Capítulo 5

Conclusões e Trabalho Futuro

Neste quinto e último capítulo são apresentadas as conclusões do trabalho realizado. É também feita uma análise à satisfação dos objetivos definidos na fase inicial do projeto e por último são feitas considerações acerca do trabalho futuro, que poderá ainda ser desenvolvido no âmbito do projeto.

5.1 Conclusões

A revisão bibliográfica permitiu concluir que embora atualmente existam várias alternativas no mercado que permitem o tratamento de resultados de *MCDTs*, nenhuma delas contempla a disponibilização destes em dispositivos móveis, como era também o caso do *eResults*. Na conjuntura atual, com o desenvolvimento que estas tecnologias têm sido alvo, esta era uma lacuna grave, que procurou ser resolvida. Ao ser feito o estudo das tecnologias a utilizar, foi concluído que a diversidade e heterogeneidade dos dispositivos móveis e seus sistemas operativos se apresenta como uma forte barreira à disponibilização homogênea de aplicações exigindo um grande custo de desenvolvimento, sendo que tem existido um grande esforço a nível empresarial e da comunidade para reduzir este impacto, com a disponibilização de ambientes de desenvolvimento multi-plataforma. Estes tipo de soluções tem proliferado e na maioria dos casos, procuram aproximar o desenvolvimento *web* do desenvolvimento *mobile*. O estudo feito a estes ambientes procurou abranger um número considerável de alternativas com abordagens diferentes e permitiu concluir que apesar da diversidade de alternativas existente, a grande maioria ainda está em fase embrionária, apresentando lacunas em alguns sectores que podem comprometer o desenvolvimento de aplicações com graus de complexidade elevados.

A implementação e resultados obtidos permitiram concluir que com a conjugação de tecnologias escolhida foi a mais indicada para o tipo de abordagem pretendido, ou seja, para a adaptação inteligente de uma aplicação *web* às necessidades específicas dos dispositivos móveis, permitindo uma redução significativa do esforço de desenvolvimento.

5.1.1 Satisfação dos Objetivos

Dos objetivos definidos no início do projeto, o único que não foi realizado foi o da criação de um módulo de notificações de entrada de resultados de *MCDTs* no sistema. Esta situação era relativamente previsível dada a extensão do projeto mas também porque a mutabilidade de que este foi alvo não permitiu a utilização da tecnologia prevista inicialmente (*push notifications*) o que dificultou a conjugação deste objectivo com os objetivos mais prioritários.

Os restantes objetivos foram cumpridos na totalidade. O protótipo desenvolvido permite a consulta de resultados de um determinado doente, a sua visualização independentemente do seu formato bem como a sua manipulação. O protótipo é funcional não só nas plataformas alvo - *Android* e *iOS*, mas também em todas as outras plataformas em que o *jQuery Mobile* é executado, nomeadamente *Blackberry*, *Windows Mobile*, entre outras.

Os objetivos não tangenciais, foram também cumpridos, uma vez que o protótipo desenvolvido não só possui uma interface intuitiva e apelativa, como mantém uma operação bastante semelhante ao seu antecessor no ambiente *desktop*, o que permitirá reduzir o risco de dificuldade de adaptação à mudança, pelos utilizadores finais.

5.2 Trabalho Futuro

O trabalho futuro passará pela conclusão dos objetivos definidos e não alcançados, bem como implementação de algumas funcionalidades adicionais, que foram propostas e debatidas no decurso do projeto. Este trabalho deverá seguir as seguintes linhas:

- Desenvolvimento do módulo de notificações através da abordagem definida no capítulo da implementação;
- Desenvolvimento do módulo de linha de histórico;
- Desenvolvimento do módulo de resultados analíticos, numa primeira fase para resultados analíticos simples e numa fase posterior para resultados analíticos evolutivos, possivelmente com a utilização da framework *infragistics* para a geração e manipulação de gráficos;
- Desenvolvimento dos módulos de configurações, nomeadamente os módulos utilizador e filtros;
- A utilização de uma *shell* que permita que a aplicação se comporte de forma nativa, permitindo um acesso mais fluido às capacidades dos dispositivos, nomeadamente para a utilização de *push notifications* e melhoramento da performance;
- Desenvolvimento de um módulo de captura de resultados, que permita aos utilizadores recorrer às capacidades dos dispositivos, nomeadamente a câmara fotográfica para introduzir resultados do tipo imagem;

Anexo A

Tabelas de casos de uso

Apresenta-se neste anexo, uma descrição mais detalhada dos casos de uso, em formato de tabular, com a mesma estrutura que seria utilizada num relatório de especificação de requisitos.

A.1 Módulo de Autenticação

Código	CU01
Nome	Login
Custo	2
Prioridade	Média
Descrição	O utilizador autentica-se no sistema
Pré-Condições	O utilizador necessita de ter uma conta criada no serviço eResults
Pós-Condições	O utilizador encontra-se autenticado na aplicação
Fluxo de acções	1. A aplicação é iniciada 2. É inserido o "Username" 3. É inserida a "Password" 4. É pressionado o botão "Entrar"
Dependências	-
Mokup	B.1.1
Testes	C.1 e C.2

Tabela A.1: CU01

Tabelas de casos de uso

Código	CU02
Nome	Logout
Custo	1
Prioridade	Média
Descrição	O utilizador faz desautentica-se do sistema
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	1. O utilizador deixa de estar autenticado na aplicação 2. É apresentado o menu de login
Fluxo de acções	1. É pressionado o botão “Sair” 2. É apresentado o menu de login
Dependências	A.1 (CU01)
Mokup	B.1.2
Testes	C.3

Tabela A.2: CU02

A.2 Módulo de pesquisa de pacientes

Código	CU03
Nome	Pesquisa Simples
Custo	3
Prioridade	Alta
Descrição	O utilizador faz uma pesquisa simples por um paciente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	São apresentados os resultados para a pesquisa feita
Fluxo de acções	1. É pressionado o menu “Documentos” 2. É preenchido um ou vários dos campos disponíveis (Nome, N° Doente, Processo ou SNS) 3. É pressionado o botão “Procurar” 4. São apresentados os resultados
Dependências	A.1 (CU01)
Mokup	B.2.1
Testes	C.4 e C.5

Tabela A.3: CU03

Tabelas de casos de uso

Código	CU04
Nome	Pesquisa detalhada
Custo	3
Prioridade	Alta
Descrição	O utilizador faz uma pesquisa detalhada, tendo ao seu dispor mais campos para poder filtrar os resultados que deseja
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	São apresentados os resultados para a pesquisa feita
Fluxo de acções	1. É pressionado o menu “Documentos” 2. É pressionado o botão "mostrar mais opções" 3. É preenchido um ou vários dos campos disponíveis (D.Nasc, T.Episódio, Aplicação, etc)
Dependências	A.1 (CU01)
Mokup	B.2.2
Testes	C.6 e C.7

Tabela A.4: CU04

Código	CU05
Nome	Pesquisa de Linha de historico
Custo	1
Prioridade	Baixa
Descrição	O utilizador faz uma pesquisa do histórico de resultados de um determinado paciente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	É apresentado o histórico de resultados de um determinado paciente
Fluxo de acções	1. É pressionado o menu “Histórico” (ou linha de histórico) 2. É preenchido um ou vários dos campos disponíveis (Nome, N° Doente, Processo ou SNS) 3. É pressionado o botão “Procurar” 4. É apresentado o histórico do paciente
Dependências	A.1 (CU01)
Mokup	B.2.3
Testes	TODO

Tabela A.5: CU05

A.3 Módulo de visualização de multimédia

Código	CU06
Nome	Visualiza Pdf
Custo	3
Prioridade	Alta
Descrição	O utilizador visualiza um documento em formato pdf, com uma interface semelhante ao "Acrobat Reader"
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe algum documento pdf na base de dados 3. O utilizador encontrou algum documento relativo a um paciente através de um dos tipos e pesquisa
Pós-Condições	O utilizador visualiza o documento
Fluxo de acções	1. É pressionado um dos resultados 2. O resultado é apresentado 3. Pressionando o botão "voltar"o utilizador poder voltar ao menu anterior
Dependências	A.1 (CU01) e A.3 (CU03) ou A.4 (CU04)
Mokup	B.3.1
Testes	TODO

Tabela A.6: CU06

Tabelas de casos de uso

Código	CU07
Nome	Visualiza Imagem
Custo	2
Prioridade	Alta
Descrição	O utilizador visualiza um documento em formato de imagem (jpg, png, etc)
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe algum documento em formato imagem na base de dados 3. O utilizador encontrou algum documento relativo a um paciente através de um dos tipos e pesquisa
Pós-Condições	O utilizador visualiza o documento
Fluxo de acções	1. É pressionado um dos resultados 2. O resultado é apresentado 3. Pressionando o botão "voltar" o utilizador poder voltar ao menu anterior
Dependências	A.1 (CU01) e A.3 (CU03) ou A.4 (CU04)
Mokup	B.3.2
Testes	TODO

Tabela A.7: CU07

Tabelas de casos de uso

Código	CU08
Nome	Visualiza Vídeo
Custo	3
Prioridade	Alta
Descrição	O utilizador visualiza um documento em formato de vídeo
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe algum documento em formato vídeo na base de dados 3. O utilizador encontrou algum documento relativo a um paciente através de um dos tipos e pesquisa
Pós-Condições	O utilizador visualiza o documento
Fluxo de acções	1. É pressionado um dos resultados 2. O resultado é apresentado 3. Pressionando o botão "voltar"o utilizador poder voltar ao menu anterior
Dependências	A.1 (CU01) e A.3 (CU03) ou A.4 (CU04)
Mokup	B.3.3
Testes	TODO

Tabela A.8: CU08

Tabelas de casos de uso

Código	CU09
Nome	Manipula Pdf
Custo	2
Prioridade	Média
Descrição	O utilizador consegue manipular um documento em formato pdf com uma interface e interação semelhante ao "Acrobat Reader"
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe algum documento em formato pdf na base de dados 3. O utilizador encontrou algum documento relativo a um paciente através de um dos tipos e pesquisa
Pós-Condições	O utilizador consegue manipular o documento
Fluxo de acções	1. É pressionado um dos resultados 2. O resultado é apresentado 3. O utilizador consegue manipular o documento, tendo uma interacção semelhante ao "Acrobat Reader" 4. Pressionando o botão "voltar"o utilizador poder voltar ao menu anterior
Dependências	A.1 (CU01), A.3 (CU03) ou A.4 (CU04) e A.6 (CU06)
Mokup	B.3.4
Testes	TODO

Tabela A.9: CU09

Tabelas de casos de uso

Código	CU10
Nome	Manipula Imagem
Custo	2
Prioridade	Média
Descrição	O utilizador consegue manipular um documento em formato de imagem
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe algum documento em formato imagem na base de dados 3. O utilizador encontrou algum documento relativo a um paciente através de um dos tipos e pesquisa
Pós-Condições	O utilizador consegue manipular o documento
Fluxo de acções	1. É pressionado um dos resultados 2. O resultado é apresentado 3. O utilizador consegue ampliar e rodar o documento, utilizando os gestos multitouch "standard" 4. Pressionando o botão "voltar"o utilizador poder voltar ao menu anterior
Dependências	A.1 (CU01), A.3 (CU03) ou A.4 (CU04) e A.7 (CU07)
Mokup	B.3.5
Testes	TODO

Tabela A.10: CU10

A.4 Módulo de notificações

Código	CU11
Nome	Adiciona pedido de notificação
Custo	3
Prioridade	Média
Descrição	O utilizador consegue adicionar um pedido de notificação, associado a um determinado paciente
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe pelo menos um paciente na base de dados
Pós-Condições	Um novo pedido de notificação é adicionada
Fluxo de acções	1. É feita uma pesquisa de um paciente 2. É pressionado o botão "subscrever" 3. É aberta automaticamente uma janela de configuração da notificação 4. É escolhido o tipo de resultado a que a notificação está associada (podem ser todas) e o tipo de notificação que se pretende receber 5. É pressionado o botão "Criar Pedido de Notificação" 6. É confirmada a criação do pedido
Dependências	A.1 (CU01) e A.3 (CU03) ou A.4 (CU04)
Mokup	B.4.1
Testes	TODO

Tabela A.11: CU11

?

Tabelas de casos de uso

Código	CU12
Nome	Consulta notificações
Custo	2
Prioridade	Média
Descrição	O utilizador consulta as suas novas notificações
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe alguma notificação nova
Pós-Condições	O utilizador consegue ver as suas notificações
Fluxo de acções	1. O utilizador verifica que tem novas notificações (marcador sobre o menu “notificações” activo) 2. É pressionado o menu “Notificações 3. É escolhida uma das novas notificações 4. É pressionado o botão "Ver Resultado" 5. É aberto o respectivo resultado
Dependências	A.1 (CU01) e A.11 (CU11)
Mokup	B.4.2
Testes	TODO

Tabela A.12: CU12

Tabelas de casos de uso

Código	CU13
Nome	Consulta pedidos de notificação
Custo	2
Prioridade	Baixa
Descrição	O utilizador consulta todos os pedidos de notificações pendentes
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe pelo menos um pedido de notificação pendente
Pós-Condições	O utilizador consegue ver todos os seus pedidos de notificações pendentes
Fluxo de acções	1. É pressionado o menu “Notificações” 2. É escolhida a opção “Consultar Pedidos” 3. Pode ser escolhido um dos pedidos e consultados os seus detalhes
Dependências	A.1 (CU01) e A.11 (CU11)
Mokup	B.4.3
Testes	TODO

Tabela A.13: CU13

Tabelas de casos de uso

Código	CU14
Nome	Edita pedidos de notificação
Custo	1
Prioridade	Baixa
Descrição	O utilizador altera um pedido de notificação pendente
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe pelo menos um pedido de notificação pendente
Pós-Condições	O utilizador altera um pedido de notificação
Fluxo de acções	1. É pressionado o menu “Notificações” 2. É escolhida a opção “Consultar Pedidos” 3. É escolhido um dos pedidos tocando sobre este ou sobre a opção “ver” 4. Todos os dados referentes ao pedido são mostrados e podem ser alterados excepto o doente a que está afeto. Nesse caso deve-se proceder em primeiro lugar à eliminação (A.15) e posteriormente a criação de um pedido novo (A.11) 5. É pressionado o botão "Guardar"
Dependências	A.1 (CU01) e A.11 (CU11)
Mokup	B.4.4
Testes	TODO

Tabela A.14: CU14

Tabelas de casos de uso

Código	CU15
Nome	Elimina pedidos de notificação
Custo	1
Prioridade	Baixa
Descrição	O utilizador elimina um pedido de notificação pendente
Pré-Condições	1. O utilizador está autenticado na aplicação 2. Existe pelo menos um pedido de notificação pendente
Pós-Condições	O utilizador elimina um pedido de notificação
Fluxo de acções	1. É pressionado o menu “Notificações” 2. É escolhida a opção “Consultar Pedidos” 3. É escolhido um dos pedidos tocando sobre este ou sobre a opção “ver” 4. É escolhida a opção "Eliminar"
Dependências	A.1 (CU01) e A.11 (CU11)
Mokup	B.4.4
Testes	TODO

Tabela A.15: CU15

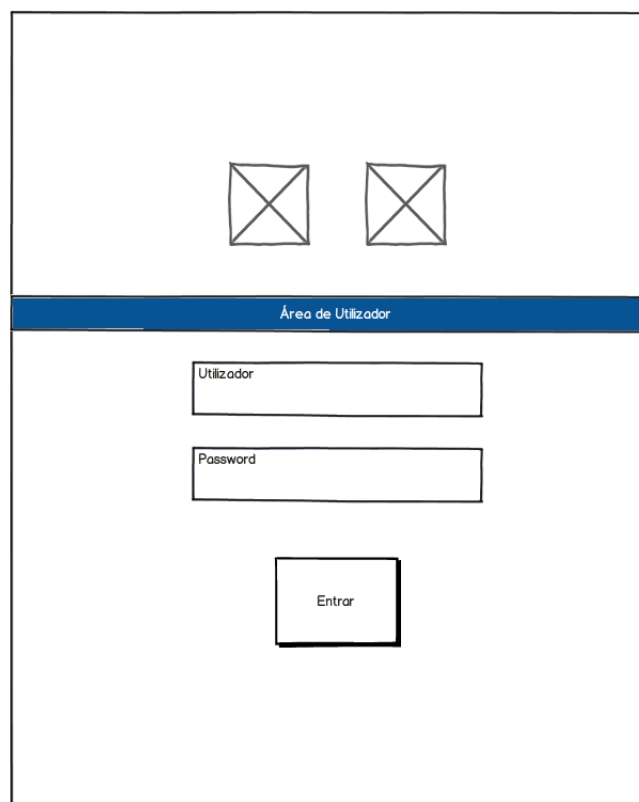
Anexo B

Mockups de Interface

Neste anexo são apresentados os mockups desenvolvidos na fase de especificação de requisitos da solução.

B.1 Módulo de Autenticação

B.1.1 Login



The mockup shows a login interface within a rectangular frame. At the top, there are two square icons, each containing an 'X'. Below these icons is a dark blue horizontal bar with the text 'Área de Utilizador' in white. Underneath the bar, there are two input fields: the first is labeled 'Utilizador' and the second is labeled 'Password'. Below the input fields is a rectangular button labeled 'Entrar'.

Figura B.1: Mockup do login, correspondente ao [A.1](#) (CU01)

B.1.2 Logout

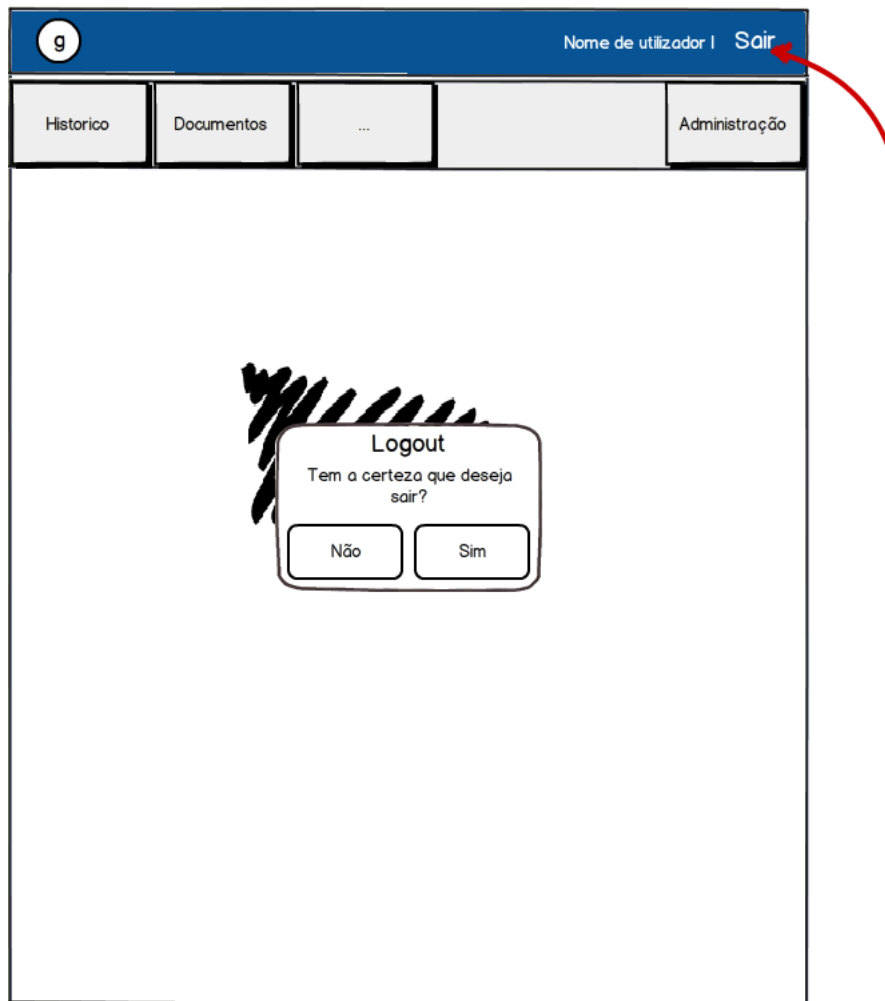
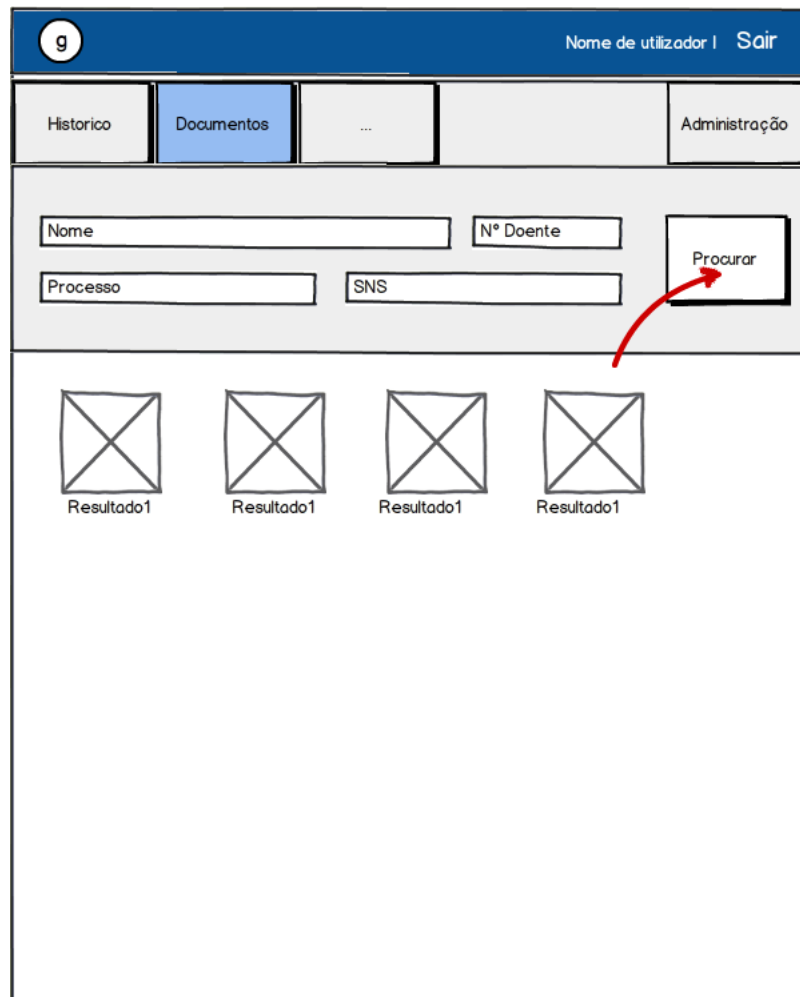


Figura B.2: Mockup do logout, correspondente ao [A.2](#) (CU02)

B.2 Módulo de pesquisa de pacientes

B.2.1 Pesquisa Simples



The mockup shows a web interface for patient search. At the top is a blue header bar with a circular icon containing the number '9' on the left and the text 'Nome de utilizador | Sair' on the right. Below the header is a navigation bar with four tabs: 'Historico', 'Documentos' (which is highlighted in blue), '...', and 'Administração'. The main content area has a light gray background. It contains four input fields arranged in two rows: 'Nome' and 'N° Doente' in the top row, and 'Processo' and 'SNS' in the bottom row. To the right of these fields is a button labeled 'Procurar'. A red curved arrow points from the 'Procurar' button down to the first of four placeholder boxes below. Each placeholder box consists of a square with an 'X' inside and the text 'Resultado1' centered underneath it.

Figura B.3: Mockup da pesquisa simples, correspondente ao [A.3](#) (CU03)

B.2.2 Pesquisa Detalhada

g Nome de utilizador | Sair

Historico Documentos ... Administração

Nome N° Doente
Processo SNS

Procurar

D.Nasc Episódio
T. Episódio T.Documento
Aplicação Id. Doc
Sexo Id. Ext

Pesquisa Por Data
Execução
Validação
✓ Episódio

Início Fim

Resultado1 Resultado2 Resultado3 Resultado4

Figura B.4: Mockup da pesquisa detalhada, correspondente ao [A.4](#) (CU04)

B.2.3 Pesquisa de linha de histórico

The mockup shows a web interface with a blue header bar containing a user icon 'g', the text 'Nome de utilizador |', and a 'Sair' button. Below the header is a navigation bar with buttons for 'Historico', 'Documentos' (highlighted in blue), '...', and 'Administração'. The main content area features a search form with four input fields: 'Nome', 'Nº Doente', 'Processo', and 'SNS'. A red arrow points to a 'Procurar' button. Below the search form is a large white box containing the text 'Resultado 1', 'Resultado 2', and 'Resultado 3'.

Figura B.5: Mockup da pesquisa de linha de histórico, correspondente ao [A.5](#) (CU05)

B.3 Módulo de visualização de multimédia

B.3.1 Visualiza Pdf

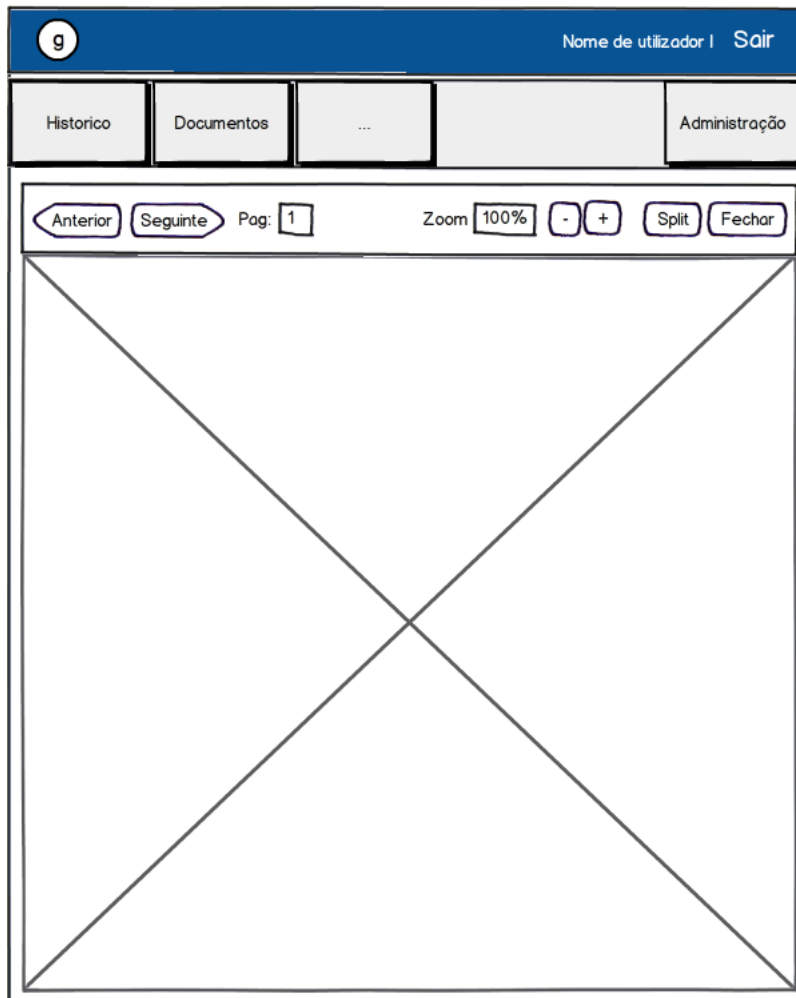


Figura B.6: Mockup da visualização de pdf, correspondente ao [A.6](#) (CU06)

B.3.2 Visualiza Pdf

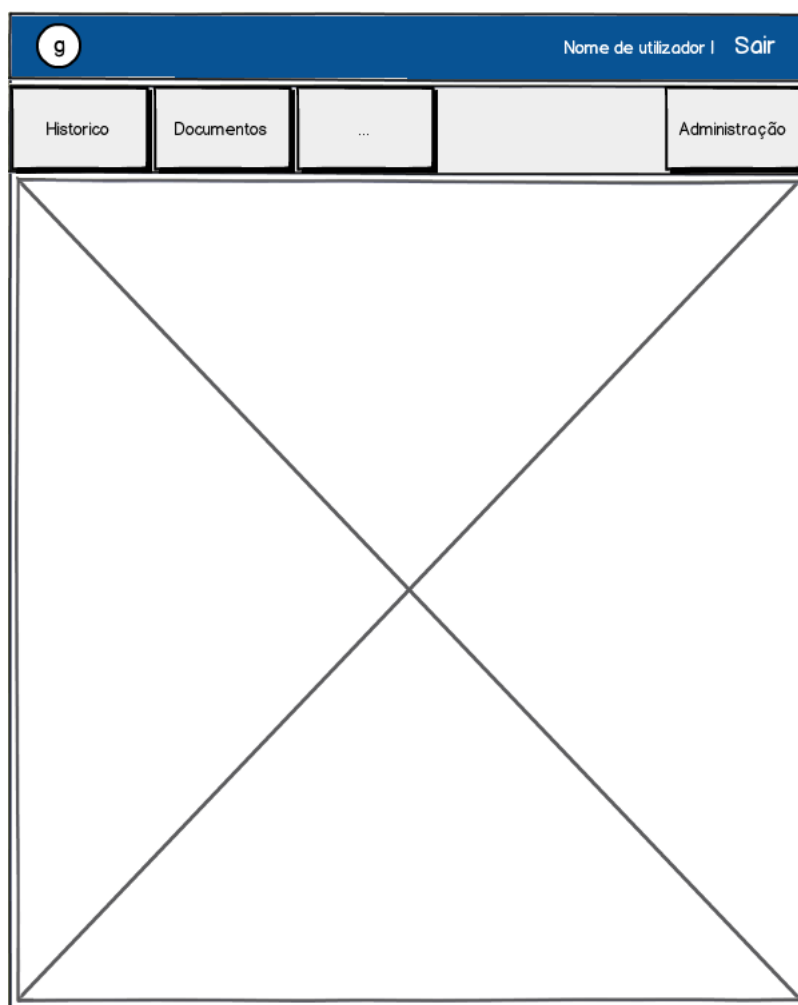


Figura B.7: Mockup da visualização de imagens, correspondente ao [A.7](#) (CU07)

B.3.3 Visualiza Vídeo

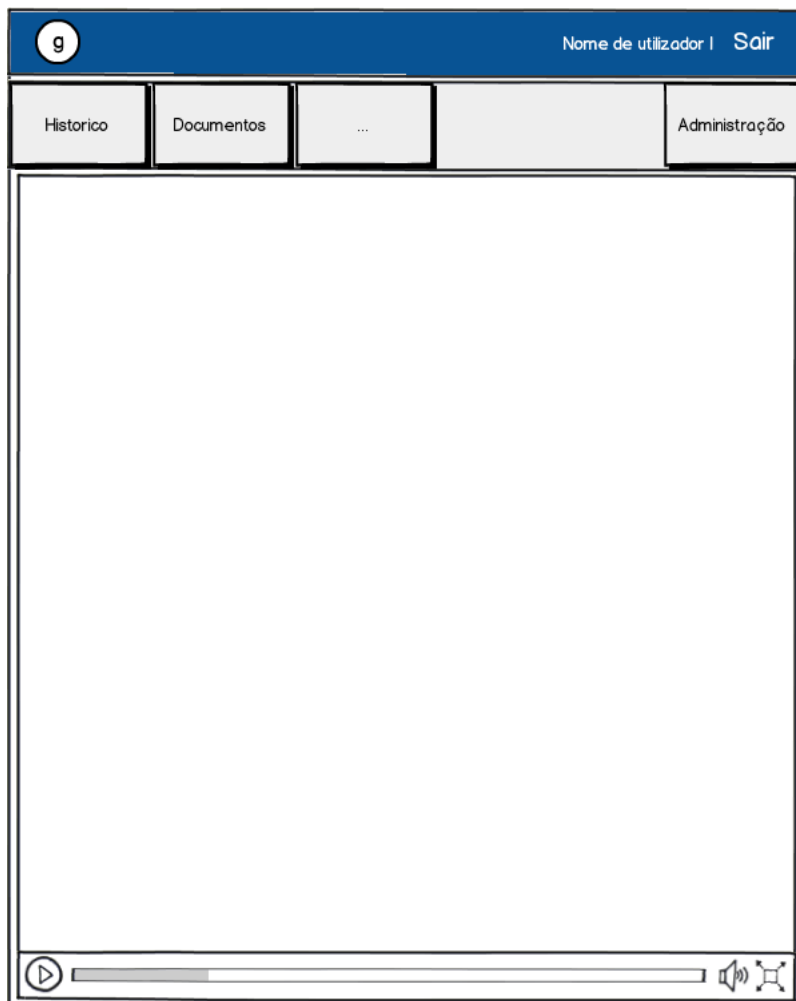


Figura B.8: Mockup da visualização de vídeos, correspondente ao [A.8](#) (CU08)

B.3.4 Manipula PDF

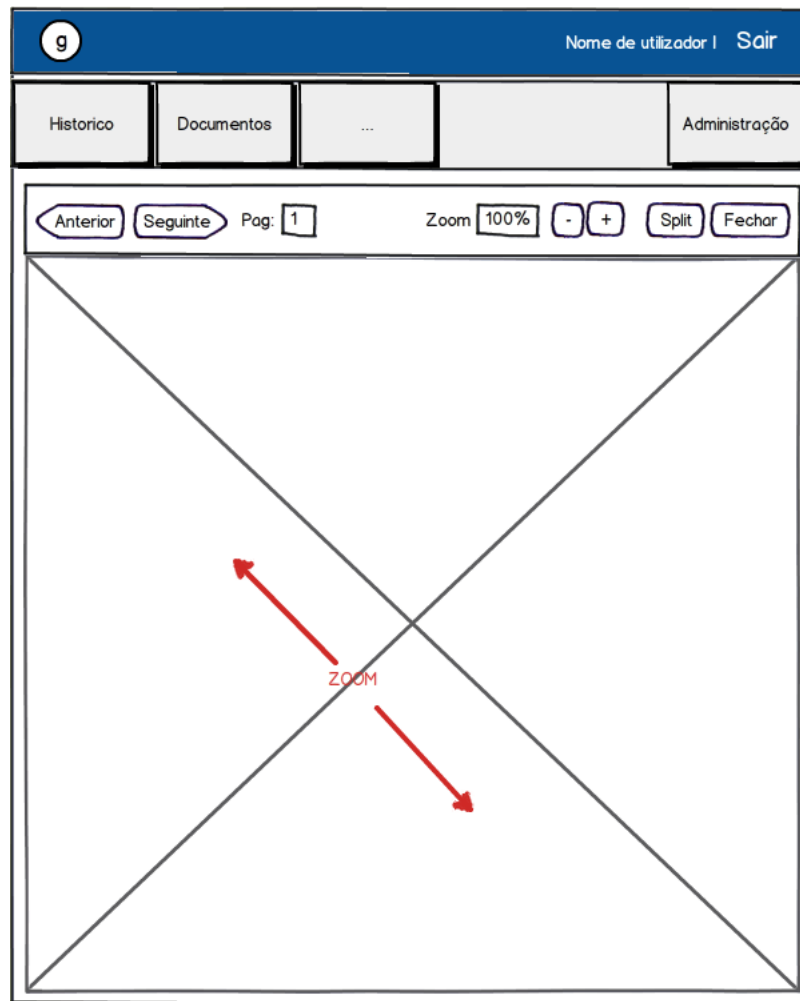


Figura B.9: Mockup da manipulação pdfs, correspondente ao [A.9](#) (CU09)

B.3.5 Manipula Imagens

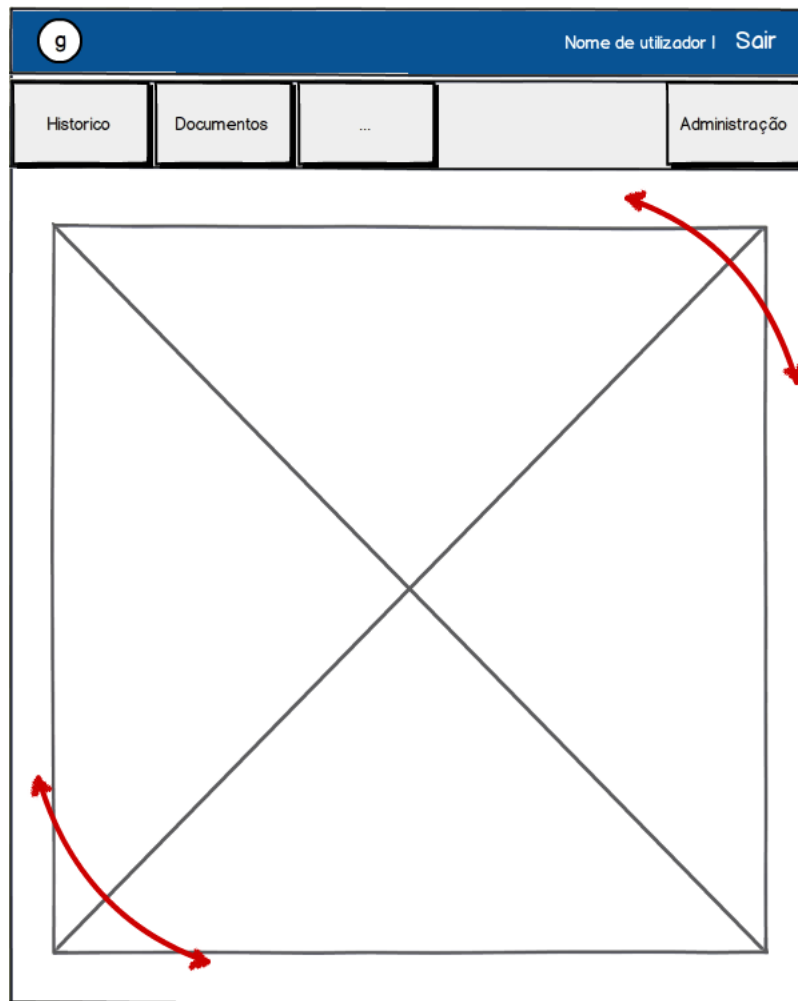


Figura B.10: Mockup da manipulação imagens, correspondente ao [A.10](#) (CU10)

B.4 Módulo de notificações

B.4.1 Adiciona pedido de notificação

9 Nome de utilizador | Sair

Historico Documentos ... Notificações Administração

Tipo de resultado

Todos os tipos ☒ ON

✓ Radiologia

✓ Imagiologia

✓ Outro1

✓ Outro2

✓ Outro3

...

Tipo de Notificação

SMS ☒ ON

EMAIL ☐ OFF

Criar Notificação

Confirmação

Tem a certeza que pretende criar esta notificação para o paciente "Nome do paciente"

Não Sim

Figura B.11: Mockup do menu de configurações de uma nova notificação, correspondente ao [A.11](#) (CU11)

B.4.2 Consulta notificações

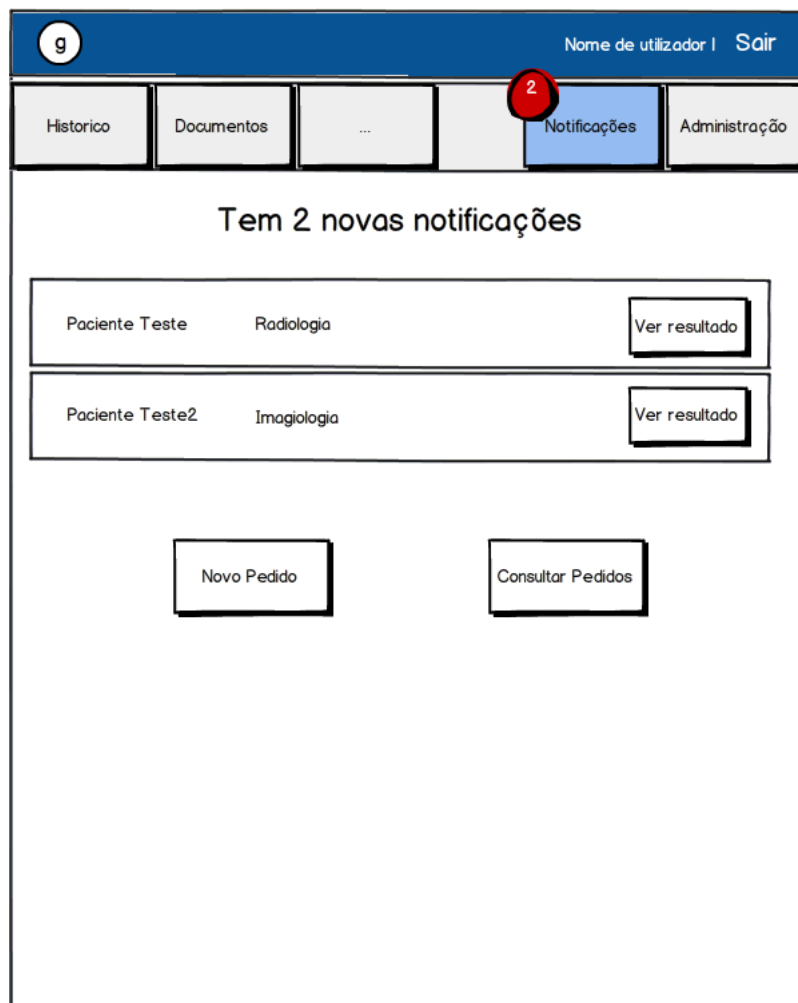


Figura B.12: Mockup da consulta de notificações, correspondente ao [A.12](#) (CU12)

B.4.3 Consulta pedidos de notificações

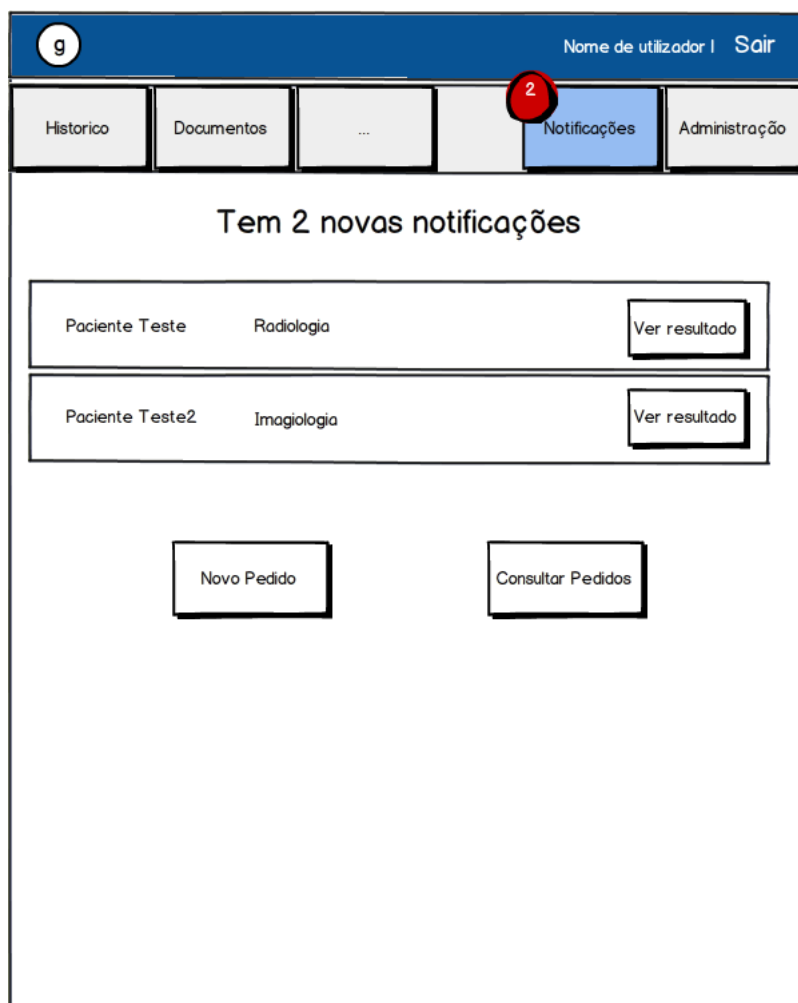
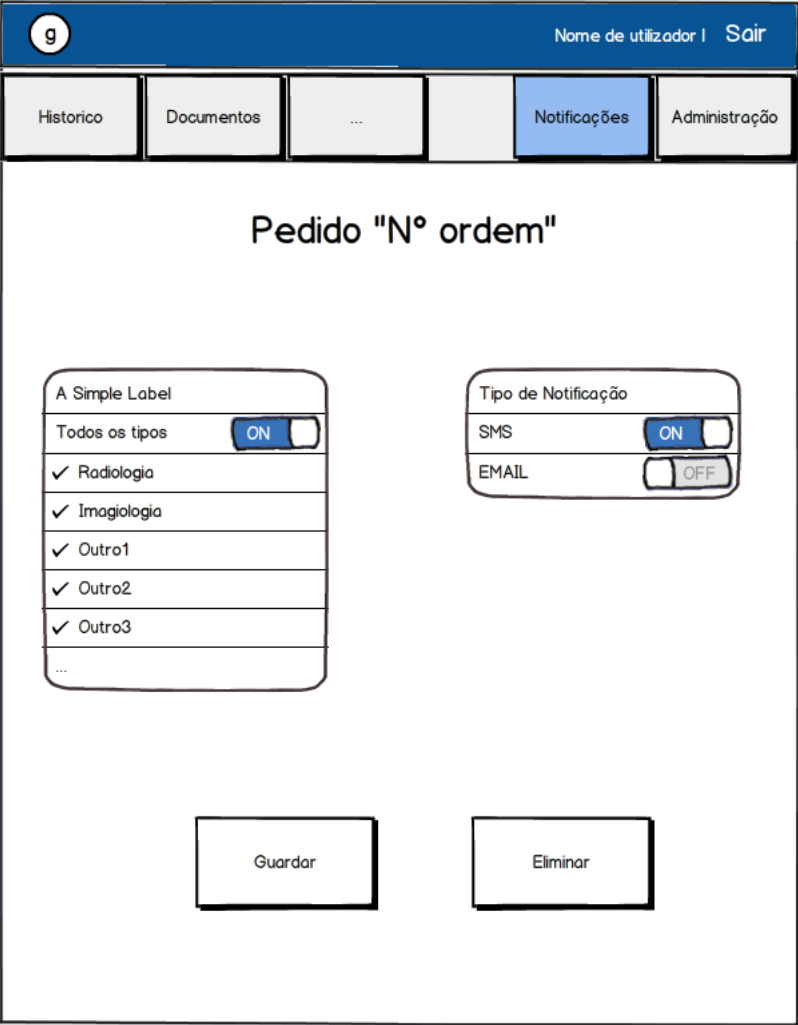


Figura B.13: Mockup da consulta de pedidos de notificações, correspondente ao [A.13](#) (CU13)

B.4.4 Edita pedidos de notificações



The mockup shows a web interface for editing pending notification orders. At the top is a blue header bar with a circular logo containing the letter 'g' on the left, and the text 'Nome de utilizador | Sair' on the right. Below the header is a horizontal navigation menu with five items: 'Historico', 'Documentos', '...', 'Notificações' (highlighted in blue), and 'Administração'. The main content area has a title 'Pedido "Nº ordem"' centered at the top. Below the title are two side-by-side panels. The left panel, titled 'A Simple Label', contains a toggle switch for 'Todos os tipos' (set to 'ON') and a list of categories with checkboxes: 'Radiologia', 'Imagiologia', 'Outro1', 'Outro2', 'Outro3', and an ellipsis. The right panel, titled 'Tipo de Notificação', contains two toggle switches: 'SMS' (set to 'ON') and 'EMAIL' (set to 'OFF'). At the bottom of the main area are two buttons: 'Guardar' and 'Eliminar'.

Figura B.14: Mockup da edição de pedidos notificações pendentes, correspondente ao [A.14](#) (CU14)

Anexo C

Tabelas de testes de aceitação

Apresenta-se neste anexo, uma descrição mais detalhada dos testes de aceitação, baseados nos casos de uso definidos no anexo [A](#), em formato de tabular, com a mesma estrutura que seria utilizada num relatório de especificação de requisitos.

C.1 Módulo de Autenticação

Código	TA01
Nome	Login correto
Condição específica a testar	Verificar se é possível autenticar correctamente um utilizador no sistema
Input	Username e Password
Pré-Condições	Existe pelo menos um utilizador no sistema
Pós-Condições	É feita a autenticação do utilizador
Procedimento	1. A aplicação é iniciada 2. É inserido o "Username" 3. É inserida a "Password" 4. É pressionado o botão "Entrar"
Dependências	-
Estado	Executado com sucesso
Caso de Uso	A.1

Tabela C.1: TA01

Tabelas de testes de aceitação

Código	TA02
Nome	Login errado
Condição específica a testar	Verificar que se alguma das credenciais de autenticação de um utilizador for mal inserida, não é possível autenticar-se na aplicação.
Input	Username e Password
Pré-Condições	-
Pós-Condições	O utilizador não se encontra autenticado
Procedimento	1. A aplicação é iniciada 2. É inserido o "Username" errado, ou é inserida a password errada ou ambos, ou não é inserida alguma das credenciais 3. É pressionado o botão "Entrar" 4. É apresentada uma mensagem de erro
Dependências	-
Estado	Executado com sucesso
Caso de Uso	A.1

Tabela C.2: TA02

Código	TA03
Nome	Logout
Condição específica a testar	Verificar que um utilizador autenticado na aplicação consegue "desautenticar-se"
Input	-
Pré-Condições	Um utilizador está autenticado na aplicação
Pós-Condições	O utilizador não se encontra autenticado e é apresentada a página de login
Procedimento	1. Está aberto qualquer módulo da aplicação, ou a página inicial 2. É pressionado no botão sair da barra superior 3. É apresentada uma mensagem de confirmação 4. É apresentada a página de login
Dependências	TA01
Estado	Executado com sucesso
Caso de Uso	A.2

Tabela C.3: TA03

C.2 Módulo de Documentos

Código	TA04
Nome	Pesquisa Simples correta
Condição específica a testar	Verificar que é possível encontrar um doente através da informação de um dos campos da pesquisa simples
Input	Dados do doente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	São apresentados os resultados do doente pesquisado
Procedimento	1. É aberto o módulo de documentos 2. É preenchido um dos campos da barra de pesquisa simples com dados de um doente (qualquer dos campos) 3. É pressionado o botão de pesquisa 4. São apresentados os resultados do doente
Dependências	TA01
Estado	Executado com sucesso
Caso de Uso	A.3

Tabela C.4: TA04

Tabelas de testes de aceitação

Código	TA05
Nome	Pesquisa Simples errada
Condição específica a testar	Verificar ao inserir um dado incorreto de um doente num dos campos da pesquisa simples não é encontrado nenhum resultado
Input	Dados do doente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	É apresentada uma mensagem indicando que não foram encontrados resultados
Procedimento	1. É aberto o módulo de documentos 2. É preenchido um dos campos da barra de pesquisa simples com dados errados, ou vários campos com dados errados ou não concordantes (de doentes diferentes) 3. É pressionado o botão de pesquisa 4. É apresentada uma mensagem indicando que não foram encontrados resultados
Dependências	TA01
Estado	Executado com sucesso
Caso de Uso	A.3

Tabela C.5: TA05

Tabelas de testes de aceitação

Código	TA06
Nome	Pesquisa Detalhada correta
Condição específica a testar	Verificar que é possível encontrar um doente através da informação de um dos campos da pesquisa detalhada
Input	Dados do doente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	São apresentados os resultados do doente pesquisado
Procedimento	1. É aberto o módulo de documentos 2. É pressionado o botão "mais" na barra de pesquisa simples para abrir a barra de pesquisa detalhada 3. É preenchido um dos campos da barra de pesquisa detalhada com dados de um doente (qualquer dos campos) 4. É pressionado o botão de pesquisa 5. São apresentados os resultados do doente
Dependências	TA01
Estado	Executado com sucesso
Caso de Uso	A.4

Tabela C.6: TA06

Tabelas de testes de aceitação

Código	TA07
Nome	Pesquisa Detalhada errada
Condição específica a testar	Verificar ao inserir um dado incorreto de um doente num dos campos da pesquisa detalhada não é encontrado nenhum resultado
Input	Dados do doente
Pré-Condições	O utilizador está autenticado na aplicação
Pós-Condições	É apresentada uma mensagem indicando que não foram encontrados resultados
Procedimento	1. É aberto o módulo de documentos 2. É pressionado o botão "mais" na barra de pesquisa simples para abrir a barra de pesquisa detalhada 3. É preenchido um dos campos da barra de pesquisa detalhada com dados errados, ou vários campos com dados errados ou não concordantes (de doentes diferentes) 4. É pressionado o botão de pesquisa 5. É apresentada uma mensagem indicando que não foram encontrados resultados
Dependências	TA01
Estado	Executado com sucesso
Caso de Uso	A.4

Tabela C.7: TA07

Tabelas de testes de aceitação

Código	TA08
Nome	Linha de histórico
Condição específica a testar	Verificar que uma pesquisa de linha de histórico apresenta os resultados correctamente
Input	-
Pré-Condições	O utilizador está autenticado na aplicação e é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa
Pós-Condições	É apresentada uma mensagem indicando que não foram encontrados resultados
Procedimento	1. É aberto o módulo de linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados por ordem cronológica
Dependências	TA01, TA04 ou TA06
Estado	Executado com sucesso
Caso de Uso	A.5

Tabela C.8: TA08

Código	TA09
Nome	Visualiza PDF
Condição específica a testar	Verificar se a aplicação consegue abrir um resultado do tipo pdf
Input	-
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa e existe pelo menos um resultado do tipo pdf na base de dados
Pós-Condições	É possível visualizar o pdf na área de visualização de resultados
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É escolhido um dos resultados, do tipo pdf 5. É aberto o pdf na área de visualização, sendo possível visualizar o mesmo
Dependências	TA01, TA04 ou TA06
Estado	Executado com sucesso
Caso de Uso	A.6

Tabela C.9: TA09

Tabelas de testes de aceitação

Código	TA10
Nome	Visualiza Imagem
Condição específica a testar	Verificar se a aplicação consegue abrir um resultado do tipo imagem
Input	-
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa e existe pelo menos um resultado do tipo imagem na base de dados
Pós-Condições	É possível visualizar a imagem na área de visualização de resultados
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É escolhido um dos resultados, do tipo imagem 5. É aberta a imagem na área de visualização, sendo possível visualizar a mesma
Dependências	TA01, TA04 ou TA06
Estado	Executado com sucesso
Caso de Uso	A.7

Tabela C.10: TA10

Tabelas de testes de aceitação

Código	TA11
Nome	Visualiza Vídeo
Condição específica a testar	Verificar se a aplicação consegue abrir um resultado do tipo vídeo
Input	-
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa e existe pelo menos um resultado do tipo vídeo na base de dados
Pós-Condições	É possível reproduzir o vídeo na área de visualização de resultados
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É escolhido um dos resultados, do tipo vídeo 5. É aberto o vídeo na área de visualização, sendo possível fazer a reprodução do mesmo
Dependências	TA01, TA04 ou TA06
Estado	Executado com sucesso
Caso de Uso	A.8

Tabela C.11: TA11

Tabelas de testes de aceitação

Código	TA12
Nome	Manipula PDF
Condição específica a testar	Verificar se é possível manipular um resultado do tipo PDF
Input	-
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa e existe pelo menos um resultado do tipo pdf na base de dados
Pós-Condições	É possível manipular um pdf na area de visualização de resultados, alterando a página actual através botões, scroll no documento, e utilizando a função de split
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É escolhido um dos resultados, do tipo pdf 5. É aberto o pdf na área de visualização, sendo possível visualizar o mesmo 6. É possível manipular o mesmo pdf
Dependências	TA01, TA04 ou TA06 e TA09
Estado	Executado com sucesso
Caso de Uso	A.8

Tabela C.12: TA12

Tabelas de testes de aceitação

Código	TA10
Nome	Manipula Imagem
Condição específica a testar	Verificar se a aplicação consegue manipular um resultado do tipo imagem
Input	-
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa e existe pelo menos um resultado do tipo imagem na base de dados
Pós-Condições	É possível manipular a imagem na área de visualização de resultados, aumentando e rodando a mesma
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É escolhido um dos resultados, do tipo imagem 5. É aberta a imagem na área de visualização, sendo possível visualizar a mesma 6. É possível manipular a mesma imagem, ampliando-a e rodando-a
Dependências	TA01, TA04 ou TA06 e TA10
Estado	Executado com sucesso
Caso de Uso	A.7

Tabela C.13: TA13

C.3 Módulo de notificações

Código	TA14
Nome	Adiciona pedido de notificação
Condição específica a testar	Verificar se o utilizador consegue adicionar um pedido de notificação
Input	Os dados do doente e configurações da notificação
Pré-Condições	O utilizador está autenticado na aplicação, é feita uma pesquisa com auxílio de uma ou das duas barras de pesquisa
Pós-Condições	É inserido um pedido de notificação, ou seja, é feita a subscrição dos resultados de um doente
Procedimento	1. É aberto o módulo de documentos ou linha de histórico 2. É feita uma pesquisa correcta com o auxílio de uma ou das duas barras de pesquisa 3. São apresentados os resultados do doente 4. É pressionado o botão subscrever 5. São adicionadas as configurações desejadas à subscrição, nomeadamente o tipo de notificação e o tipo de resultados 6. É pressionado o botão adicionar pedido 7. É adicionado um pedido de notificação
Dependências	TA01, TA04 ou TA06
Estado	Executado sem sucesso
Caso de Uso	A.11

Tabela C.14: TA14

Tabelas de testes de aceitação

Código	TA15
Nome	Consulta notificações
Condição específica a testar	Verificar se o utilizador consegue consultar as suas novas notificações
Input	Os dados do doente e configurações da notificação
Pré-Condições	O utilizador está autenticado na aplicação e existe um resultado novo de um dos pedidos de notificação do utilizador
Pós-Condições	É possível consultar as novas notificações do utilizador
Procedimento	1. É inserido um pedido de notificação para um doente 2. É inserido um resultado para esse mesmo doente 3. É verificado o sinal sobre o menu de notificações avisando que existe uma nova notificação 4. É pressionado o menu notificações 5. É verificado que existe uma nova entrada 6. É pressionado sobre essa entrada e verificado que é possível visualizar o resultado 7. É repetido o processo para um numero maior de resultados e para pedidos com restrições de resultados e outros tipos de notificações
Dependências	TA01, TA04 ou TA06 e TA14
Estado	Executado sem sucesso
Caso de Uso	A.12

Tabela C.15: TA15

Tabelas de testes de aceitação

Código	TA16
Nome	Consulta pedidos de notificações
Condição específica a testar	Verificar se o utilizador consegue consultar os seus pedidos de notificações
Input	Os dados do doente e configurações da notificação
Pré-Condições	O utilizador está autenticado na aplicação e existe pelo menos um pedido de notificação desse utilizador
Pós-Condições	É possível consultar os pedidos de notificações do utilizador
Procedimento	1. É inserido um pedido de notificação para um doente 2. É pressionado o menu notificações 3. É pressionado o botão "consultar pedidos" 4. É verificado que é possível consultar o pedido de notificação inserido
Dependências	TA01, TA04 ou TA06 e TA14
Estado	Executado sem sucesso
Caso de Uso	A.13

Tabela C.16: TA16

Tabelas de testes de aceitação

Código	TA17
Nome	Edita pedidos de notificações
Condição específica a testar	Verificar se o utilizador consegue editar os seus pedidos de notificações
Input	Os dados do doente e configurações da notificação
Pré-Condições	O utilizador está autenticado na aplicação e existe pelo menos um pedido de notificação desse utilizador
Pós-Condições	É possível editar os pedidos de notificações do utilizador
Procedimento	1. É inserido um pedido de notificação para um doente com determinadas configurações 2. É pressionado o menu notificações 3. É pressionado o botão "consultar pedidos" 4. É escolhido o pedido de notificação inserido anteriormente e pressionado o botão "editar" 5. É alterada uma ou mais configurações da notificação e pressionado o botão "alterar notificação" 6. É escolhido novamente o mesmo pedido de notificação 7. É verificado que as alterações feitas foram efectuadas
Dependências	TA01, TA04 ou TA06, TA14 e TA16
Estado	Executado sem sucesso
Caso de Uso	A.14

Tabela C.17: TA17

Tabelas de testes de aceitação

Código	TA18
Nome	Elimina pedidos de notificações
Condição específica a testar	Verificar se o utilizador consegue eliminar os seus pedidos de notificações
Input	Os dados do doente e configurações da notificação
Pré-Condições	eO utilizador está autenticado na aplicação existe pelo menos um pedido de notificação desse utilizador
Pós-Condições	É possível eliminar os pedidos de notificações do utilizador
Procedimento	1. É inserido um pedido de notificação para um doente com determinadas configurações 2. É pressionado o menu notificações 3. É pressionado o botão "consultar pedidos" 4. É escolhido o pedido de notificação inserido anteriormente e pressionado o botão "eliminar" 5. É verificado que o pedido de notificação foi eliminado
Dependências	TA01, TA04 ou TA06, TA14 e TA16
Estado	Executado sem sucesso
Caso de Uso	A.15

Tabela C.18: TA18

Referências

- [AGL10] Sarah Allen, Vidal Graupera e Lee Lundrigan. *Pro Smartphone Cross-Platform: iPhone, BlackBerry, Windows Mobile and Android Development and Distribution*. Apress, 2010.
- [Apl11] Confidentia Tecnologias Informáticas Aplicadas. Software appolo, 2011. Disponível em: <http://www.confidentia.pt/pt/software-appolo>, consultado no mês de Dezembro de 2011.
- [Cam08] Joel Campos. Módulo para gestão documental na solução eresults da cpc-hs. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2008.
- [Con04] World Wide Web Consortium. Web services architecture, 2004. Disponível em: <http://www.w3.org/TR/ws-arch/>, consultado no mês de Janeiro de 2012.
- [Con07a] World Wide Web Consortium. Soap version 1.2 part 1: Messaging framework, 2007. Disponível em: <http://www.w3.org/TR/2007/REC-soap12-part1-20070427/#extensibility>, consultado no mês de Janeiro de 2012.
- [Con07b] World Wide Web Consortium. Soap version 1.2 part 2: Adjuncts, 2007. Disponível em: <http://www.w3.org/TR/2007/REC-soap12-part2-20070427/#soapforrpc>, consultado no mês de Janeiro de 2012.
- [Cor11] Oracle Corporation. The java ee 6 tutorial, Junho 2011.
- [DCA07] Chappell David Chappell e Associates. Introducing windows communication foundation, 2007. Disponível em: <http://msdn.microsoft.com/library/dd943056.aspx>, consultado no mês de Janeiro de 2012.
- [Enc12a] PC Magazine Encyclopedia. Definition of mobile app, 2012. Disponível em: http://www.pcmag.com/encyclopedia_term/0,2542,t=mobile+app&i=60015,00.asp, consultado no mês de Maio de 2012.
- [Enc12b] PC Magazine Encyclopedia. Definition of smartphone, 2012. Disponível em: http://www.pcmag.com/encyclopedia_term/0,2542,t=Smartphone&i=51537,00.asp, consultado no mês de Dezembro de 2011.
- [Fie02] ROY T. Fielding. *Principled Design of the Modern Web Architecture*. PhD thesis, University of California, Irvine, 2002.
- [FS11] SA First Solutions. Siima gestão de serviços clínicos, 2011. Disponível em: <http://www.first-global.com/pt/Pages/SiiMA.aspx>, consultado no mês de Dezembro de 2011.

REFERÊNCIAS

- [Gli09] Glintt. Glintt, sector da saúde, 2009. Disponível em: <http://www.glintt.com/sectores/saude>, consultado no mês de Novembro de 2011.
- [HSD11] Gustavo Hartmann, Geoff Stead e Asi DeGani. Cross-platform mobile development. Março 2011. Disponível em: <http://www.triballabs.net/2011/07/cross-platform-mobile-app-development/>, consultado no mês de Novembro de 2011.
- [Inc11a] Appcelerator Inc. Getting started with titanium studio, 2011. Disponível em: <https://wiki.appcelerator.org/display/tis/Getting+Started+with+Titanium+Studio#GettingStartedwithTitaniumStudio-TitaniumFundamentals>, consultado no mês de Novembro de 2011.
- [Inc11b] Appcelerator Inc. Push notifications with urban airship, 2011. Disponível em: <https://wiki.appcelerator.org/display/guides/Push+Notifications+with+Urban+Airship>, consultado no mês de Novembro de 2011.
- [Inc11c] Appcelerator Inc. Titanium platform, 2011. Disponível em: <http://www.appcelerator.com/products/titanium-cross-platform-application-development/>, consultado no mês de Novembro de 2011.
- [Inc11d] Appcelerator Inc. Titanium studio, 2011. Disponível em: <http://www.appcelerator.com/products/titanium-studio/>, consultado no mês de Novembro de 2011.
- [Inc11e] Rhomobile Inc. Rhohub - first development as a service for mobile, 2011. Disponível em: <https://app.rhohub.com/>, consultado no mês de Novembro de 2011.
- [Inc11f] Rhomobile Inc. Rhomobile - mobilize your enterprise apps, 2011. Disponível em: <http://rhomobile.com/>, consultado no mês de Novembro de 2011.
- [Inc12a] Sencha Touch Inc. Sencha docs touch 2.0 - making the web mobile, 2012. Disponível em: <http://docs.sencha.com/touch/2-0/>, consultado no mês de Maio de 2012.
- [Inc12b] Sencha Touch Inc. Sencha touch 2 build mobile web apps with html5, 2012. Disponível em: <http://www.sencha.com/products/touch/>, consultado no mês de Maio de 2012.
- [LAS10] Peter Lubbers, Brian Albers e Frank Salim. *Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development*. Apress, 2010.
- [MBD⁺10] Wallace B. McClure, Martin Bowling, Craig Dunn, Chris Hardy e Rory Blyth. *Professional iPhone Programming with MonoTouch and .NET/C#*. Wiley Publishing, 2010.
- [Mic12] Microsoft. What is windows communication foundation, 2012. Disponível em: [http://msdn.microsoft.com/en-us/library/ms731082\(v=VS.110\).aspx](http://msdn.microsoft.com/en-us/library/ms731082(v=VS.110).aspx), consultado no mês de Janeiro de 2012.

REFERÊNCIAS

- [Mob12a] Mobilewalla. Today's app count, 2012. Disponível em: http://www.mobilewalla.com/Desktop/AppIntel_AppCount.htm?filterDevicePlatform, consultado no mês de Maio de 2012.
- [mob12b] mobithinking. Mobile applications: native v web apps – what are the pros and cons? - interview to global intelligence alliance (gia)'s lie lu, 2012. Disponível em: <http://mobithinking.com/native-or-web-app>, consultado no mês de Maio de 2012.
- [Mor08] Pedro Moreira. *eresults - exploração de dados analíticos de meios complementares de diagnóstico e terapêutica*. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2008.
- [Nal11] Abhishek Nalwaya. *Rhomobile Beginner's Guide - Step-by-step instructions to build an enterprise mobile web application from scratch*. Packt Publishing, 2011.
- [NCSGdI11] LDA New Company Soluções Globais de Inovação. A nossa solução - clínicas médicas, 2011. Disponível em: <http://clinicas.newcompany.pt/solucao02.php>, consultado no mês de Dezembro de 2011.
- [ND11] Daniel Y. Na e Brian C. DeRocher. Html5: What's different for userexperience design and the web? *Sigma Journal*, pages 45–51, Outubro 2011. Disponível em: <http://www.noblis.org/NewsPublications/Publications/TechnicalPublications/SigmaJournal/Pages/default.aspx>, consultado em Novembro de 2011).
- [Nit11a] Nitobi. How phonegap works, 2011. Disponível em: <http://phonegap.com/about>, consultado no mês de Novembro de 2011.
- [Nit11b] Nitobi. Phonegap build: Write once, compile in the cloud, run anywhere, 2011. Disponível em: <https://build.phonegap.com/>, consultado no mês de Novembro de 2011.
- [Nit11c] Nitobi. Tools, 2011. Disponível em: <http://phonegap.com/tools>, consultado no mês de Novembro de 2011.
- [Pop12] Mike Pope. Introducing asp.net web pages 2 - getting started, 2012. Disponível em: <http://www.asp.net/web-pages/tutorials/introducing-aspnet-web-pages-2/getting-started>, consultado no mês de Maio de 2012.
- [Rod08] Alex Rodriguez. *Restful web services: The basics*. Technical report, IBM, Novembro 2008.
- [WW07] Hugh J. Watson e Barbara H. Wixom. The current state of business intelligence. *Computer*, 40(9):96–99, Setembro 2007.
- [Xam11] Xamarin. Introduction to monotouch, 2011. Disponível em: http://docs.xamarin.com/ios/getting_started/Introduction_to_MonoTouch, consultado no mês de Novembro de 2011.