

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



FEUP

Adaptação de funcionalidades do portal Infovini a dispositivos móveis

Nuno Miguel Sanches Ferreira de Almeida

Relatório de Projecto

Mestrado Integrado em Engenharia Informática e Computação

Orientador: António Pimenta Monteiro (Professor Doutor)

Julho de 2008

Adaptação de funcionalidades do portal Infovini a dispositivos móveis

Nuno Miguel Sanches Ferreira de Almeida

Relatório de Projecto

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo júri:

Presidente: Ana Cristina Ramada Paiva Pimenta (Professora Doutora)

Arguente: Manuel Bernardo Barbosa (Professor Doutor)

Vogal: António Miguel Pontes Pimenta Monteiro (Professor Doutor)

17 de Julho de 2008

Resumo

A disseminação generalizada de dispositivos móveis com um número crescente de capacidades que vão muito para além das funcionalidades básicas de que outrora dispunham reflecte os avanços tecnológicos que se têm verificado ao nível das Tecnologias da Informação. O acesso à Internet é hoje em dia uma característica comum neste tipo de aparelhos, razão pela qual se têm vindo a desenvolver soluções na Web adaptadas a este tipo de suportes.

O *Infovini – Vinhos de Portugal* é um portal orientado à promoção e divulgação de vinhos desenvolvido pelo INEGI (Instituto de Engenharia Mecânica e Gestão Industrial) com o apoio de instituições do sector vitivinícola que pretende tornar-se numa referência nacional e internacional na divulgação dos vinhos portugueses na Internet. A concepção de soluções para a disponibilização das suas funcionalidades em dispositivos móveis é um objectivo prioritário para a consolidação da imagem de qualidade e inovação associada ao Infovini.

A apresentação de conteúdos em dispositivos móveis difere da apresentação em computadores pessoais devido fundamentalmente às restrições ao nível da memória disponível, capacidade de processamento, interface com o utilizador e velocidades de acesso à rede. O trabalho realizado no âmbito do projecto consistiu, numa primeira fase, no estudo das funcionalidades do portal passíveis de disponibilização em suporte móvel, que foram ordenadas de acordo com prioridades definidas juntamente com a equipa de desenvolvimento. A esta análise seguiu-se o levantamento de requisitos e o estudo do estado da arte relacionado com a disponibilização de conteúdos e desenvolvimento de aplicações para dispositivos móveis. Tendo presentes as várias alternativas possíveis, decidiu-se sobre a melhor forma de apresentação considerando os requisitos definidos anteriormente e fez-se o estudo das tecnologias que possibilitariam o desenvolvimento da solução proposta. Com base nesse estudo e nos requisitos não funcionais estabelecidos, concluiu-se que a disponibilização deveria ser feita sob a forma de aplicações desenvolvidas em Java ME que esporadicamente e consoante as opções do utilizador, transferissem e apresentassem dados contactando Web Services ou procedimentos da camada de lógica de negócio do portal, ambos implementados em PHP. Uma vez definida a arquitectura da solução, seguiu-se o estudo dos protocolos, bibliotecas e *frameworks* que possibilitaram a concepção da solução proposta.

As três aplicações criadas, em conjunto com as *interfaces* desenvolvidas do lado do servidor, permitem a um utilizador de um dispositivo vulgar com acesso à Internet e suporte a Java MIDP 2.0 ter acesso a sugestões de vinhos em função da ocasião em que se encontra e procurar vinhos existentes na base de dados do Infovini, assim como visualizar mapas de itinerários e informações sobre pontos de interesse, particularmente úteis no contexto do enoturismo - o turismo associado à cultura vitícola, uma forma de turismo em franca expansão nos países produtores de vinho. A arquitectura adoptada possibilita e facilita o desenvolvimento de futuras aplicações que pretendam aceder a informação presente na base de dados a partir dos serviços disponibilizados, contribuindo também para a missão do Infovini de promover e divulgar o sector vitivinícola.

Abstract

The generalized dissemination of mobile devices, with a large number of capabilities that go far beyond the functionality they were primarily designed for, reflects the technological improvements that have been made on the IT industry. Nowadays, the ability to access the Internet is a common feature of those devices and the reason why a growing number of solutions on the Web adapted to mobile supports have been developing.

Infovini – Wines of Portugal is a Web portal aimed to promoting the Portuguese wines. Developed by INEGI (Instituto de Engenharia Mecânica e Gestão Industrial), supported the by Wine Committees, the portal intends to be a reference, both nationally and internationally, spreading the Portuguese wines over the Internet. One of the Infovini top priorities is to create solutions to make it available to mobile devices.

As mobile devices differ from laptops, the information must be presented bearing in mind restrictions on available memory, processing power, graphical user interface and network bandwidth. Firstly, the portal's relevant functionalities able to adapt to mobile supports were analyzed. Those functionalities were sorted according to levels of priority set by the author and the development team. Upon the fulfillment of this task came the analysis of the requirements needed for the next stage of the project. Alongside it was studied the art of developing applications for mobile devices and the best approach was decided bearing in mind the previously defined requirements. After dedicated study, it was decided to provide the information on Java ME support. The system was designed so that Internet connections are only established when the user reaches an information that needs data transferences using Web Services or procedures of the portal's business logic layer, both written in PHP. After defining the architecture of the system to develop, research was made on protocols, libraries and frameworks to make it possible to create the proposed solution.

The user of a mobile device with Java MIDP 2.0 support and capable of accessing the Internet, is now able to use the three applications created alongside with the interfaces developed on the server side to browse wine suggestions for a particular occasion, search for a specific wine on the database or even consult itineraries and information about points of interest, a particularly useful tool on oenotourism context – an expanding business related with tourism on wine culture. The adopted architecture simplifies the development of new applications which intend to access data present on the database using the services created, contributing to Infovini's goal of marketing and promoting the wine sector.

Agradecimentos

Gostaria de agradecer ao meu supervisor, Nuno Ramos, pela disponibilidade para o esclarecimento de dúvidas e por todo o apoio prestado ao longo da minha estadia no INEGI. Os meus agradecimentos vão também para todos os elementos da equipa com a qual trabalhei, pela ajuda, simpatia e pelo óptimo ambiente que proporcionaram, nomeadamente a Ana Marques, o Fernando Jesus, o Miguel Fernandes e o João Cardoso, meu colega da FEUP e também a estagiar no INEGI.

Aproveito também para endereçar os meus agradecimentos à Professora Doutora Henriqueta Nóvoa, gestora de projecto, por toda a preocupação demonstrada e incentivos, assim como ao Professor Doutor António Pimenta Monteiro, meu orientador, pelo apoio relativamente à documentação e orientação ao nível das tecnologias.

Não podia deixar de agradecer também à minha família por todo o apoio, nomeadamente aos meus pais, à minha avó e à Guida, assim como aos meus amigos de curso Daniel Cordeiro, Hernâni Fernandes e Luis Ferreira, aos meus amigos de sempre Mário Costa e Ricardo Teixeira e a todos os que simpatizam comigo.

Por fim agradeço ao INEGI pelas boas condições de trabalho proporcionadas principalmente no novo edifício.

Conteúdo

1.	Introdução.....	1
1.1.	Contexto / Enquadramento.....	1
1.2.	Projecto	2
1.3.	Estrutura do relatório	3
2.	Enquadramento geral	5
2.1.	Apresentação de conteúdos em dispositivos móveis	5
2.1.1.	Sites adaptados	5
2.1.2.	Aplicações	7
2.2.	Estado da arte	7
2.2.1.	Sites otimizados para dispositivos móveis	7
2.2.2.	Aplicações para dispositivos móveis	9
2.2.3.	Conclusão	10
2.3.	Funcionalidades a disponibilizar	11
2.3.1.	“Uma Ocasão, um Vinho”	11
2.3.2.	“Pesquisa de Vinhos”	12
2.3.3.	“Rotas do Vinho”	14
2.4.	Forma de disponibilização adoptada	16
2.5.	Requisitos do sistema.....	17
2.5.1.	“Uma Ocasão, um Vinho”	17
2.5.2.	“Pesquisa de Vinhos”	18
2.5.3.	“Rotas do Vinho”	19
2.5.4.	Requisitos não funcionais	20
2.6.	Revisão tecnológica	21
2.6.1.	.NET CF	21
2.6.2.	Programação para Symbian OS.....	23
2.6.3.	Java ME.....	24
3.	Especificação.....	28
3.1.	Linguagens de implementação	28
3.1.1.	Cliente	28
3.1.2.	Servidor	28
3.2.	Arquitectura do portal Infovini.....	29
3.3.	Arquitecturas adoptadas.....	31
3.3.1.	Arquitectura de “Uma Ocasão, um Vinho” e “Pesquisa de Vinhos”	31
3.3.2.	“Rotas dos Vinhos”	35
3.4.	Tecnologias adoptadas	40
3.4.1.	Ambiente de desenvolvimento	40
3.4.2.	Implementações de SOAP	41
3.4.3.	Interface	43
4.	Implementação.....	48

4.1.	Método de armazenamento local	48
4.2.	Características comuns.....	49
4.3.	Servidor	50
4.3.1.	Web Services.....	50
4.3.2.	PhpThumb	51
4.3.3.	Procedimentos do lado do servidor para “Rotas do Vinho”	52
4.4.	Aplicação “Uma Ocasão, um Vinho”	58
4.4.1.	Definição da ocasião.....	58
4.4.2.	Visualização de sugestões.....	59
4.4.3.	Visualização de resultados	60
4.5.	Aplicação “Pesquisa de Vinhos”	63
4.5.1.	Pesquisa básica	63
4.5.2.	Pesquisa avançada	64
4.5.3.	Visualização de resultados	65
4.6.	Aplicação “Rotas dos Vinhos”	66
4.6.1.	Definição da região e do itinerário	66
4.6.2.	Mapa de itinerário.....	66
4.6.3.	Lista de pontos de interesse.....	67
4.6.4.	Ficha de ponto de interesse	68
5.	Conclusões	71
5.1.	Perspectivas de desenvolvimento futuro.....	71
5.2.	Reacções.....	72
	Referências	73
	Apêndice A: Planeamento do projecto	74
	Apêndice B: Relatório Intermédio – Análise do portal Infovini	75

Lista de figuras:

Fig. 1: Logótipo do INEGI.....	1
Fig. 2: O portal Infovini.	2
Fig. 3: Comunicação em WAP 1.0.	6
Fig. 4: Site da Google optimizado para dispositivos móveis.	9
Fig. 5: Aplicação <i>Mobile Weather</i>	10
Fig. 6: Funcionalidade "Uma Ocasão, um Vinho" (portal).....	12
Fig. 7: Formulário de pesquisa de vinhos (portal).	13
Fig. 8: Exemplo de uma ficha de vinho (portal).....	13
Fig. 9: Escolha da rota na funcionalidade "Rotas do Vinho" (portal).	14
Fig. 10: Escolha do itinerário na funcionalidade "Rotas dos Vinhos" (portal).	15
Fig. 11: Percurso com informação sobre um local na funcionalidade "Rotas do Vinho" (portal).....	16
Fig. 12: Diagrama de casos de uso para "Uma Ocasão, um Vinho".	18
Fig. 13: Diagrama de casos de uso para "Pesquisa de Vinhos".....	19
Fig. 14: Diagrama de casos de uso para "Rotas do Vinho".....	20
Fig. 15: Arquitectura de um ambiente .NET CF.	22
Fig. 16: Fases de compilação de uma aplicação .NET CF.....	23
Fig. 17: Arquitectura de um ambiente Java ME.....	25
Fig. 18: Diagrama de camadas do portal Infovini.	30
Fig. 19: Utilização do UDDI na descoberta de serviços.	32
Fig. 20: Diagrama de camadas adoptado, com Web Services.....	35
Fig. 21: Diagrama de camadas do servidor para "Rotas do Vinho".	38
Fig. 22: Diagrama da comunicação de "Rotas do Vinho" para apresentação de itinerários (dispositivo capaz de abrir URL longos).	39
Fig. 23: Diagrama da comunicação de "Rotas do Vinho" para apresentação de itinerários (dispositivo incapaz de abrir URL longos).	39
Fig. 24: Diferenças ao nível do <i>choice group</i> inicial.	44
Fig. 25: <i>Screenshots</i> do menu inicial da aplicação de sugestões de vinhos.	45
Fig. 26: <i>Screenshot</i> de um exemplo do site do MWT num Nokia N80.....	46
Fig. 27: Arquitectura adoptada em “Uma Ocasão, um Vinho” e “Pesquisa de Vinhos” (lado do cliente).	47
Fig. 28: Diagrama de actividades do início da execução das aplicações.	50
Fig. 29: Diagrama de tabelas relativas à aplicação "Rotas do Vinho".....	53
Fig. 30: Eliminação de pontos irrelevantes em <code>getItinerarioMapa.php</code>	56
Fig. 31: Exemplo de fases na definição da ocasião na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.	59

Fig. 32: Sugestão genérica na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.	60
Fig. 33: Lista de vinhos na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis. .	61
Fig. 34: Diagrama de actividades da transferência de vinhos nas aplicações "Uma Ocasão, um Vinho" e "Pesquisa de Vinhos".	61
Fig. 35: Ficha de vinho na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis. ..	62
Fig. 36: Menu de rótulo na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis. .	63
Fig. 37: Diagrama de actividades da transferência de um rótulo nas aplicações "Uma Ocasão, um Vinho" e "Pesquisa de Vinhos".	63
Fig. 38: Pesquisa básica da aplicação "Pesquisa de Vinhos" para dispositivos móveis.	64
Fig. 39: Pesquisa avançada na aplicação "Pesquisa de Vinhos" para dispositivos móveis.	64
Fig. 40: Resultados das pesquisas básica e avançada na aplicação "Pesquisa de Vinhos" para dispositivos móveis.	65
Fig. 41: Definição da rota e do itinerário na aplicação "Rotas do Vinho" para dispositivos móveis.	66
Fig. 42: Mapa de itinerário na aplicação "Rotas do Vinho" para dispositivos móveis.	67
Fig. 43: <i>Screenshots</i> relativos a uma legenda de um mapa na aplicação "Rotas do Vinho". ..	68
Fig. 44: Ficha de local na aplicação "Rotas do Vinho" para dispositivos móveis.	69
Fig. 45: Imagem de um local na aplicação "Rotas do Vinho" para dispositivos móveis.	69
Fig. 46: Mapa de local na aplicação "Rotas do Vinho" para dispositivos móveis.	70

Lista de Tabelas

Tabela 1: Métodos HTTP e REST.....	34
Tabela 2: Reduções dos pontos das <i>polylines</i>	55

Lista de acrónimos:

.NET CF	.NET Compact Framework
AJAX	Assynchronous JavaScript And XML
API	Application Programming Interface
CDC	Connected Device Configuration
CIL	Common Intermediate Language
CLDC	Connected, Limited Device Configuration
CLR	Common Language Runtime
CSS	Cascading Style Sheets
CVRVV	Comissão de Viticultura da Região dos Vinhos Verdes
FIRE	Flexible Interface Rendering Engine
GPRS	General Packet Radio Service
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated development environment
IMAP	Internet Message Access Protocol
IMP	Information Modulo Profile
IP	Internet Protocol
ISP	Internet Service Provider
ITU	International Telecommunication Union
IVDP	Instituto dos Vinhos do Douro e Porto
IVV	Instituto da Vinha e do Vinho
J4ME	Java For Me
JAR	Java ARchive
Java ME	Java Platform, Micro Edition
JIT	Just-In-Time
KVM	“Kilo” Virtual Machine
MIDP	Mobile Device Profile
MWT	Micro Window Toolkit

PDA	Personal Digital Assistant
PHP	PHP: Hypertext Preprocessor
REST	Representational State Transfer
RMS	Record Management System
RPC	Remote Procedure Call
SDK	Software Development Kit
SGML	Standard Generalized Markup Language
SIS	Symbian Installation Source
SMS	Short Message Service
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol (originalmente)
UDDI	Universal Description, Discovery and Integration
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WML	Wireless Markup Language
WSDL	Web Services Description Language
XHTML	Extensible Hypertext Markup Language

1. Introdução

O presente relatório descreve o projecto de conclusão do Mestrado Integrado em Engenharia Informática e Computação (MIEIC), da Faculdade de Engenharia da Universidade do Porto (FEUP), realizado pelo autor no Instituto de Engenharia Mecânica e Gestão Industrial (INEGI) no período compreendido entre 18 de Fevereiro e 7 de Julho de 2008.

Ao longo deste capítulo será feita a apresentação da empresa, da equipa e do projecto, serão descritos os objectivos que se pretendem alcançar com o mesmo e por fim será apresentada a estrutura do documento.

1.1. Contexto / Enquadramento

O INEGI (Fig. 1) é uma instituição que se define como uma Associação Privada sem Fins Lucrativos vocacionada para a realização de actividades de inovação. Nasceu em 1986 no seio do Departamento de Engenharia Mecânica e Gestão Industrial (DEMEGI) da Faculdade de Engenharia da Universidade do Porto (FEUP) e assume um papel de parceiro da indústria em projectos de Investigação e Desenvolvimento.



Fig. 1: Logótipo do INEGI

De entre os vários departamentos que compõem a instituição destaca-se a unidade de Gestão e Engenharia Industrial (GEIN) que engloba diversas áreas, entre elas a área de Sistemas de Informação e Comércio Electrónico onde o autor se insere. De entre os vários projectos associados à equipa multidisciplinar da área referida destaca-se o Infovini – Vinhos de Portugal¹ (Fig. 2), um portal de promoção e divulgação de vinhos que está a ser desenvolvido com o apoio de organismos do sector vitivinícola, nomeadamente o Instituto da Vinha e do Vinho, Instituto Público (IVV, IP) e o Instituto dos Vinhos do Douro e Porto, Instituto Público (IVDP, IP).

¹ Disponível em <http://www.infovini.com/>.



Fig. 2: O portal Infovini.

O trabalho realizado insere-se nas áreas de Desenvolvimento de Software para Dispositivos Móveis, Interacção e Multimédia e Tecnologias de Informação e Comunicações e consistiu na adaptação para suportes móveis de um conjunto de funcionalidades do portal.

1.2. Projecto

Com a generalização do uso de dispositivos móveis capazes de aceder a conteúdos disponibilizados na Web e dada a ambição da equipa de desenvolvimento em tornar o Infovini o portal de referência na divulgação dos vinhos nacionais na Internet, surgiu a necessidade de conceber uma solução que fosse capaz de facilitar a interacção entre os utilizadores destes aparelhos e os conteúdos do portal. A comodidade associada ao uso de dispositivos de bolso que permitem efectuar ligações à rede a partir de qualquer ponto do globo fez com que o seu uso para este efeito se tornasse cada vez mais popular. Devido fundamentalmente às dimensões reduzidas dos ecrãs, às limitações de memória, à menor capacidade de processamento e à limitada largura de banda das ligações à Internet, a apresentação de conteúdos de forma optimizada para estes aparelhos difere da exposição dos mesmos em computadores pessoais. Daí surgiu a necessidade de adaptação das funcionalidades já existentes no portal, cujo desenvolvimento tinha sido inicialmente orientado apenas a utilizadores de computadores comuns. Todas as especificidades relacionadas com *hardware/software* tais como a plataforma, a capacidade de processamento, a memória, a largura de banda, os comandos existentes ou a resolução de ecrã diferem entre os vários

modelos. A implementação de uma solução para utilizadores de dispositivos móveis teve forçosamente que ter em conta a exigência de flexibilidade associada a estes pressupostos.

O projecto consistiu, numa primeira fase, no estudo do estado da arte relacionado com a apresentação de conteúdos em dispositivos móveis, assim como na análise das funcionalidades mais relevantes para disponibilizar e dos requisitos envolvidos. Tendo em conta a necessidade de minimização da quantidade de dados transferidos e das vantagens inerentes à apresentação de algumas informações sem necessidade de ligação à Internet, concluiu-se que a melhor forma de disponibilização seria sob a forma de aplicações. Com o estudo das tecnologias que se seguiu, inferiu-se que a mais adequada tendo em conta os requisitos não funcionais estabelecidos seria o Java ME.

As aplicações desenvolvidas poderão envolver a transferência de dados do servidor onde se encontra alojado o portal Infovini, tendo sido por isso necessário apurar a tecnologia de implementação do lado do servidor assim como o protocolo de comunicação usado. De forma a reutilizar a camada de lógica de negócio já implementada em PHP que serve o portal Infovini e uma vez que não existiam desvantagens inerentes, foi esta a linguagem utilizada do lado do servidor. Considerando o requisito de interoperabilidade e as vantagens inerentes ao desenvolvimento de uma solução concordante com os *standards*, a forma de comunicação escolhida para duas das funcionalidades adaptadas foi baseada em Web Services usando o protocolo SOAP. Toda a análise comparativa das tecnologias e as justificações pela escolha de umas em detrimento de outras é documentada no presente relatório, nos capítulos 2 e 3.

O produto desenvolvido cumpre com os requisitos estabelecidos e tem um carácter inovador na medida em que não existe qualquer solução no sector vitivinícola que se assemelhe. As aplicações criadas oferecem aos utilizadores de dispositivos móveis a liberdade de acederem a informação detalhada sobre os vinhos presentes na base de dados do Infovini assim como a roteiros turísticos e a informação sobre os respectivos pontos de interesse a partir de qualquer ponto do globo, desde que exista a possibilidade de ligação à Internet. Parte dos dados é armazenada localmente, conferindo assim utilidade às aplicações mesmo em situações em que não seja possível estabelecer ligações à rede.

1.3. Estrutura do relatório

Para além da introdução, este documento contém mais quatro capítulos.

No segundo capítulo é descrito o estado da arte relacionado com a disponibilização de conteúdos em dispositivos móveis e com o desenvolvimento de aplicações para estes aparelhos. É também documentada a revisão tecnológica que foi realizada relativamente às principais tecnologias a utilizar.

O terceiro capítulo é dedicado à especificação, sendo apresentado com detalhe o problema que se pretende resolver, a arquitectura anterior do portal e a solução proposta, sem detalhes de implementação.

No quarto capítulo são apresentados detalhes de nível mais baixo relativos à implementação das soluções descritas no terceiro capítulo.

No quinto capítulo é apresentado um resumo do trabalho realizado e tiradas conclusões relativamente à satisfação dos objectivos propostos inicialmente. São também analisadas perspectivas de desenvolvimento futuro.

2. Enquadramento geral

Este capítulo contém uma secção introdutória breve destinada a apresentar os principais problemas e formas de apresentação de conteúdos em dispositivos móveis. É também descrito o estado da arte expondo soluções existentes no mercado no domínio do desenvolvimento de soluções para estes suportes, tanto no sector vitivinícola como em outros contextos. Por fim é efectuada uma revisão tecnológica das principais ferramentas passíveis de utilização no âmbito do projecto.

2.1. Apresentação de conteúdos em dispositivos móveis

A apresentação de conteúdos em dispositivos móveis difere da apresentação em computadores pessoais devido fundamentalmente às restrições ao nível da memória disponível, capacidade de processamento e interface com o utilizador. Os acessos à rede também são realizados a menores velocidades e podem eventualmente envolver custos associados elevados, que podem depender do tempo de ligação ou da quantidade de dados transferida. Além disto, uma vez que a grande maioria destes aparelhos dispõe de teclados ITU-T² ou de ecrãs tácteis com teclados virtuais onde a introdução de texto não se faz tão comodamente como em teclados normais, é imperativo que a navegação se faça com o mínimo possível de *input* textual por parte do utilizador.

2.1.1. Sites adaptados

A criação de sites optimizados para dispositivos móveis é uma forma de adaptação de conteúdos pensada para os utilizadores destes aparelhos que pretendam aceder a páginas Web a partir dos seus *browsers*. Os dispositivos com capacidades de ligação à rede contêm aquilo que se chamam *microbrowsers* (ou *mini-browsers*), ou seja, *browsers* para navegação Web adaptados às características do aparelho. Apesar dos *browsers* de alguns telemóveis com menores capacidades de processamento serem apenas capazes de interpretar sites WAP³ (em WML⁴), os mais recentes já possuem características muito próximas dos navegadores para computadores pessoais comuns, conseguindo interpretar código XHTML⁵ ou mesmo HTML⁶ *standard*.

O WAP é um *standard* aberto para a apresentação de conteúdos em clientes capazes de ligação sem fios à Internet. Na sua primeira versão exigia a existência de um *gateway*

² O ITU-T ou ITU Telecommunication Standardization Sector é uma organização responsável por definir *standards* para o sector das telecomunicações em parceria com a ITU - International Telecommunication Union. Os teclados de telefones ITU-T contêm teclas de 0 a 9 que também permitem a introdução dos 26 caracteres do alfabeto, onde cada uma delas pode ser usada para digitar mais do que uma letra.

³ Wireless Application Protocol.

⁴ Wireless Markup Language.

⁵ Extensible Hypertext Markup Language.

⁶ Hypertext Markup Language.

WAP que servia de intermediário entre o dispositivo e o servidor onde se encontravam alojadas as páginas WML. Este *gateway* funcionava como um *proxy* que recebia o pedido WAP, transformava-o num pedido HTTP e enviava ao servidor. A resposta WML recebida era depois verificada e transformada em WML binário, interpretado pelo *microbrowser*. O diagrama de funcionamento encontra-se esquematizado na Fig. 3.

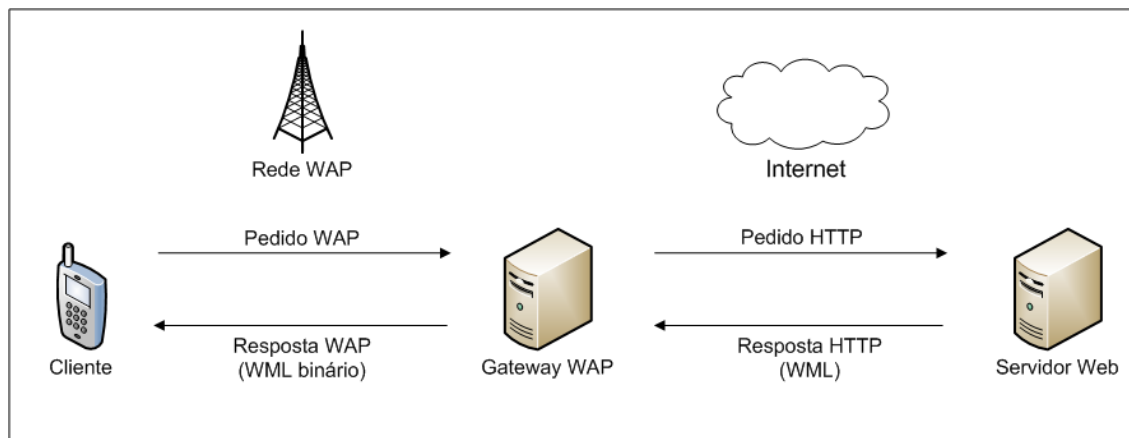


Fig. 3: Comunicação em WAP 1.0.

Com o aparecimento de dispositivos com maiores capacidades de processamento e maiores velocidades de transferência de dados surgiu o WAP 2.0. Esta especificação suporta XHTML⁷ assim como CSS⁸. A tarefa de criação de páginas ficou assim simplificada já que as mesmas passaram a ser acessíveis tanto por computadores comuns como por dispositivos móveis, desde que escritas em XHTML (apesar de que os *browsers* dos aparelhos mais recentes são já muito semelhantes aos de computadores de secretária no sentido em que já são capazes de interpretar todos os elementos do HTML *standard*, assim como JavaScript e AJAX⁹). No entanto, devido aos tamanhos reduzidos dos ecrãs, ao facto de as velocidades de transferência ainda serem bastante menores do que as velocidades disponibilizadas pelos ISP¹⁰ comuns e devido a existir ainda uma percentagem significativa de utilizadores de aparelhos mais antigos que não dispõem destas funcionalidades, faz sentido a criação de páginas optimizadas para dispositivos móveis, com um número reduzido de imagens e apenas os conteúdos mais relevantes. O Infovini não se encontra actualmente adaptado às restrições inerentes à utilização de dispositivos móveis. Não existe uma versão optimizada para estes aparelhos e a versão existente contém elementos em Flash, JavaScript e AJAX, não suportados por grande parte deles. Algumas das funcionalidades encontram-se implementadas em Flash, ficando desta forma inacessíveis a todos os utilizadores de dispositivos cujos *browsers* não o suportem. A navegação faz-se a partir de uma barra de menus também ela em Flash, o que dificulta ou mesmo impossibilita o acesso a determinadas secções do site.

⁷ O XHTML é uma linguagem de *markup* com as mesmas *tags* do HTML e baseada em XML que surgiu fundamentalmente porque se pretendia que os *browsers* dos dispositivos com menores capacidades de processamento não fossem obrigados a interpretar páginas escritas em HTML baseado em SGML, já que é muito menos dispendioso a nível de recursos interpretar XML.

⁸ Cascading Style Sheets.

⁹ Asynchronous JavaScript and XML.

¹⁰ Internet Service Provider.

2.1.2. Aplicações

As aplicações são uma outra forma de desenvolvimento para suportes móveis, neste caso independente do *browser* presente no dispositivo. Existem actualmente várias tecnologias sobre as quais se podem implementar aplicações deste tipo, que serão abordadas mais adiante na secção 2.6. Uma solução baseada em aplicações permite estabelecer ligações esporádicas à rede para transferir dados. É uma solução vantajosa quando existe uma quantidade significativa de *inputs* por parte do utilizador que podem ser inseridos sem necessidade de ligação à Internet. Esta abordagem permite também ter um elevado grau de confiança sobre a forma como os conteúdos são mostrados, sem haver a necessidade de preocupações com o suporte a tecnologias subjacentes (como por exemplo a JavaScript ou a CSS no caso dos *browsers*). Os dados transferidos podem dizer respeito apenas ao conteúdo ao invés de serem orientados também à apresentação. Além disto, as aplicações permitem armazenar dados de forma persistente ou conterem elas mesmas informação que pode ser apresentada ao longo da sua execução sem necessidade de conectar à rede. Esta forma de apresentação de conteúdos tem a desvantagem de certas tecnologias estarem dependentes da plataforma, como no caso do Symbian ou do .NET CF¹¹ por exemplo, falados na secção dedicada à revisão tecnológica.

2.2. Estado da arte

Como foi referido, existem duas formas distintas de disponibilização de conteúdos adaptados a dispositivos móveis. De seguida apresentam-se os resultados do estudo das soluções existentes actualmente tanto ao nível da optimização de sites como da criação de aplicações. A análise que foi feita engloba soluções existentes para o sector vitivinícola e também outras soluções tecnologicamente próximas daquilo que se pretende, não necessariamente ligadas ao sector.

2.2.1. Sites optimizados para dispositivos móveis

Além do Infovini, existe actualmente uma grande variedade de sites nacionais dedicados ao sector vitivinícola mas nenhum deles dispõe de uma versão para dispositivos móveis. Estes sites podem ser organizados em três diferentes categorias: instituições, empresas e lojas. As instituições (tanto governamentais como comissões vitivinícolas e institutos) têm como missão regulamentar a produção, divulgar produtores, vinhos, eventos e notícias, assim como eventualmente promover o enoturismo nas diferentes regiões. Exemplos disso são os sites do IVV, I. P.¹² e do IVDP, I. P.¹³. Os objectivos principais dos sites de empresas são a apresentação da marca, a divulgação dos seus vinhos, a publicação de notícias sobre eventos e prémios e em certos casos possibilitar compras *online*. Entre os sites desta categoria encontram-se por exemplo o da SOGRAPE¹⁴ e o do grupo Aveleda¹⁵. As lojas

¹¹ Compact Framework.

¹² Disponível em <http://www.ivv.min-agricultura.pt/>.

¹³ Disponível em <http://www.ivdp.pt/>.

¹⁴ Disponível em <http://www.sogrape.pt/>.

¹⁵ Disponível em <http://www.aveleda.pt/>.

dedicam-se fundamentalmente à venda dos seus produtos através do comércio electrónico, disponibilizando também directórios de empresas e produtores, notícias, eventos, conselhos para servir e apreciar vinhos, assim como alguma informação histórica. Exemplos de sites deste tipo são por exemplo o da *Lusowine*¹⁶. A navegação nos sites analisados através de dispositivos móveis exige deslocamentos horizontais e tempos de espera longos devido ao carregamento de imagens, assim como impossibilita o acesso a determinadas secções devido à inclusão de itens em Flash, por exemplo.

A única solução pensada nos utilizadores de dispositivos móveis a nível nacional que se conhece é a do portal *NovaCrítica-vinho*¹⁷ que dispõe de um sistema em que se envia uma mensagem SMS para um número (com um custo associado) e se recebe como resposta um URL que possibilita o acesso a partir do dispositivo ao serviço de visualização de notas de prova publicadas pela equipa *NovaCrítica*, por um período de 30 dias.

Existem alguns sites estrangeiros de *e-commerce* de vinhos que já têm em conta as exigências de acessibilidade relativas ao desenvolvimento para suportes móveis como o *WineZap*¹⁸ que redirecciona o utilizador deste tipo de aparelhos para a versão optimizada¹⁹ do mesmo. Outros, como é o caso do *Mobile Wine Club*²⁰ disponibilizam páginas com menos imagens e com um *layout* que se adapta às dimensões do dispositivo. No entanto não se conhecem exemplos de portais de promoção e divulgação (não direccionados ao comércio electrónico) adaptados a estes aparelhos - exigem normalmente o uso da barra de deslocamento horizontal (como por exemplo o *Wines from Austria*²¹, dedicado à divulgação dos vinhos austríacos), a quantidade de imagens/animações mostradas não é optimizada tendo em conta o dispositivo e certos conteúdos não são acessíveis devido ao facto de a tecnologia usada para os apresentar não ser suportada por alguns *microbrowsers*. Um exemplo disso é a inclusão de itens em Flash, como acontece no *Infovini* e noutros portais internacionais de divulgação como no *Wines of France*²², dedicado à promoção dos vinhos franceses.

Fora da esfera do sector vitivinícola, existe um número crescente de sites que têm em conta as normas de acessibilidade ou que disponibilizam versões adaptadas a dispositivos móveis como é o caso da versão *mobile* do *Google*²³ (Fig. 4), da *Wikipedia*²⁴ e do *GMail*²⁵.

¹⁶ Disponível em <http://www.lusowine.com/>.

¹⁷ Disponível em <http://www.novacritica-vinho.com/>.

¹⁸ Disponível em <http://winezap.com/>.

¹⁹ Disponível em <http://winezap.com/mobile/a.cfm>.

²⁰ Disponível em <http://mobilewineclub.net/>.

²¹ Disponível em <http://www.winesfromaustria.com/>.

²² Disponível em <http://www.frenchwinesfood.com/>.

²³ Disponível em <http://www.google.com/> (caso detecte que o cliente é um dispositivo móvel).

²⁴ Disponível em <http://en.wap.wikipedia.org/>.

²⁵ Disponível em <http://m.gmail.com/>.



Fig. 4: Site da Google optimizado para dispositivos móveis.

2.2.2. Aplicações para dispositivos móveis

Não se conhecem aplicações para dispositivos móveis relacionadas com o sector vitivinícola, tanto no mercado nacional como no internacional.

Existe actualmente um vasto leque de aplicações (não relacionadas com o sector vitivinícola) que usam a mesma abordagem tecnológica que se pondera usar neste projecto, estabelecendo ligações esporádicas à rede para obter dados a pedido do utilizador. Um exemplo disso é a aplicação *Mobile Weather*²⁶ (Fig. 5), que contacta um Web Service da *Yahoo!* pedindo previsões meteorológicas para um determinado local.

²⁶ Disponível em <http://www.ubahnstation.net/projects/mweather/mweather.html>.

Fig. 5: Aplicação *Mobile Weather*.

Usando esta aplicação, o utilizador evita os tempos de espera resultantes da abertura do site existente para o efeito. A qualquer altura o utilizador pode usar a aplicação para se ligar à Internet e transferir as previsões para as localidades que tiver definido (cerca de 2.5 *kilobytes* de dados por localidade). A definição da localização cujas previsões se pretende consultar é feita *offline*, já que as localizações disponíveis se encontram armazenadas na própria aplicação (ocupando cerca de 750 *kilobytes*). Uma vez definida uma localização, a aplicação estabelece conexão e retorna os dados essenciais como a temperatura, a direcção/velocidade do vento, as horas de amanhecer/anoitecer e a descrição do estado do tempo (“Partly Cloudy” ou “Mostly Cloudy”, por exemplo). Um outro exemplo de uma aplicação para dispositivos móveis é a disponibilizada pela equipa do *GMail*²⁷, desenvolvida em Java e que permite a sincronização com a caixa de entrada por IMAP. A equipa do Google disponibiliza também a aplicação *Google Maps* para dispositivos com SymbianOS 9 e para Pocket PC com Windows Mobile²⁸.

2.2.3. Conclusão

Exceptuando o serviço disponibilizado pela *NovaCrítica-vinho*, não existem soluções a nível nacional para a apresentação em suportes móveis de conteúdos da Web relacionados com o sector vitivinícola. Esta situação deve-se principalmente ao facto de as equipas de desenvolvimento terem descurado a crescente utilização destes aparelhos para aceder à Internet e considerarem que o desenvolvimento de soluções adaptadas aos mesmos não se justifica. Com este projecto pretende-se colmatar esta lacuna e desta forma dar um passo importante na promoção dos vinhos portugueses e do portal Infovini, que poderá então

²⁷ Disponível em <http://gmail.com/app/>.

²⁸ Disponíveis em <http://www.google.pt/gmm/>.

afirmar-se como um dos pioneiros, tanto nacional como internacionalmente, na disponibilização de conteúdos relacionados com o sector de forma optimizada para dispositivos móveis.

2.3. Funcionalidades a disponibilizar

Depois do estudo do portal *Infovini* e de uma reunião com a equipa responsável pela elaboração e pelos conteúdos do mesmo, concluiu-se que as funcionalidades a adaptar no âmbito do projecto são, por ordem decrescente de prioridade:

- "Uma Ocasão, um Vinho";
- "Pesquisa de Vinhos";
- "Rotas do Vinho".

Todas as funcionalidades foram analisadas e ordenadas por ordem de prioridades, tendo dessa análise resultado um relatório intermédio que pode ser consultado no Apêndice B. Na presente secção são abordadas as três funcionalidades que foram seleccionadas para adaptação no âmbito do projecto.

2.3.1. “Uma Ocasão, um Vinho”

Esta funcionalidade encontra-se no portal implementada em Flash (Fig. 6) e a partir dela o utilizador pode visualizar uma lista de sugestões acerca dos vinhos mais apropriados para uma determinada ocasião. Essa ocasião é uma combinação entre a situação (romântica, festa, convívio, entre outras), o tipo de refeição (carne, peixe, saladas, entre outros) e o seu subtipo (que no caso de “vegetariano” poderá ser “com soja”, por exemplo, e no caso de “carne” poderá ser “branca” e “grelhada”, por exemplo).



Fig. 6: Funcionalidade "Uma Ocasião, um Vinho" (portal).

A adaptação desta funcionalidade é particularmente útil para utilizadores de dispositivos móveis já que é interessante que em qualquer lugar possam receber sugestões de vinhos para uma ocasião do momento. Durante uma convivência/reunião com pessoas conhecidas (ou mesmo um pouco antes) é muitas vezes útil, principalmente se alguns dos elementos forem apreciadores, ter uma ideia das características que deve ter o vinho que mais se adequa à ocasião, como por exemplo o tipo, a casta ou a região. A sua adaptação é também crucial devido ao facto de ter sido originalmente implementada em Flash, o que faz com que esteja inacessível a todos os utilizadores cujos dispositivos não suportem esta tecnologia.

No portal Infovini, a visualização das etapas que definem uma ocasião é feita a partir de interrogações sucessivas à base de dados, à medida que se avança. No caso da aplicação para dispositivos móveis, uma vez que importa minimizar as ligações à rede, a definição da ocasião poderá ser feita localmente, tendo a própria aplicação informação sobre as diferentes etapas.

2.3.2. "Pesquisa de Vinhos"

Permite pesquisar vinhos segundo um conjunto de atributos considerados mais relevantes, a saber: nome, produtor, casta, tipo de vinho, região, colheita/idade e preço. Na Fig. 7 pode ver-se um formulário de pesquisa.

Montes Claros Pesquisar

Opções avançadas

produtor casta

Branco Alentejo colheita/idade preço

1 vinhos. Pág. 1 | 1

i
O Infovini ainda se encontra em desenvolvimento, por isso informamos que é apenas possível efectuar pesquisas nas regiões do Alentejo, Douro e Porto e Vinhos Verdes.

Montes Claros Reserva Branco
Branco | Alentejo
Adega Cooperativa de Borb...

Fig. 7: Formulário de pesquisa de vinhos (portal).

Os resultados são disponibilizados como uma lista de hiperligações para as páginas dos vinhos que contêm informações sobre os mesmos (como o tipo, a região ou o produtor) assim como imagens que poderão ser da garrafa, do rótulo e/ou do contra-rótulo (no caso da ficha de vinho que se segue existe apenas a imagem do rótulo), tal como na Fig. 8.

Ficha de Vinho

Montes Claros Reserva Branco

Nome Montes Claros Reserva Branco

Tipo Branco

Região Alentejo

Designação oficial DOC

Produtor Adega Cooperativa de Borba

Cor Cor esverdeada a dourada

Aroma Aroma limpo e de carácter elegante com notas de frutos tropicais, limão e tangerina, com barrica discreta, ligeira baunilha

Sabor Macio e muito fresco ao sabor, acidez bem presente, persistente a frutos citrinos e tropicais, estruturado e harmonioso

Rótulo

Fig. 8: Exemplo de uma ficha de vinho (portal).

A implementação desta funcionalidade adaptada a dispositivos móveis foi considerada prioritária já que pode ser de grande utilidade tanto para apreciadores como para o público em geral dispor de um catálogo que permita pesquisar por vinhos e visualizar algumas das suas informações em qualquer lugar. Num dispositivo móvel pretende-se que seja possível escolher um vinho de uma lista de resultados e visualizar as suas informações (incluindo a imagem do rótulo, considerada a mais relevante das três imagens para fácil identificação de um vinho numa prateleira).

2.3.3. “Rotas do Vinho”

Esta última das três funcionalidades escolhidas para adaptação sugere itinerários a turistas interessados em conhecer adegas, vinhas e tradições das regiões vitivinícolas depois do utilizador indicar a rota pretendida (Fig. 9).

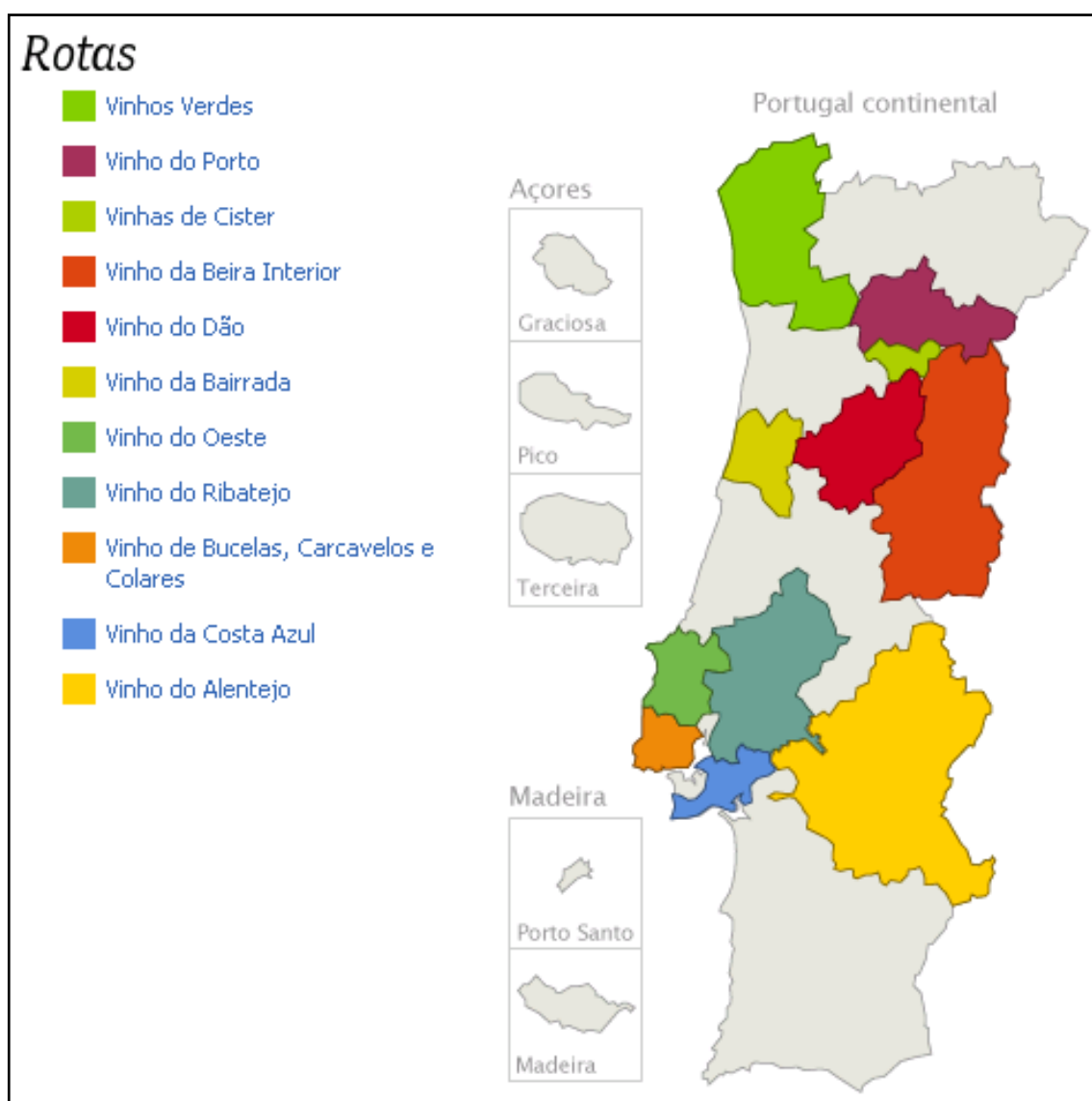


Fig. 9: Escolha da rota na funcionalidade "Rotas do Vinho" (portal).

Actualmente, dependendo da rota escolhida, pode indicar de 2 a 8 itinerários que são definidos pela sua entidade responsável. Cada itinerário pode ter um nome associado. No caso de não ter, é mostrado apenas o seu número identificador, como na Fig. 10.

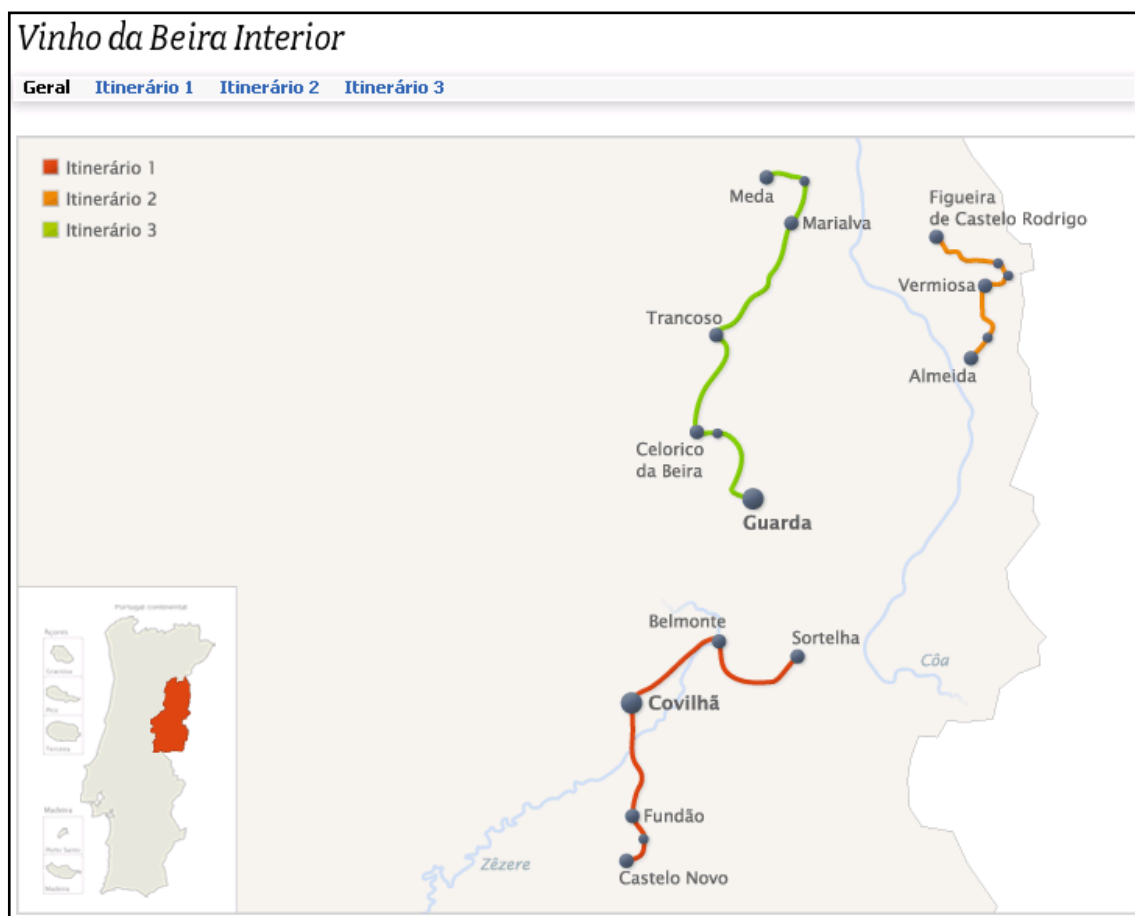


Fig. 10: Escolha do itinerário na funcionalidade "Rotas dos Vinhos" (portal).

Um itinerário é um percurso entre vários pontos de interesse. Cada percurso é definido como um conjunto de pontos. Cada itinerário é mostrado como uma imagem do Google Maps dando ao utilizador a possibilidade de fazer *zoom*, deslocar-se no mapa e visualizar uma descrição e uma fotografia de cada local ao seleccioná-lo. Na Fig. 11 pode ver-se uma imagem de um percurso com informação sobre um ponto de interesse.



Fig. 11: Percurso com informação sobre um local na funcionalidade "Rotas do Vinho" (portal).

Esta é uma funcionalidade considerada prioritária já que é útil no contexto do enoturismo ter a possibilidade de consultar percursos quando não se tem acesso a um computador com internet (um exemplo típico dessa situação ocorre quando se está a viajar e se pretende saber o que se pode encontrar na região) nem a um dispositivo de navegação por satélite. Pretende-se que os itinerários disponíveis sejam mostrados numa lista e que ao escolher cada um deles se possam ver as imagens do Google Maps no dispositivo móvel, proporcionando ao utilizador uma experiência próxima daquela que tem quando acede ao portal a partir de um computador pessoal.

2.4. Forma de disponibilização adoptada

O modo escolhido para a adaptação das funcionalidades enunciadas anteriormente a dispositivos móveis foi através de aplicações. As razões que levaram a escolher este formato foram as seguintes:

- Nas três funcionalidades, pretende-se que o utilizador tenha a liberdade de definir concretamente aquilo que pretende visualizar enquanto em modo *offline*. No caso da primeira aplicação poderá definir a ocasião (que envolve diversas etapas), no caso da segunda funcionalidade poderá introduzir os parâmetros de pesquisa e no caso da terceira deverá poder escolher qual a região e o percurso que pretende. Só depois de definida a informação concreta à qual pretende aceder é que o utilizador é convidado a

estabelecer conexão. Se se optasse por criar uma versão do site otimizada, esta escolha seria feita *online*, implicando o *download* de informação que poderia estar armazenada localmente.

- É uma mais-valia que parte da informação de “Uma Ocasão, um Vinho” e de “Rotas do Vinho” possa estar disponível sem necessidade de ligação à Internet. No caso da primeira, pretende-se que as sugestões genéricas de vinhos (que mostram o tipo, região e castas indicadas dada a ocasião) possam ser disponibilizadas enquanto *offline* de modo a conferir utilidade à aplicação mesmo quando não existe forma de estabelecer ligação à rede. No caso de “Rotas do Vinho” pretende-se que o utilizador possa ter acesso aos nomes dos locais de todos os itinerários e aos contactos das quintas para que em viagem possa, sem necessidade de ligar à Internet, saber como as contactar de forma a marcar visitas ou pedir informações. Desenvolver uma versão do portal otimizada para dispositivos móveis em vez de aplicações implicava que a consulta de qualquer tipo de informação iria obrigar ao estabelecimento de uma ligação à Internet, o que eventualmente poderia não ser possível ou conveniente do ponto de vista do utilizador.
- Interessa minimizar a quantidade de dados a transferir, excluindo informação relativa à apresentação. A apresentação fica assim dependente da forma como as aplicações tratam o conteúdo.
- A independência em relação ao *browser* do dispositivo confere maior flexibilidade no que respeita à construção de interfaces e à interacção com o utilizador.

2.5. Requisitos do sistema

Os requisitos das aplicações a desenvolver foram apurados com base em casos de uso. Seguem-se os diagramas de casos de uso para cada uma das três aplicações e respectivos comentários aos mesmos.

2.5.1. “Uma Ocasão, um Vinho”

Tal como na aplicação em Flash presente no portal Infovini, pretende-se que o utilizador visualize três sugestões genéricas de vinhos indicando o tipo, a região e a(s) casta(s). A partir de cada sugestão, deverá ser dada a possibilidade de ligação à internet para permitir ao utilizador visualizar uma lista aleatória de vinhos com as características sugeridas. A partir dessa lista deverá ser possível visualizar fichas de vinho com os seguintes campos:

- nome do vinho;
- idade ou colheita;
- nome do produtor;
- região;
- tipo;
- consumo;

- temperatura ideal do vinho;
- teor alcoólico.

A visualização de rótulos deverá ser opcional para o utilizador já que a sua transferência pode envolver custos e deverá ser possível a partir das fichas de vinhos. Na Fig. 12 pode ver-se o diagrama de casos de uso que reflecte estes requisitos:

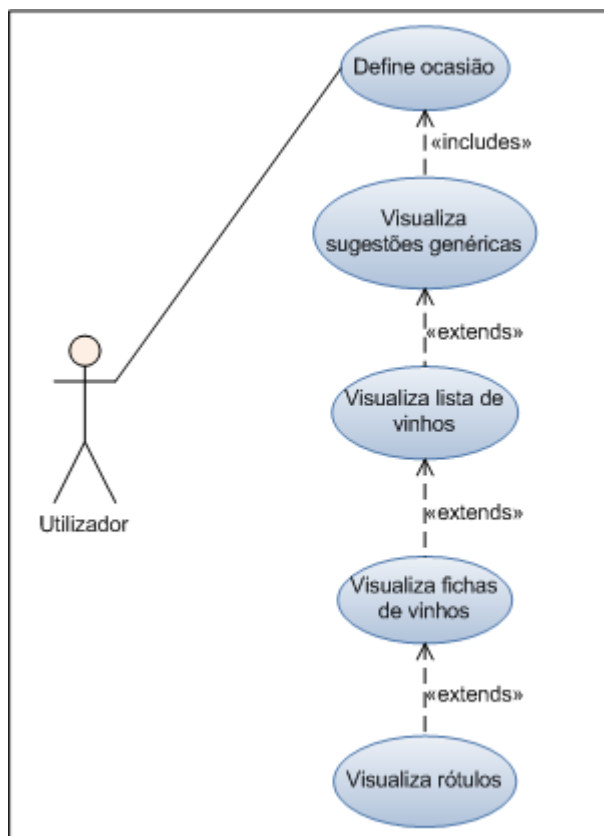


Fig. 12: Diagrama de casos de uso para "Uma Ocasião, um Vinho".

2.5.2. “Pesquisa de Vinhos”

Tal como na pesquisa do portal, pretende-se que seja possível especificar quais os parâmetros de pesquisa. Adicionalmente, pretende-se também que a versão para dispositivos móveis permita uma pesquisa básica, com apenas um campo de pesquisa genérico que se pode referir a qualquer um dos parâmetros, já que a introdução de texto nos teclados destes aparelhos não é tão cómoda como nos teclados dos computadores pessoais. Manter-se-á o formulário de pesquisa avançada com todos os campos presentes na versão do portal com excepção do campo de pesquisa do preço uma vez que ainda não existe informação relativa aos mesmos na base de dados. A aplicação deverá portanto, na pesquisa avançada, incluir os seguintes campos de pesquisa:

- nome do vinho;
- nome do produtor;
- casta;

- colheita/idade;
- tipo (como lista *dropdown*);
- região (como lista *dropdown*).

Uma vez que esta é uma aplicação direccionada unicamente para a pesquisa de vinhos, prevê-se que seja usada em situações em que se procure informações sobre um vinho específico. Desta forma, a lista de vinhos retornada não deverá ser aleatória mas sim ordenada segundo um parâmetro escolhido pelo utilizador. Deverá também ser mostrado o número total de vinhos encontrado (além dos recebidos pela aplicação) assim como a possibilidade de avançar na lista para os próximos resultados. Excluindo estas pequenas diferenças, o funcionamento a partir do momento em que os resultados são obtidos é semelhante ao da aplicação “Uma Ocasão, um Vinho”. O diagrama de casos de uso correspondente pode ser consultado na Fig. 13.

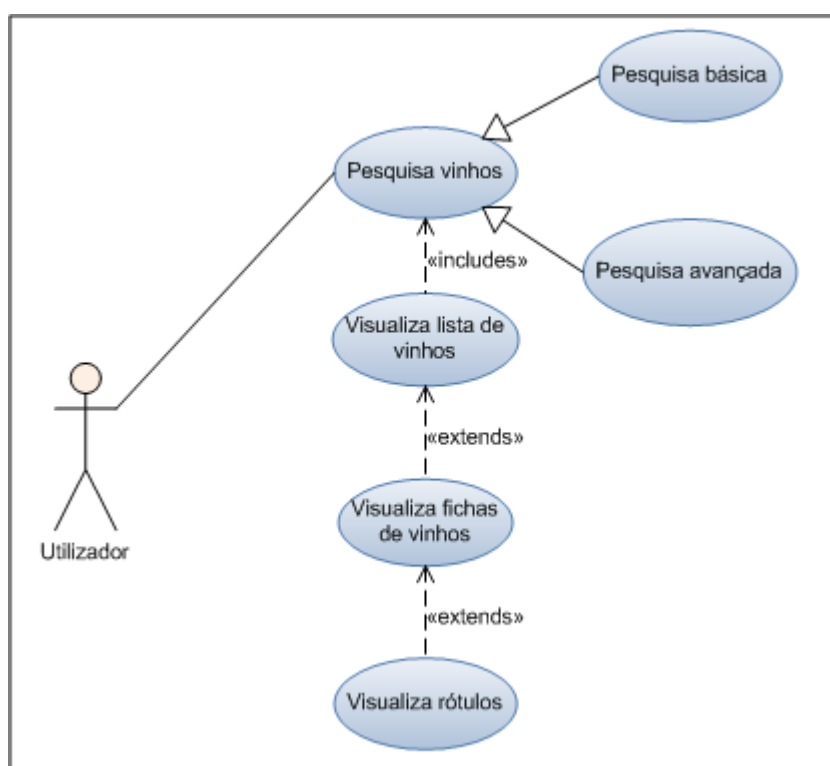


Fig. 13: Diagrama de casos de uso para "Pesquisa de Vinhos".

2.5.3. “Rotas do Vinho”

A aplicação destinada a mostrar roteiros turísticos consoante a região escolhida deverá permitir aos utilizadores visualizar tanto os mapas como as legendas associadas aos mesmos, com as respectivas descrições e fotografias. De modo a dotar a aplicação de utilidade mesmo quando não existe ligação à rede, deverão ser incluídas no próprio código as descrições das quintas²⁹, já que contêm informações turísticas úteis como moradas, telefones e horários das

²⁹ Existem quatro tipos de locais: quintas, povoações, monumentos e parques naturais.

visitas guiadas. As descrições das povoações, monumentos e parques naturais não necessitam de estar armazenadas localmente já que consistem em informações históricas e curiosidades que não se consideram cruciais em viagem. Deverão ser portanto acessíveis a partir de ligações à rede. O utilizador deverá também ser capaz de efectuar deslocamentos e alterar os níveis de *zoom* dos mapas. O diagrama de casos de uso referente a esta funcionalidade pode ser consultado na Fig. 14.

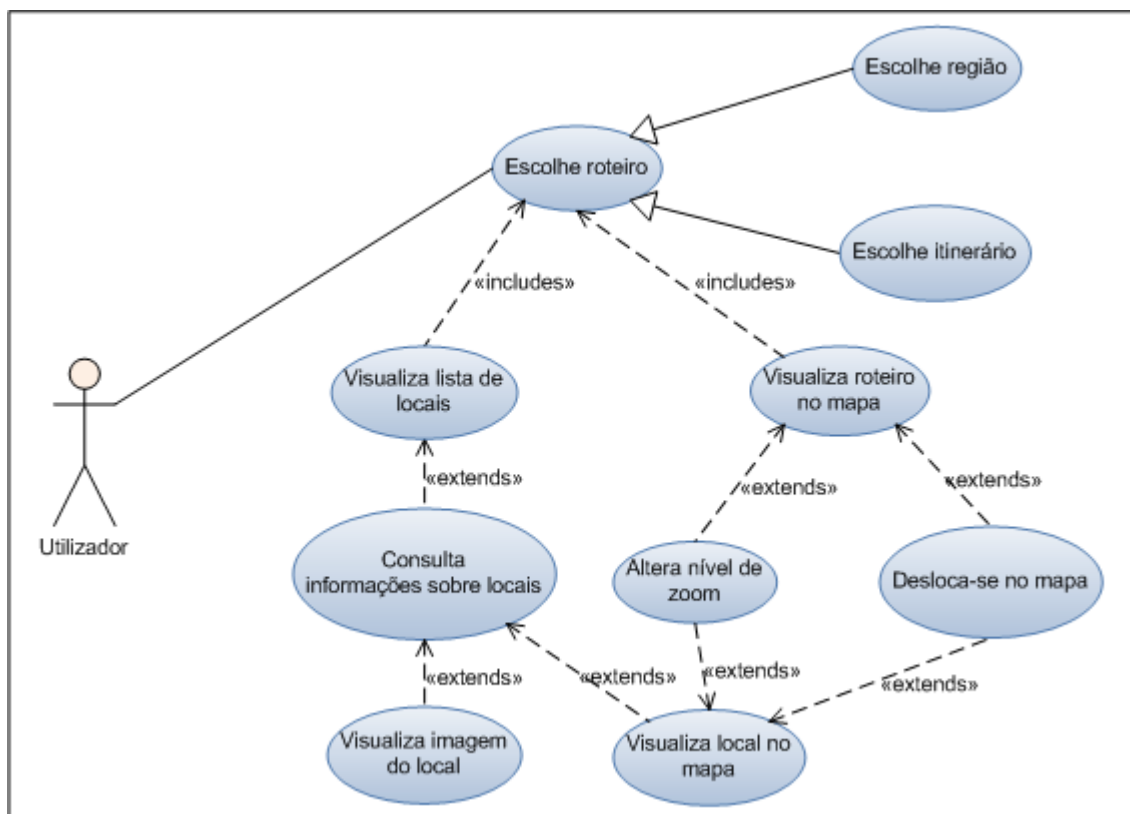


Fig. 14: Diagrama de casos de uso para "Rotas do Vinho".

2.5.4. Requisitos não funcionais

Existem alguns requisitos não funcionais a ter em conta no sistema a desenvolver, que são, por ordem decrescente de importância:

- **portabilidade:** uma vez que se pretende que as aplicações possam ser executadas em múltiplas plataformas de forma a abranger a maior parte dos dispositivos móveis existentes no mercado.
- **usabilidade:** já que se pretende que a interação do utilizador com as aplicações seja simples e intuitiva.
- **interoperabilidade:** seria uma mais-valia que o sistema criado permitisse que eventuais novas aplicações implementadas em diferentes tecnologias pudessem usar as infra-estruturas de comunicação desenvolvidas no âmbito do projecto.
- **flexibilidade:** pois é importante que as aplicações sejam capazes de se adaptar às características dos dispositivos de forma a aproveitar ao máximo as suas potencialidades.

2.6. Revisão tecnológica

É uma prioridade que as aplicações sejam compatíveis com o maior número de dispositivos móveis possível, de forma a abrangerem o maior número de potenciais utilizadores. As principais tecnologias para o desenvolvimento de aplicações para dispositivos móveis, tendo em conta as plataformas dominantes no mercado são .NET CF, Symbian C++ e Java ME³⁰.

Tanto .NET CF como Symbian C++ são dependentes da plataforma e permitem a criação de aplicações para dispositivos móveis com Windows CE ou Symbian OS, respectivamente.

As aplicações Java são amplamente suportadas pela grande parte dos dispositivos móveis actualmente. Os primeiros telemóveis a suportar a primeira versão do MIDP³¹ (um perfil Java ME abordado mais adiante) começaram a ser comercializados em 2001 e desde então o suporte ao Java foi-se generalizando. A maior parte dos dispositivos comercializados a nível mundial, com excepção feita ao mercado da América do Norte³² [Can], dispõe de sistemas operativos Symbian³³ [Can] nos quais o suporte a Java ronda os 98%³⁴ [GSM08]. Em outros dispositivos o suporte a Java não é tão abrangente mas é ainda assim elevado, rondando os 69%³⁵ [GSM08] no caso dos dispositivos Microsoft (o sistema operativo mais usado na Europa a seguir ao Symbian).

Do lado do servidor, a única tecnologia que interessa estudar é aquela em que toda a lógica de negócio foi implementada (neste caso PHP), tal como referido anteriormente. De seguida é feita uma abordagem mais aprofundada de cada uma destas tecnologias.

2.6.1. .NET CF

A .NET Compact Framework é uma versão da *framework* .NET concebida para a criação de aplicações para dispositivos móveis com sistemas operativos Windows (Pocket PC e *smartphones* com Windows Mobile, assim como dispositivos a correr Windows CE .NET 4.1 ou superiores). Contém bibliotecas específicas para dispositivos móveis que não existem na *framework* .NET.

A .NET CF usa o CLR³⁶, uma camada responsável pela execução das aplicações que corre acima do sistema operativo. O diagrama da Fig. 15 mostra a arquitectura de um ambiente .NET CF.

³⁰ Java Platform, Micro Edition.

³¹ Mobile Information Device Profile.

³² Na América do Norte a liderança é repartida entre sistemas operativos Palm e Microsoft.

³³ Dados relativos ao quarto trimestre de 2007 obtidos dos comunicados à imprensa da Canalys, uma respeitada empresa de consultoria e análise de mercados relacionados com tecnologias da informação.

³⁴ Este valor foi obtido comparando os resultados das pesquisas por todos os dispositivos Symbian com os resultados de aqueles que suportam Java.

³⁵ Este valor foi obtido comparando os resultados das pesquisas por todos os dispositivos Symbian com os resultados de aqueles que suportam Java.

³⁶ Common Language Runtime.



Fig. 15: Arquitectura de um ambiente .NET CF.

Estas aplicações podem ser criadas usando o ambiente de desenvolvimento Visual Studio com a .NET CF (como Visual Studio 2003, 2005 e 2008) em qualquer uma das linguagens suportadas (como C/C++, VB.NET ou C#). Numa primeira fase, o compilador .NET converte o código fonte para CIL³⁷ que depois, em *runtime*, é compilado para código binário nativo pelo compilador do CLR - o *JIT*³⁸ *compiler*. O código compilado é independente da linguagem usada na implementação. As diferentes fases de compilação encontram-se esquematizadas na Fig. 16.

³⁷ Common Intermediate Language.

³⁸ Just-In-Time.

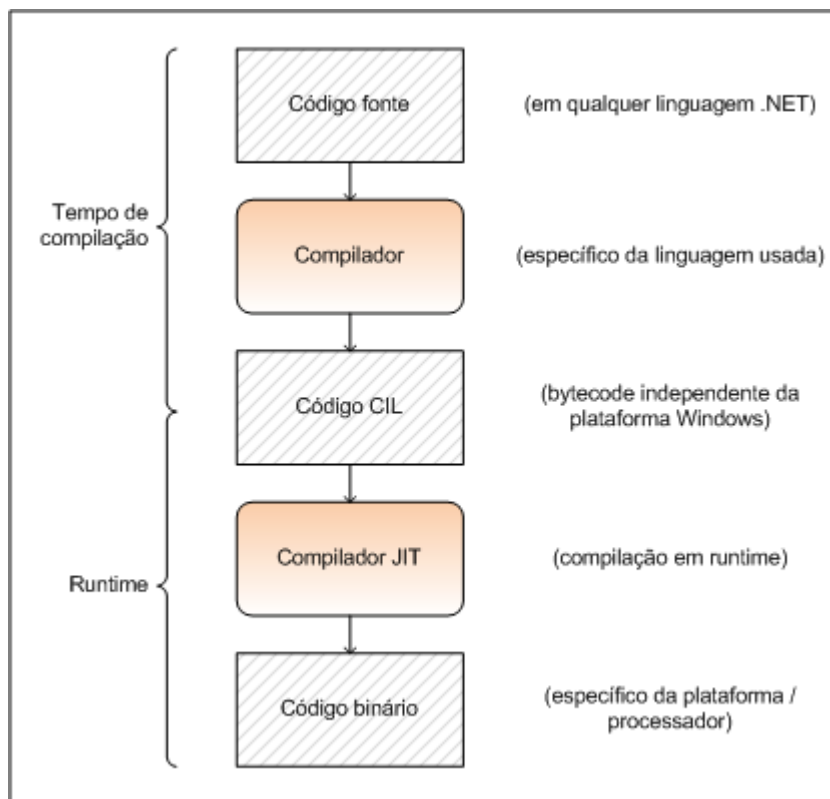


Fig. 16: Fases de compilação de uma aplicação .NET CF.

As aplicações criadas podem interagir com bases de dados SQL Server CE, que é uma versão compacta do SQL Server para dispositivos móveis com sistemas operativos Windows.

A grande limitação das aplicações .NET CF no âmbito do projecto prende-se com o facto de não serem independentes da plataforma. O dispositivo móvel em questão teria de dispor de uma plataforma Windows CE, o que não acontece com os telemóveis que normalmente (tendo em conta o mercado actual) correm Symbian e mesmo os PDA podem correr outros sistemas operativos como Palm OS ou versões do Linux.

Esta limitação é impeditiva uma vez que o utilizador que não disponha de um dispositivo a correr uma versão do Windows CE (que tenha por exemplo um telemóvel comum com Symbian) não poderia usar as aplicações criadas.

2.6.2. Programação para Symbian OS

É possível desenvolver aplicações para Symbian que correm directamente a partir da camada do sistema operativo. O desenvolvimento deste tipo de aplicações é feito utilizando um IDE, existindo alguns especializados para o efeito. Também é possível fazê-lo com o Microsoft Visual Studio com o auxílio de um *plugin*. Os ficheiros instaladores gerados após a compilação são ficheiros SIS³⁹ com extensão .SIS ou .SISX⁴⁰.

³⁹ Symbian Installation Source.

⁴⁰ Os ficheiros com extensão .SISX só são compatíveis com dispositivos a correr Symbian 9.1 ou superiores.

A linguagem normalmente usada é o C++ mas é também possível desenvolver em Python, Visual Basic, VB.NET, C#, entre outras. As últimas três linguagens enunciadas poderiam ser usadas recorrendo a um *plugin* para o Visual Studio criado pela AppForge. Esta empresa cessou as suas actividades em 2007 e foi adquirida pela Oracle que optou por não dar continuidade ao projecto, acabando com os serviços de suporte e a manutenção das licenças (para se poder usar o *plugin* era necessário renovar a licença anualmente), situação que dificultou o desenvolvimento.

Existem duas grandes desvantagens da programação em Symbian C++. Uma delas tem a ver com a complexidade do código. Para programar para Symbian C++ é necessário ter-se em conta aspectos de baixo nível que introduzem complexidade no código. O mecanismo de tratamento de excepções do C++ foi posto de parte tendo-se argumentado que a sua inclusão aumentava o tamanho do código compilado. Devido a isto, é o programador que tem que ter em conta aspectos de baixo nível, tendo por exemplo que fazer *pushs* e *pops* a uma *cleanup stack* (que mantém apontadores para os dados alocados no *heap* e que é usada para os limpar caso uma excepção ocorra, já que nesse caso não interessa mantê-los em memória). Além de introduzirem complexidade no código, estas técnicas fazem com que o programador se distancie das verdadeiras funcionalidades da aplicação que deseja implementar. A outra desvantagem tem a ver com o facto de as aplicações desenvolvidas com base nesta tecnologia só serem compatíveis com dispositivos a correr Symbian (telemóveis / *smartphones*), o que vai contra o objectivo de criar aplicações que abranjam a maior gama de aparelhos possível.

2.6.3. Java ME

O Java ME, anteriormente denominado J2ME, é uma especificação da plataforma Java desenvolvida com o intuito de disponibilizar um grupo de bibliotecas da API do Java para o desenvolvimento de aplicações para dispositivos com recursos limitados (tais como telemóveis e PDA). O facto de ser independente da plataforma, correndo numa camada superior à do sistema operativo (numa máquina virtual), permite que o mesmo código binário possa correr em sistemas diferentes e que as aplicações sejam testadas previamente com emuladores. Quanto aos Pocket PC, alguns já trazem de origem uma máquina virtual Java instalada, outros não. Para os utilizadores de Pocket PC que não dispõem de uma KVM⁴¹ (denominação dada a máquina virtuais Java para dispositivos com recursos muito limitados) de origem, existem algumas distribuições proprietárias como a Esmertec Jbed Advanced Phone Engine⁴². A única máquina virtual gratuita conhecida de momento é a MySaifu JVM⁴³ mas nos testes efectuados não correu as aplicações correctamente.

Em Java ME existem os conceitos de configuração e de perfil, que podem ou não ser suportados pelos dispositivos. A arquitectura de um ambiente Java ME é a mostrada na Fig. 17.

⁴¹ “Kilo” Virtual Machine.

⁴² Disponível em <http://www.esmertec.com>.

⁴³ Disponível em http://www2s.biglobe.ne.jp/~dat/java/project/jvm/index_en.html.



Fig. 17: Arquitectura de um ambiente Java ME.

Segue-se uma explicação detalhada destes dois conceitos.

2.6.3.1. Configurações

Configurações são subconjuntos de bibliotecas Java de baixo nível e de características da sua máquina virtual presentes num ambiente Java ME. Existem duas configurações:

CLDC (Connected Limited Device Configuration)

O CLDC é uma especificação orientada a aplicações Java ME para dispositivos com recursos limitados como *paggers* e telemóveis, contendo por isso um subconjunto restrito de classes. Os requisitos mínimos recomendados são [Jor07]:

- processador de 16 ou 32 bit;
- 128 KB (CLDC 1.0) ou 160 KB (CLDC 1.1) de memória não volátil disponível para a máquina virtual e para as bibliotecas CLDC;
- 32 KB de memória volátil disponível para o *runtime* da máquina virtual;
- baixo consumo de energia, frequentemente disponível a partir de uma bateria;
- ligação a algum tipo de rede, normalmente sem fios, intermitente e de largura de banda limitada.

Em combinação com um perfil (como o MIDP), fornece aos programadores uma base sólida para a criação de aplicações Java. Existem duas especificações do CLDC: a 1.0 e a 1.1. Esta última inclui algumas funcionalidades não incluídas na 1.0 como suporte a operações de vírgula flutuante, inclusão de novas classes, adição de alguns métodos em determinadas classes de forma a tornar a especificação mais próxima do J2SE e algumas correcções de *bugs* existentes na primeira especificação.

CDC (Connected Device Configuration)

O CDC é uma configuração orientada a dispositivos com mais recursos como PDA. As características típicas destes dispositivos são [Jor07]:

- Processador de 32 bit;

- 2MB de memória disponível para a plataforma Java;
- Conectividade a algum tipo de rede, frequentemente sem fios, com largura de banda limitada;
- Interface com o utilizador com alguma sofisticação.

2.6.3.2. Perfis

Os dispositivos Java ME implementam perfis. Os perfis assentam sobre as configurações e consistem num conjunto de bibliotecas de mais alto nível. O MIDP é um exemplo de um perfil e foi criado para dispositivos com um qualquer tipo de *display*, contendo por isso uma API para interagir com uma interface. O IMP⁴⁴ é um outro perfil que não inclui suporte para a *package* javax.microedition.lcdui (uma API para implementação de interfaces suportada pelo MIDP) e que foi criado para dispositivos sem qualquer tipo de interface de utilizador ou com um *display* muito rudimentar como máquinas de *vending*.

No caso do presente projecto interessa portanto o perfil MIDP já que o suporte a uma GUI⁴⁵ é fundamental.

MIDP 1.0 e MIDP 2.0

Existem actualmente três versões de MIDP - 1.0, 2.0 e 2.1, estando já em discussão a especificação MIDP 3. A versão 2.0, actualmente suportada pela maioria dos telemóveis, oferece melhorias significativas relativamente a MIDP 1.0, como a obrigatoriedade em suportar distribuição *Over-The-Air*, uma API de interface gráfica estendida, suporte para reprodução de áudio e vídeo, conectividade expandida (em MIDP 1.0 só havia suporte para HTTP enquanto que na versão 2.0 existe suporte para HTTPS e *sockets*, por exemplo), entre outras. No que diz respeito à GUI, existe um conjunto significativo de melhorias introduzido com a segunda versão do perfil. Em MIDP 2.0, a API da interface de alto nível permite o uso de mais componentes como *choice groups* do tipo POPUP (similares a *dropdown lists*) enquanto que em MIDP 1.0 só havia a possibilidade de usar *choice groups* dos tipos MULTIPLE (semelhantes a *check boxes*) ou EXCLUSIVE (semelhantes a *radio buttons*). Existe também a possibilidade de incluir comandos em alertas (permitindo fazer perguntas ao utilizador usando estes itens), assim como *gauges* (barras de progresso). Em relação à API de baixo nível, o MIDP 2.0 oferece uma série de melhorias como uma API para jogos 2D, possibilidade de usar modo *full screen*, entre outros.

Ao longo dos testes que foram sendo realizados, reparou-se que o uso da API de alto nível por vezes difere de dispositivo para dispositivo. Por exemplo, alguns dispositivos reservam um botão para comandos do tipo EXIT (para sair da aplicação), ignorando a prioridade que foi estabelecida para o mesmo (numa aplicação com vários comandos, é possível definir-se uma prioridade para cada um deles de forma a se ter acesso directo aos mais importantes e os outros aparecerem dentro de um menu). Nesses dispositivos, mesmo que se tenha definido um comando com prioridade 0⁴⁶ e se tenha atribuído prioridade inferior

⁴⁴ Information Module Profile.

⁴⁵ Graphical User Interface.

⁴⁶ Neste caso, quanto mais baixo for o valor, mais alta é a prioridade. O valor 0 corresponde à prioridade máxima.

(superior a 0) ao botão de EXIT, este aparece com acesso directo, podendo relegar o outro para dentro de um menu, caso o dispositivo só disponibilize dois botões. Além disto, notaram-se diferenças muito significativas no aspecto de alguns itens como *choice groups*, que diferiam bastante mesmo para dispositivos da mesma marca (neste caso fala-se concretamente de dois dos modelos onde foram realizados testes: o Nokia N80 e o Nokia 6230i). Esta situação é conhecida e será abordada mais à frente, na secção 3.5.3. Para contornar o problema, foram criadas ao longo do tempo algumas *frameworks* que fazem uso da API de baixo nível para melhorar e unificar o aspecto dos componentes, tornando a interface mais profissional e mais “standardizada”. A maior parte das bibliotecas deste tipo que foram encontradas no estudo das tecnologias requerem suporte para MIDP 2.0, tais como a FIRE⁴⁷ e a J4ME⁴⁸, que serão abordadas mais adiante na secção 3.5.3.

A especificação MIDP 2.0 implica o cumprimento dos seguintes requisitos mínimos de hardware [Jor07]:

Ecrã:

- Tamanho: 96x54 píxeis.
- Profundidade de cor: 1 bit.
- Forma do pixel (*aspect ratio*): aproximadamente 1:1.

Entrada:

- Teclado de telefone ITU-T ou teclado QWERTY.

Memória:

- 256 KB de memória não volátil para os componentes MIDP.
- 8 KB de memória não volátil para dados persistentes criados pelas aplicações.
- 128 KB de memória volátil para o *runtime* do Java.

Comunicação:

- Comunicação sem fios, nos dois sentidos, possivelmente intermitente e com largura de banda limitada.

MIDP 2.1

O MIDP 2.1 foi definido de forma a tentar aproximar as diferentes implementações de MIDP 2.0 dos diversos dispositivos e reduzir a fragmentação. Especifica que tecnologias Java têm obrigatoriamente que estar presentes para um dispositivo ser considerado compatível com MIDP 2.1. Por exemplo, algumas implementações do MIDP 2.0 permitiam o uso de determinados eventos no modo gráfico (como Pointer Events ou Repeat Events), enquanto que outras não. Não era por isso garantido que certos dispositivos com suporte a MIDP 2.0 tivessem determinadas características. Agora, caso um dispositivo esteja conforme as normas estabelecidas em MIDP 2.1, é certo que o suporte a esse tipo de funcionalidades existe.

⁴⁷ Flexible Interface Rendering Engine.

⁴⁸ Java For Me.

3. Especificação

Uma vez que inicialmente ainda não estava definido o conjunto de funcionalidades a desenvolver, o planeamento inicial agregava a implementação de todas as aplicações numa única etapa. Após a definição desse conjunto de funcionalidades foi possível especificar o tempo previsto para cada uma das funcionalidades. O planeamento do projecto encontra-se disponível no Apêndice A.

3.1. Linguagens de implementação

3.1.1. Cliente

Uma vez que é preponderante que as aplicações a desenvolver no âmbito do projecto abranjam a maior gama de dispositivos possível, decidiu adoptar-se o Java ME. Não existe actualmente nenhuma tecnologia de desenvolvimento de aplicações para dispositivos móveis com o grau de portabilidade desta tecnologia, que como foi referido anteriormente, permite criar aplicações suportadas por virtualmente todos os telemóveis actuais existentes no mercado. Esta escolha não compromete a interface nem as funcionalidades a disponibilizar já que a configuração/perfil escolhidos em conjunto com algumas bibliotecas/*frameworks* de terceiros (referidas mais adiante na secção 3.5), disponibilizam um vasto leque de classes que as permitem implementar. Desta forma possibilita-se o uso das aplicações a utilizadores de telemóveis assim como de outros dispositivos (como Pocket PC ou até mesmo PlayStation Portable) bastando para isso terem uma KVM instalada.

A configuração alvo será CLDC (compatível com dispositivos com poucos recursos que suportem Java) e a versão será a 1.1 já que é comum em dispositivos com MIDP 2.0 e porque contém classes requeridas pela *framework* J4ME que a versão 1.0 não tem (a *framework* J4ME será analisada na secção dedicada à interface). O perfil será o MIDP 2.0 já que, para além de todas as razões enunciadas anteriormente, a grande maioria dos telemóveis actualmente disponibilizados suporta esta versão do perfil e porque a *framework* J4ME exige que os dispositivos alvo a suportem. Não será usado MIDP 2.1 uma vez que não há muitos dispositivos que o suportem e porque as melhorias introduzidas com o mesmo não serão necessárias para a implementação das aplicações em causa.

3.1.2. Servidor

A camada de lógica de negócio encontra-se implementada em PHP 5. O PHP é uma linguagem orientada a objectos (desde o PHP 3) cujo código executa do lado do servidor Web e é amplamente usada actualmente no desenvolvimento de páginas Web dinâmicas. É uma linguagem *open-source* cuja primeira versão foi disponibilizada em 1994. Existem diversas versões da mesma, sendo a última *release* a 5.2.6.

Dado que existe uma lógica de negócio já implementada no site Infovini (em PHP) que devolve os resultados necessários à aplicação, há todo o interesse em reutilizá-la até

porque essa é uma das vantagens e objectivos de uma arquitectura em camadas. A arquitectura do sistema deve ser pensada de modo a permitir a interoperabilidade de aplicações Java (do lado do cliente) com classes PHP (do lado do servidor).

3.2. Arquitectura do portal Infovini

A arquitectura do portal Infovini encontra-se organizada em camadas. A interface do portal Infovini (camada superior) encontra-se estruturada em blocos que podem ser componentes base como menus, *banners* e também módulos e itens, sendo estes últimos compostos por código PHP, HTML e JavaScript. A disposição visual dos blocos é definida por templates associados a cada página.

Os módulos encontram-se na pasta "modulos" e são blocos que se podem encontrar em diferentes secções do portal. Um exemplo de um módulo é o de *login*, que está presente em todas as áreas de forma a permitir a autenticação do utilizador em qualquer página. Outros estão presentes apenas em determinadas páginas como é o caso do módulo "links relacionados" que permite aceder directamente a outras páginas do Infovini com informação relacionada com aquela que se está a consultar.

Os itens encontram-se na pasta "versaoHTML" e são mais específicos de cada secção, relacionando-se directamente com a página acedida. Exemplos disso são as páginas informativas relativas às videiras, castas, práticas culturais, doenças e pragas (acessíveis a partir de Produzir → Viticultura).

A informação sobre que módulos e que itens deverão aparecer em cada zona do portal está armazenada na base de dados (camada inferior). As chamadas às funções da lógica de negócio (camada intermédia) são feitas fundamentalmente a partir das duas pastas anteriormente referidas ("modulos" e "versaoHTML") e também da pasta "common", mais particularmente do ficheiro `ajax.php`. Os acessos feitos a partir deste ficheiro estão relacionados com as zonas do portal que envolvem AJAX. Um exemplo disso é a zona de pesquisa de distribuidores (acessível a partir de Vender → Distribuidores), onde existem duas *combo boxes*, uma para o país e outra para a cidade onde esta última é actualizada com uma lista de cidades, depois de escolhido o país.

As páginas, tanto a inicial como as outras (acedidas sempre a partir do mesmo ficheiro – `pagina.php`), também acedem directamente à camada de lógica de negócio. A primeira, `index.php`, acede apenas para incluir a classe "Utilizador" (é necessário fazê-lo antes da chamada a `session_start()`, já que se mantém em sessão um objecto correspondente ao utilizador autenticado) e para saber que destaques existem para mostrar. `Pagina.php`, além de incluir a classe "Utilizador", também acede à camada de lógica de negócio para adicionar à base de dados o registo da visita e para aceder a informações detalhadas sobre a página à qual se está a aceder.

Na Fig. 18 pode ver-se o diagrama de camadas do portal Infovini. Apenas as pastas mais relevantes foram incluídas no diagrama. As pastas de imagens, objectos Flash, folhas de estilos (entre outras) foram omitidas por não serem relevantes neste contexto.

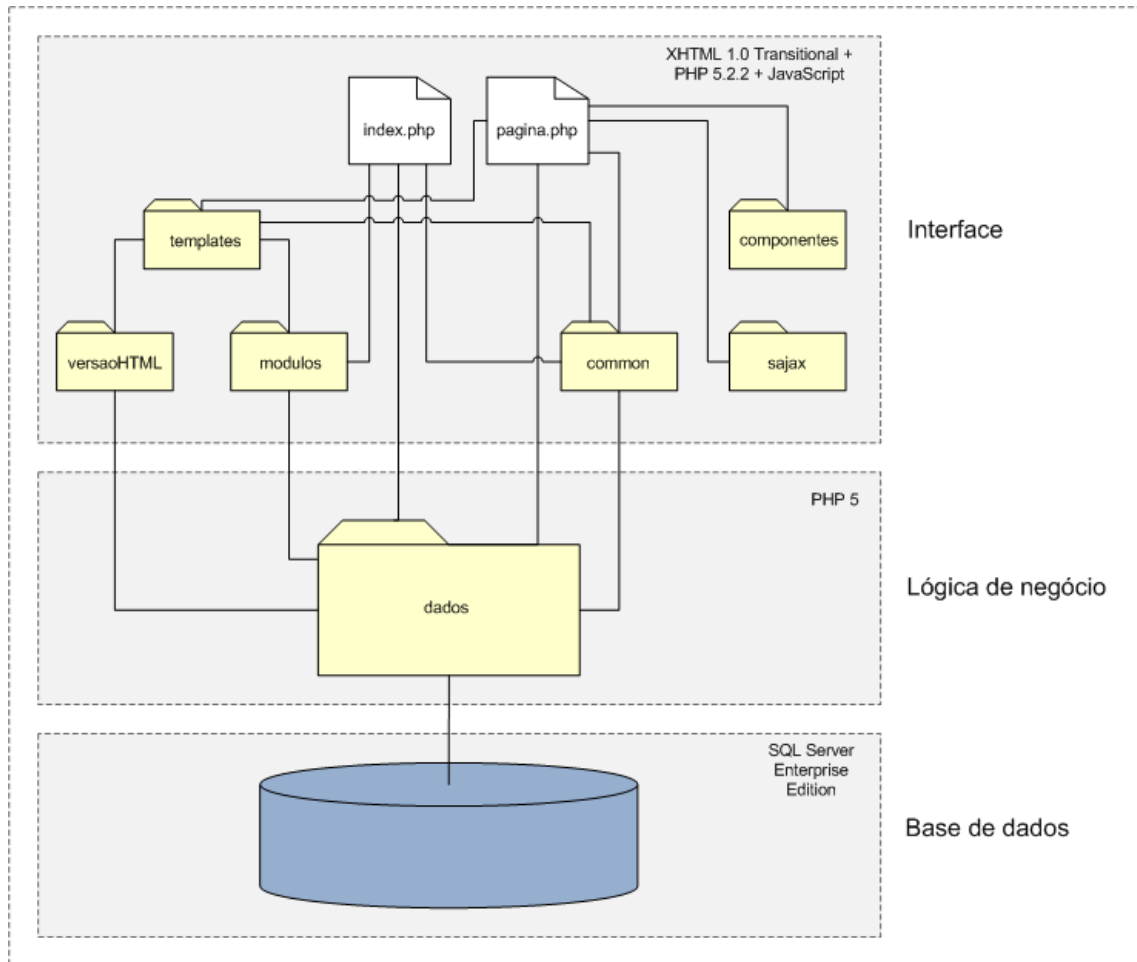


Fig. 18: Diagrama de camadas do portal Infovini.

`index.php` é a página inicial do site, enquanto que `pagina.php` se encarrega de tratar todas as outras páginas. Esta última recebe por HTTP GET um código correspondente à página à qual o utilizador está a tentar aceder e invoca as classes da lógica de negócio passando-o como argumento, de forma a obter algumas informações da página em causa e saber que *template*, módulos e itens deverá mostrar. Posto isto importa o *template* correspondente e chama a função

```
function applyTemplate($pagina, $itens, $modulos)
```

presente em todos os ficheiros de *templates* existentes. Esta função recebe como argumentos um *array* com as informações relativas à página em questão, assim como dois *arrays* com os itens e módulos que estarão presentes na página, respectivamente.

Os *templates* resumem-se à função `applyTemplate` cujo cabeçalho foi mostrado em cima e contém o código HTML de criação de tabelas (o *layout* do site está estruturado por tabelas) e posicionamento do *banner*. Chamam também as funções auxiliares contidas em `common/utills.php` encarregues de apresentar os módulos e itens da página em questão, que foram obtidos previamente em `pagina.php` e recebidos como argumentos.

A pasta `common` contém funções auxiliares como por exemplo código de conexão à base de dados, ficheiros de *header* e *footer*, funções de uso recorrente, assim como o já referido ficheiro de AJAX.

A pasta `sajax` contém os ficheiros da ferramenta *open-source* Sajax, usada para facilitar o uso da *framework* Ajax.

A pasta `componentes` contém componentes usados em determinadas zonas do *site* como por exemplo comentários ou curiosidades.

3.3. Architecturas adoptadas

Os módulos cliente e servidor das duas primeiras aplicações transferem entre si tipos de dados compostos (*arrays* de *strings*, como se verá adiante) uma vez que é necessário manipular diferentes tipos de informações relativas a vinhos tais como nome, idade e teor alcoólico, entre outras. A terceira aplicação não envolve a transferência de mais do que um tipo de dados simultaneamente, daí que se tenha adoptado uma arquitectura diferente. De seguida são apresentadas as duas soluções arquitecturais escolhidas.

3.3.1. Arquitectura de “Uma Ocasão, um Vinho” e “Pesquisa de Vinhos”

Como já foi referido, do lado do servidor será aproveitada a camada de lógica de negócio. A solução proposta deverá ser capaz de garantir a interoperabilidade entre máquinas já que enquanto de um lado se tem uma camada de lógica de negócio implementada em PHP, disponibilizada numa máquina Windows por um servidor IIS, do outro tem-se um cliente Java ME a correr num dispositivo com um de muitos sistemas operativos possíveis (Windows Mobile, SymbianOS ou Palm OS, por exemplo). O objectivo é portanto introduzir uma nova camada por cima da lógica de negócio e ao mesmo nível da camada de interface do portal que funcione como um *interface* para as aplicações Java ME. Uma arquitectura orientada a serviços (SOA) possibilita a interoperabilidade entre diferentes tecnologias. Os Web Services são usados para implementar uma arquitectura SOA para comunicação sobre protocolos *standard* da Internet como HTTP. De seguida é feita uma breve abordagem a este tipo de sistemas, à linguagem XML e aos diferentes estilos de Web Services.

Web Services

Os Web Services são sistemas desenhados para suportar interacção entre máquinas de forma “interoperável” numa rede. O formato usado nas mensagens é normalmente o XML. As arquitecturas baseadas em Web Services constituem uma solução para este tipo de situações em que se pretende que máquinas com plataformas diferentes comuniquem numa rede. Os Web Services podem ser usados de várias formas, sendo as mais comuns XML-RPC⁴⁹, SOAP e REST. Quando é necessária a geração automática de código do lado do cliente, deve existir do lado do servidor um documento WSDL⁵⁰ que contém a descrição das operações suportadas pelo Web Service e que é consultado pelo cliente para a geração de *stubs*. Poderá ainda existir

⁴⁹ RPC de “Remote Procedure Call”.

⁵⁰ Web Service Definition Language.

um registo universal como o UDDI onde se publicam os serviços. Este registo poderá ser depois usado por clientes que procurem serviços de um determinado tipo, retornando-lhes a localização dos documentos WSDL. Pode ver-se um esquema deste funcionamento na Fig. 19.

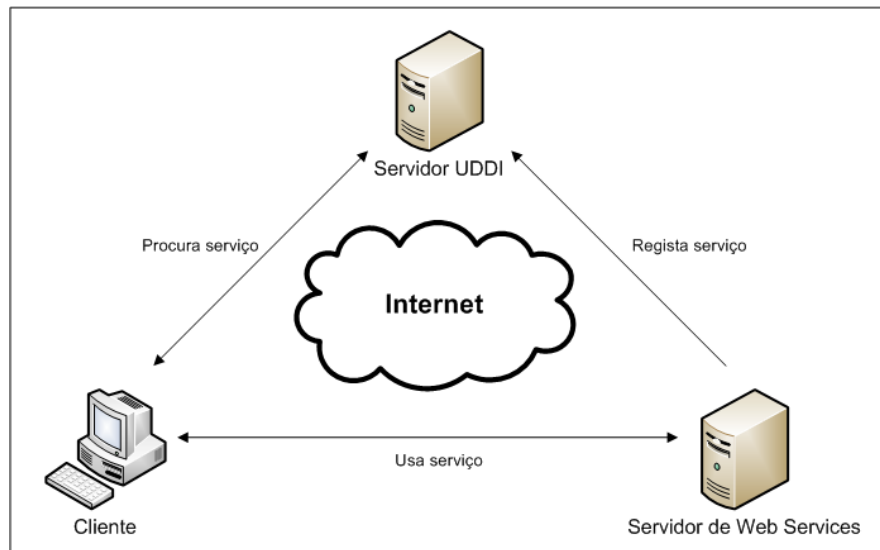


Fig. 19: Utilização do UDDI na descoberta de serviços.

De seguida é feita uma breve abordagem à linguagem XML, às especificações XML-RPC e SOAP e ao estilo REST.

3.3.1.1. Linguagem

XML

XML é uma recomendação do W3C para geração de linguagens de *markup* (linguagens cujo texto descreve a sua estrutura e formatação). É considerada extensível já que permite aos utilizadores definir os seus próprios elementos. A estrutura é facilmente inteligível e baseia-se em *tags*. Para um documento ser considerado bem formado necessita de seguir algumas regras básicas de sintaxe como ter um elemento raiz, ter uma *tag* de fecho para cada *tag* de abertura, entre outras bastantes simples. Este formato pode representar estruturas de dados comuns como listas ou árvores. Ao contrário dos formatos binários, o XML é baseado em texto e desta forma legível pelo programador (ou por outra pessoa) o que permite saber ao certo que dados estão a ser transmitidos e facilita tarefas de *debugging*. O *parsing* é também simples e pouco exigente a nível de processamento quando comparado com outras linguagens de *markup* como SGML⁵¹.

Entre as principais desvantagens destacam-se o facto de ser redundante quando comparada com representações binárias. A sintaxe é considerada “verbosa” em comparação com outros formatos não extensíveis.

⁵¹ Standard Generalized Markup Language.

Considera-se que esta deverá ser a solução adoptada já que além da sua extensibilidade e baixa exigência ao nível do processamento das mensagens, existe no mercado uma vasta gama de bibliotecas capazes de processar XML por ser uma recomendação W3C. É a linguagem mais utilizada quando se pretende garantir a interoperabilidade entre aplicações na Internet, o que vai também de encontro à intenção de facilitar o eventual desenvolvimento de outras aplicações que façam uso dos Web Services.

3.3.1.2. Estilos

XML-RPC

O XML-RPC foi originalmente criado em 1998. Este protocolo é baseado em Remote Procedure Calls e é mais simples em comparação com SOAP, que será abordado a seguir. Permite a transferência de tipos básicos, estruturas, *arrays* e código binário (codificado em Base64). Não suporta a atribuição de nomes, de forma que os parâmetros são passados por posição, o que exige algum cuidado por parte do programador relativamente à sua ordem. Não permite a passagem de tipos de dados definidos pelo utilizador ou de objectos. A especificação final XML-RPC data de Junho de 2003, não tendo sofrido alterações desde então. Com a adição de novas funcionalidades, o XML-RPC veio a dar origem ao SOAP.

SOAP

O SOAP é um protocolo para comunicação por Web Services baseado no estilo SOA⁵². Foi originalmente criado em 1998 por uma equipa com o apoio da Microsoft e é actualmente a especificação W3C para comunicação orientada a serviços e o protocolo mais amplamente adoptado para este tipo de interacção. É mais robusto e mais complexo em comparação com XML-RPC e neste caso os parâmetros são passados por nome. É incluída informação detalhada sobre cada campo e é possível o envio de tipos de dados definidos pelo programador. Este protocolo pressupõe a existência de um “envelope” que é o elemento raiz da mensagem e que define o *namespace* (que identifica o *envelope* como um *SOAP envelope*) assim como o estilo de codificação (*encodingStyle*) que consiste num URL de um *schema* que indica as regras de serialização de partes da mensagem. Um envelope contém obrigatoriamente um elemento *body* onde vai residir o corpo da mensagem, ou seja, onde os campos/parâmetros são incluídos. Pode opcionalmente conter um *header* e um elemento contendo eventuais mensagens de erro. A principal desvantagem ao nível da *performance* em relação ao XML-RPC é incluir um *overhead* superior no tamanho das mensagens devido ao facto de, como referido, incluir um elemento *envelope* e informação mais detalhada acerca de cada campo. A especificação SOAP tem sofrido alterações ao longo do tempo, sendo a última versão (SOAP 1.2) datada de Abril de 2007.

REST⁵³

Este estilo arquitectural foi originalmente proposto em 2000 por um dos principais criadores do protocolo HTTP, Roy Fielding. É baseado no protocolo HTTP, mais particularmente nos métodos POST, GET, PUT e DELETE.

⁵² Service-Oriented Architecture.

⁵³ Representational State Transfer.

Estes quatro métodos podem ser associados a funções de selecção (no caso do GET) ou de modificação (no caso dos outros três) e é nesse pressuposto que assenta a ideia do REST. Para se implementar uma solução deste tipo, é necessário em primeiro lugar saber quais são os recursos que se pretende disponibilizar. Posteriormente é preciso definir-se o formato. Por exemplo, se se tiver definido como recurso um vinho, um pedido GET deverá retornar informação sobre o mesmo, num determinado formato, por exemplo XML. Se se pretende adicionar um vinho, então o método HTTP a usar deverá ser o POST. A Tabela 1 mostra uma possível associação entre os métodos HTTP e as operações básicas das bases de dados.

Tabela 1: Métodos HTTP e REST.

Métodos HTTP	Operações básicas de bases de dados
GET	Seleção
POST	Inserção
PUT	Modificação
DELETE	Remoção

A decisão sobre o uso de cada método fica ao cargo do programador, no entanto é sugerida esta lógica. As operações GET devem sempre ser seguras, ou seja, não devem implicar qualquer alteração do lado do servidor.

No caso das aplicações para dispositivos móveis que se pretende implementar, os pedidos basear-se-iam apenas em GET já que se necessita apenas de obter dados provenientes do servidor. Uma grande vantagem do REST é não necessitar de bibliotecas adicionais para se implementar, já que se baseia unicamente no protocolo HTTP.

3.3.1.3. Conclusão

As três soluções referidas são baseadas em texto. Uma solução binária seria logicamente mais rápida ao nível do transporte mas traria complexidade tanto às mensagens trocadas como ao código. O protocolo adoptado foi o SOAP por vários motivos. O principal teve a ver com o facto de ser uma recomendação W3C (ao contrário de XML-RPC e REST). Os objectivos fundamentais do Infovini são a promoção e a divulgação do sector vitivinícola e é por isso vantajoso criar uma plataforma baseada em *standards* que facilite o desenvolvimento de outras aplicações que façam uso dos serviços de forma concordante com os mesmos. Além disso, a maior complexidade relativamente a XML-RPC é transparente para o utilizador e para o programador que desenvolve as aplicações ou que no futuro as modifica/melhora pois existem actualmente bibliotecas (faladas mais adiante) tanto para o cliente como para o servidor que tornam bastante simples o processamento de pedidos/respostas em SOAP. Os *arrays* retornados contendo a informação necessária sobre cada vinho ocuparão em média 418 *bytes* ou 3344 *bits* (este valor inclui *tags* de abertura e fecho e foi obtido calculando a média dos tamanhos ocupados por 30 *arrays* de 30 vinhos aleatórios), o que é um valor baixo tendo em conta as velocidades típicas dos telemóveis

actuais de 32 - 48 *kbps* por GPRS, por exemplo [Mob08]. Além das razões enunciadas, o facto de permitir evoluções futuras se se decidir melhorar as aplicações de forma a incluir objectos por exemplo, pesou favoravelmente na escolha final em relação às outras duas alternativas.

Na Fig. 20 é apresentado um diagrama com a solução proposta.

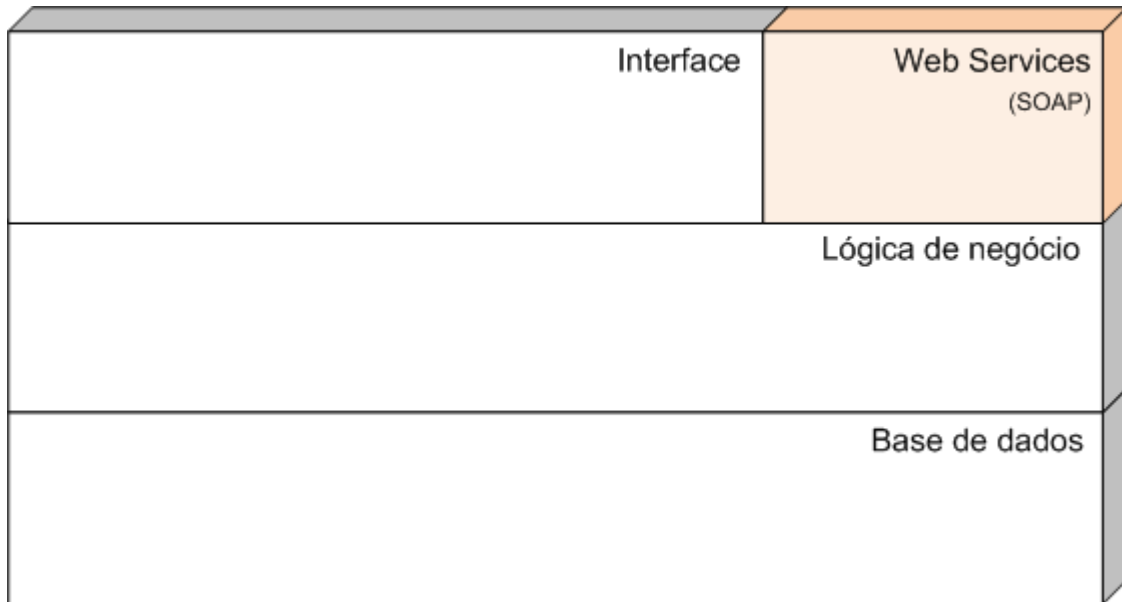


Fig. 20: Diagrama de camadas adoptado, com Web Services.

Tal como referido, a camada adicional relativa aos Web Services estará colocada ao nível da camada da interface do portal, logo acima da camada de lógica de negócio.

3.3.2. “Rotas dos Vinhos”

A última aplicação a desenvolver no âmbito do projecto, relativa à apresentação de percursos turísticos em suportes móveis, tem uma arquitectura diferente das duas anteriores e usa a Google Static Maps API, uma API disponibilizada pela equipa da Google que difere da API dinâmica por poder ser usada em ambientes sem suporte a JavaScript, retornando apenas imagens (daí a designação “Static”) de mapas. Será feita uma breve abordagem à mesma e de seguida será mostrada a arquitectura usada no contexto da aplicação.

3.3.2.1. Google Static Maps API

De forma a permitir a visualização de imagens de mapas em páginas Web sem a necessidade de suporte a JavaScript, a equipa da Google Maps disponibilizou uma API que devolve uma imagem de acordo com um conjunto de características especificadas no próprio URL. Os parâmetros são os seguintes:

- `center` (obrigatório se não existirem marcadores nem caminhos): corresponde ao centro do mapa, equidistante das bordas. Recebe dois valores, separados por vírgulas correspondentes à latitude e longitude. Será usado no contexto da aplicação e variará consoante o deslocamento (em relação às coordenadas iniciais) que o utilizador decidir

aplicar ao mapa. Cada valor de deslocamento variará também consoante o nível de *zoom* no momento.

- `zoom` (obrigatório se não existirem marcadores nem caminhos): corresponde ao nível de *zoom* que pode variar entre 0 e 19. Será usado no âmbito do desenvolvimento da aplicação, permitindo ao utilizador aumentar ou diminuir o seu nível, aproximando-se ou afastando-se do mapa.
- `size` (obrigatório): corresponde ao tamanho, em pixéis, da imagem (largura e altura). Será usado de forma a adaptar o tamanho das imagens ao ecrã dos dispositivos.
- `format` (opcional): corresponde ao formato da imagem pretendido, que poderá ser GIF (usado por omissão), JPG ou PNG. Será utilizado no contexto da aplicação já que poderão ser usadas imagens JPG ou PNG, dependendo da capacidade do dispositivo de decodificar JPG (a especificação MIDP 2.0 só exige que estes suportem o formato PNG, no entanto, caso o dispositivo consiga, é proveitoso que sejam usadas imagens em JPG devido ao nível de compressão ser superior e desta forma fazer com que a quantidade de dados transferida seja menor).
- `maptype` (opcional): poderá ser “roadmap” (por omissão), apresentando o mapa tal como apareceria na versão dinâmica do Google Maps. Poderá também escolher-se o tipo “mobile” e nesse caso a mesma imagem é mostrada com ligeiras alterações de forma otimizada para ecrãs mais pequenos. Será portanto usado no âmbito da aplicação o tipo “mobile”.
- `markers` (opcional): utilizado para definir marcadores que se podem colocar sobre o mapa. Cada marcador é caracterizado pela sua latitude, longitude, tamanho, cor e letra do alfabeto (A-Z). O número máximo de marcadores permitido por imagem é de 50. Este parâmetro será utilizado para a representação dos pontos de interesse.
- `path` (opcional): usado para mostrar caminhos compostos por dois ou mais pontos, até um máximo de 50. Cada caminho é composto por um modelo de cor (ou RGB ou a sua variante RGBA com *alpha channel*), código hexadecimal da cor pretendida, largura do traço e latitude/longitude de cada um dos pontos. Será usado para especificar os caminhos.
- `hl` (opcional): corresponde ao idioma usado nos *labels* dos quadrados que compõem o mapa. Não será usado no contexto da aplicação uma vez que os *labels* das regiões envolvidas encontram-se apenas em português.
- `key` (obrigatório): chave fornecida pela Google Maps API aquando do registo. O seu uso é obrigatório por isso será usada.

A Google Static Maps API impõe um limite de 1000 visualizações de mapas por IP, por cada 24 horas. Se o limite for excedido, é imposto um bloqueio temporário ao IP e se houver reincidência, o bloqueio passa a ser permanente. Caso não existisse este limite, o dispositivo móvel poderia apenas fazer o pedido à camada de lógica de negócio do Infovini, que se encarregaria de criar o URL e de seguida fazer ela própria o pedido ao servidor do Google Static Maps para depois reencaminhar a imagem para o aparelho. No entanto esta abordagem foi abandonada (parcialmente, como se verá a seguir) pois desta forma cada utilizador do serviço ficaria limitado ao número de imagens que os outros utilizadores já

tenham solicitado, já que o IP que ficaria registado na base de dados do Google Static Maps seria sempre o do servidor. Assim, optou por se desenvolver o sistema de forma a ser o próprio dispositivo a contactar directamente o serviço do Google Static Maps, fazendo com que o limite de 1000 visualizações passe a ser por cada utilizador. A camada de lógica de negócio fica encarregue de todas as tarefas envolvendo a criação do URL que envia ao dispositivo, para que este se encarregue de o abrir.

Esta abordagem funcionou como esperado nos dispositivos mais recentes. No entanto aconteceu uma situação inesperada que teve a ver com o facto de certos dispositivos (mais antigos) não suportarem a abertura de endereços Web demasiado longos (com mais de 600 caracteres em alguns casos, quando normalmente são precisos mais de 1500 dado o elevado número de parâmetros). A única forma de passagem de argumentos na Google Static Maps é no próprio URL, de forma que esta situação teve que ser resolvida desenvolvendo o sistema de forma a funcionar de duas maneiras, dependendo da capacidade do dispositivo. A determinação da capacidade de suporte a URL extensos é determinada pela própria aplicação. Caso não exista, a aplicação indica-o no pedido que faz à camada de lógica de negócio e apenas neste caso o procedimento PHP contacta directamente o serviço do Google Static Maps e reencaminha a imagem para o aparelho. Uma vez que é imperativo evitar que o limite de visualizações seja ultrapassado, foi necessário fazer *logging* de cada pedido que é feito e da respectiva data, de forma a, nos pedidos subsequentes, se poder verificar se o limite está prestes a ser atingido e se possa decidir se se vai contactar o serviço do Google Maps ou retornar uma mensagem a indicar que tal não é possível. O funcionamento em ambos os casos poderá ser consultado na Fig. 22 e na Fig. 23.

3.3.2.2. Arquitectura de “Rotas do Vinho”

A aplicação relativa à apresentação de percursos turísticos não envolve, em nenhum momento, a transferência de tipos de dados compostos. Tal como está especificado nos casos de uso, esta aplicação necessita apenas de transferir *strings* relativas a descrições de alguns pontos de interesse que não se encontram armazenados localmente, *strings* relativas a URL da Google Static Maps API e imagens (mapas de itinerários, mapas de locais e fotografias). Tipos de dados simples podem ser facilmente recebidos a partir de pedidos HTTP sem necessidade de protocolos adicionais e da inclusão de camadas suplementares sobre a lógica de negócio. Desta forma, optou por se criar quatro procedimentos em PHP acessíveis da mesma forma que se acede a uma página Web (pode ver-se o resultado da invocação a partir de um *browser* Web), mostrados na Fig. 21.

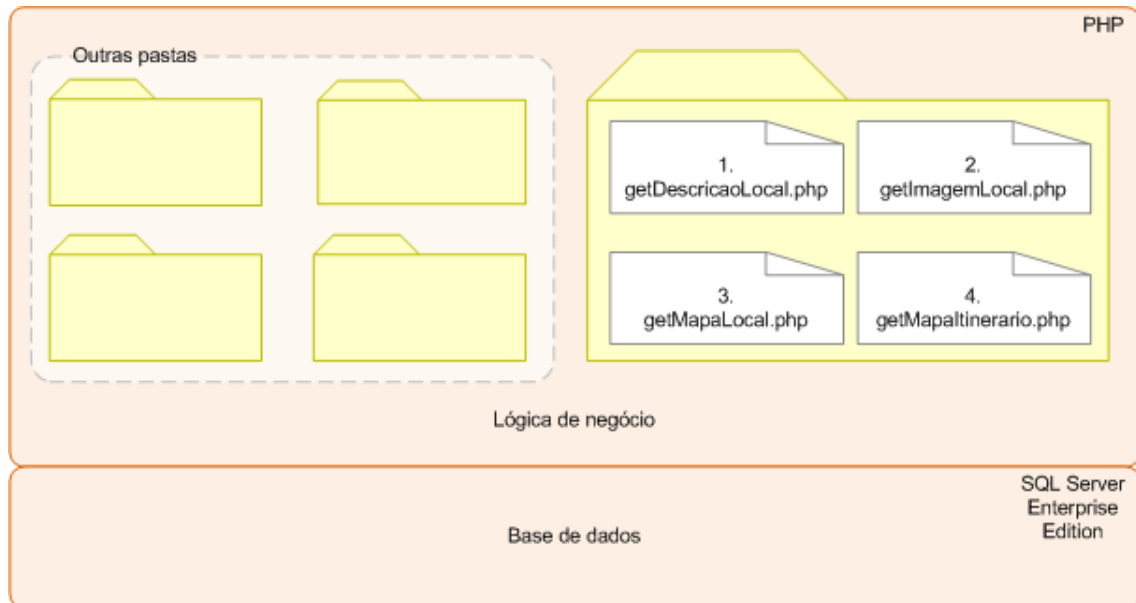


Fig. 21: Diagrama de camadas do servidor para "Rotas do Vinho".

Um deles (marcado com “1”) deverá retornar uma *string* obtida da base de dados descrevendo um local. O procedimento marcado com “2” será responsável por retornar uma imagem de um local (normalmente uma fotografia, tal como aparece no portal quando se clica sobre um ponto de interesse). O terceiro deverá ser responsável pelo envio de um mapa onde apenas aparece o local seleccionado. Esta funcionalidade não existe no portal do Infovini mas achou-se interessante disponibilizar no contexto dos dispositivos móveis já que devido ao facto de a navegação não se fazer tão rapidamente como num computador pessoal, pode tornar-se complicado isolar um determinado local no mapa. Este tem em conta o local escolhido, as dimensões do ecrã do dispositivo, o nível de *zoom*, as coordenadas do centro do mapa e o formato da imagem pretendidos. Uma vez que o URL, no caso de se tratar de um local sem inclusão de caminhos (ou “polylines”) é curto, logo suportado por qualquer dispositivo, não há necessidade neste caso de se desenvolver o sistema de forma a eventualmente ser o próprio servidor a contactar directamente o serviço do Google Static Maps. O último procedimento é responsável por criar e enviar um URL do Google Static Maps API tendo em conta o itinerário escolhido e tal como o anterior, as dimensões do ecrã do dispositivo, o nível de *zoom*, as coordenadas do centro do mapa e o formato da imagem pretendidos. Pode alternativamente enviar a própria imagem, caso o dispositivo não seja capaz de abrir endereços Web demasiado longos.

Em suma, uma vez que as aplicações comunicam directamente com a camada de lógica de negócio, a arquitectura do lado do servidor envolve apenas as camadas de base de dados e lógica de negócio. De notar que esta última, caso o dispositivo não seja capaz de abrir URL extensos, comunica ela própria com o servidor da Google Static Maps API, recebendo a imagem e reencaminhando-a para o dispositivo. Na Fig. 22 e na Fig. 23 podem ver-se os diagramas do funcionamento do sistema, dependendo do suporte à abertura de URL longos por parte da implementação do Java ME do dispositivo móvel.

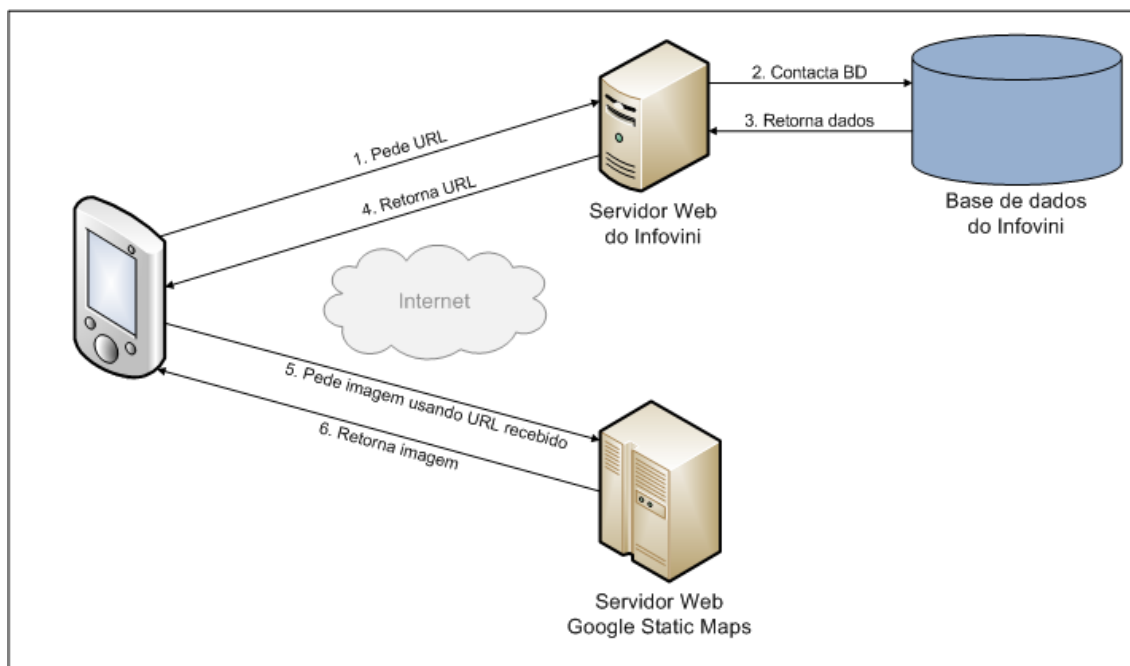


Fig. 22: Diagrama da comunicação de "Rotas do Vinho" para apresentação de itinerários (dispositivo capaz de abrir URL longos).

Como se pode ver, no caso de o dispositivo (aqui representado como um PDA) suportar a abertura de URL extensos, recebe-o do servidor Web do Infovini e abre-o, contactando assim directamente o servidor do Google Static Maps.

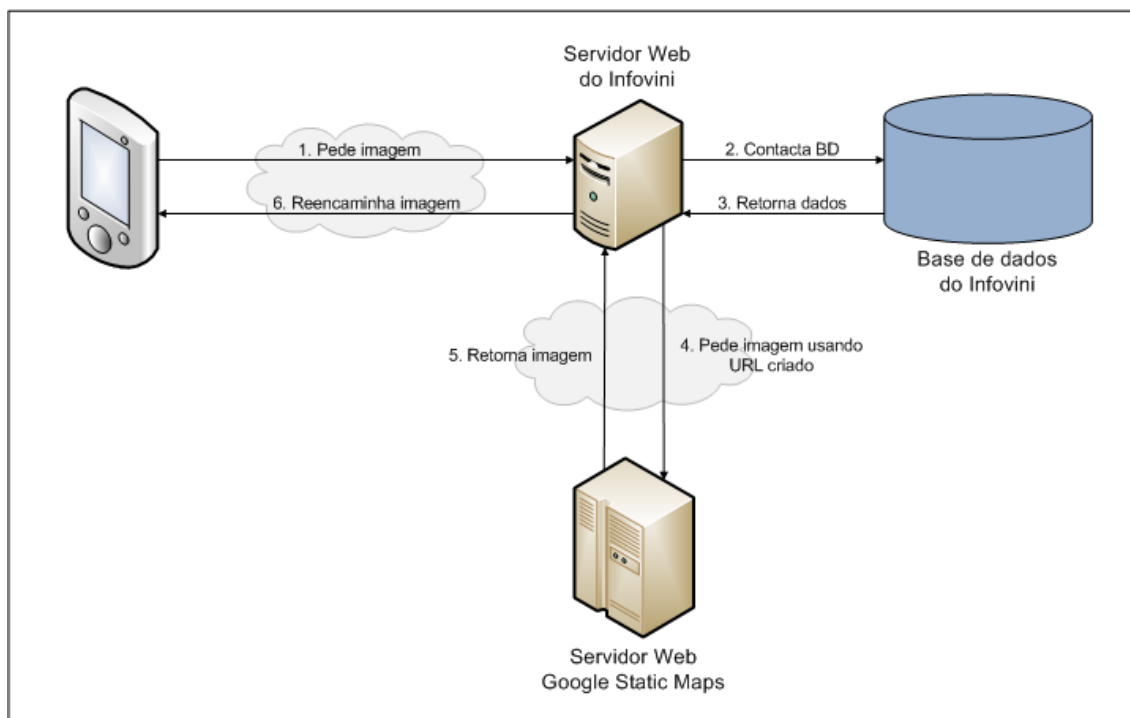


Fig. 23: Diagrama da comunicação de "Rotas do Vinho" para apresentação de itinerários (dispositivo incapaz de abrir URL longos).

Tal como na imagem acima, no caso de não suportar URL demasiado extensos, o dispositivo faz apenas um pedido ao servidor do Infovini e recebe a imagem do mapa. Toda a comunicação envolvendo o servidor do Google Static Maps se dá de forma transparente do ponto de vista da aplicação.

Importa salientar que os diagramas anteriores apenas mostram a comunicação entre dispositivos móveis, servidores do Infovini e do Google Static Maps e base de dados. Todas as operações envolvidas tanto do lado da aplicação como do lado do servidor são omitidas nesta fase e abordadas no capítulo 4, dedicado à implementação.

3.4. Tecnologias adoptadas

Foram analisadas algumas ferramentas viáveis no âmbito do que tinha ficado definido em relação às tecnologias e arquitecturas a utilizar. Nesta secção é feita uma abordagem às ferramentas analisadas e feita justificação das escolhas de umas em detrimento de outras. Importa referir que a subsecção dedicada às implementações de SOAP é apenas referente às aplicações “Uma Ocasão, um Vinho” e “Pesquisa de Vinhos” pois, como já foi dito, “Rotas do Vinho” não envolve Web Services.

3.4.1. Ambiente de desenvolvimento

O ambiente de desenvolvimento escolhido foi a versão completa do Netbeans 6.0.1, a última estável na altura de escrita deste relatório (a versão 6.1 já se encontra disponível mas como *Beta release*). Genericamente o Netbeans é um IDE⁵⁴ Java *open-source* criado pela Sun Microsystems que disponibiliza um conjunto de funcionalidades bastante úteis e recorrentes nos IDE actuais tais como *refactoring*, controlo de versões, navegador de ficheiros do projecto e de classes, entre outras. A versão escolhida inclui o Wireless Toolkit 2.5.2 que é um conjunto de ferramentas que facilita a criação de aplicações Java ME para dispositivos móveis. Entre as funcionalidades mais interessantes, destacam-se o emulador (que permite reduzir muito o tempo necessário para testes), a possibilidade de testar a compatibilidade do código produzido com CLDC 1.0/1.1 e MIDP 1.0/2.0/2.1, a possibilidade de adicionar emuladores de terceiros e também uma ferramenta de “obfuscation” que permite otimizar o tamanho das aplicações compiladas e dificultar acessos não autorizados ao código fonte de forma a prevenir a violação da propriedade intelectual do mesmo (apesar de que neste caso o principal propósito do uso desta ferramenta é reduzir o tamanho das aplicações). É também possível definir o nível de *obfuscation* desejado. Devido aos recursos limitados dos dispositivos móveis, reduzir o tamanho das aplicações é importante, de forma que esta ferramenta se reveste de uma grande utilidade.

O Eclipse é um outro IDE que em conjunto com o *plugin* para desenvolvimento para dispositivos móveis *eclipseME*⁵⁵, integra muitas das funcionalidades referidas anteriormente. Apesar de no geral a maior parte das discussões encontradas sobre que IDE é mais adequado para o desenvolvimento de aplicações Java ME dar vantagem ao Netbeans, a escolha do

⁵⁴ Integrated Development Environment.

⁵⁵ Disponível em <http://eclipseme.org>.

mesmo não se baseou na comparação de funcionalidades existentes nestes dois ambientes de desenvolvimento mas sim no facto de o NetBeans permitir fazer tudo aquilo que se pretende sem necessidade de configurações exaustivas ou instalação de *plugins* – após a instalação da versão pretendida, pode dar-se início ao desenvolvimento.

3.4.2. Implementações de SOAP

À medida que o protocolo SOAP se foi generalizando, foram surgindo diversas implementações do mesmo para as diferentes linguagens de programação. No caso das aplicações “Um Ocasão, um Vinho” e “Pesquisa de Vinhos” pretende-se adoptar uma implementação SOAP para PHP (para o servidor) e outra para Java ME (para o cliente). Segue-se uma análise das ferramentas estudadas para usar no âmbito destas duas aplicações.

3.4.2.1. Servidor

NuSOAP

O NuSOAP é um conjunto de classes desenvolvidas pela NuSphere e Dietrich Ayala que permite a implementação de Web Services em PHP e que pode ser usada sob os termos da licença LGPL. Para usar esta implementação de Web Services basta incluir o ficheiro `nusoap.php` que contém as classes necessárias. É uma solução amplamente usada por programadores que desejam implementar Web Services em PHP e surgiu como uma das soluções quando o PHP não suportava nativamente SOAP (a outra era o PEAR::SOAP, falado mais adiante neste secção). Contém algumas funcionalidades úteis para *debugging* como por exemplo variáveis do objecto cliente que contêm as mensagens SOAP relativas a pedidos e a respostas.

PHP-SOAP

O PHP-SOAP é uma extensão para PHP que surgiu com o PHP5. Ao contrário do NuSOAP, foi desenvolvida em C (NuSOAP foi escrito em PHP), o que a torna mais rápida. Uma grande desvantagem do PHP-SOAP, principalmente tendo em conta o âmbito deste projecto, é não suportar outras codificações de caracteres que não UTF-8. Grande parte dos caracteres que são passados (nomeadamente as letras acentuadas) teria que ser codificada para UTF-8 antes de ser enviada para posteriormente ser decodificada novamente para a codificação original (neste caso ISO-8859-1) de forma a ser correctamente apresentável. Devido à existência de outras boas bibliotecas de distribuição livre que permitem implementar servidores de Web Services SOAP em PHP, optou por se procurar uma que não precisasse de conversões de caracteres.

PEAR::SOAP

O PEAR::SOAP é uma outra implementação de Web Services usando SOAP que tal como o NuSOAP foi escrita em PHP (o que a torna potencialmente mais lenta que PHP-SOAP). A diferença mais relevante em relação ao NuSOAP tem a ver com o facto de o seu desenvolvimento ter sido mantido mais tempo (a última *release* é de 2007, sensivelmente dois anos depois da última do NuSOAP).

Conclusão

Apesar de existirem melhores perspectivas de desenvolvimento futuro do PHP-SOAP em relação ao NuSOAP (que aparentemente não sofreu melhorias desde a *release* 0.7.1 de Julho de 2005), o facto de obrigar à conversão explícita de cada um dos campos a enviar pelo servidor para UTF-8 (e de consequentemente obrigar o cliente a reverter o processo) fez com que não se adoptasse esta solução. Além disso, a velocidade que se ganharia em ter uma implementação de SOAP escrita em C é considerada desprezável tendo em conta o número de acessos esperados assim como a quantidade de dados a transmitir. Apesar de esta solução não ter sido adoptada, se os dados a transmitir não necessitassem de conversão, esta seria a opção mais adequada por ser teoricamente mais rápida e por ter melhores perspectivas de desenvolvimento futuro (desde que o PHP começou a incluir uma extensão para lidar com Web Services SOAP que o desenvolvimento das anteriores bibliotecas de terceiros foi sendo abandonado). Se no futuro esta extensão for melhorada de modo a permitir a transmissão de caracteres não codificados em UTF-8, deverá ser adoptada. A configuração é tão simples que em poucos minutos será possível ter o servidor completamente funcional.

A outra opção, o PEAR::SOAP, teria sido também uma escolha viável já que é também simples de implementar e o seu desenvolvimento manteve-se durante mais tempo que o desenvolvimento do NuSOAP. Ao nível da *performance*, não se conhecem diferenças entre uma solução e outra. No entanto, nos testes efectuados, o NuSOAP mostrou que pode fazer tudo aquilo que se pretende e praticamente não necessita de configurações (basta copiar a pasta e incluir `nusoap.php` no código do servidor), enquanto que a última *release* do PEAR::SOAP tem três dependências obrigatórias - pacotes PEAR 1.4.0b1 ou superior, PEAR e HTTP_Request⁵⁶.

Pelos motivos enunciados, a implementação de SOAP escolhida para o lado do servidor foi o NuSOAP.

3.4.2.2. Cliente

JSR 172

Do lado do cliente, as duas principais implementações de SOAP para Java ME são o JSR 172 e o kSOAP.

O JSR 172 é a especificação de Web Services em Java ME da Sun Microsystems. Contém dois pacotes opcionais, um para processar XML e uma API para invocação de Web Services remotos que é um subconjunto da especificação JAX-RPC⁵⁷ com algumas diferenças relativamente a esta. A grande desvantagem do JSR 172 reside no facto de não ser suportado por todos os dispositivos com MIDP 2.0. Outra desvantagem desta tecnologia prende-se com o facto de forçar os *stubs* a serem estáticos o que significa que têm que ser previamente criados usando um documento WSDL.

kSOAP

O kSOAP é uma biblioteca *open-source* para clientes Java ME de Web Services. A versão actual é o kSOAP2 que é uma versão melhorada da original. A última *release*, 2.1.1,

⁵⁶ Informação recolhida da página oficial da versão 0.11.0 - <http://pear.php.net/package/SOAP/download/0.11.0>.

⁵⁷ “Java API for XML-based RPC”, que providencia suporte a protocolos RPC baseados em XML.

data de Junho de 2006. Assim como o JSR 172, necessita de bibliotecas de *parsing* de XML, neste caso o kXML, uma solução *open-source* que neste momento se encontra na versão 2.

Para implementar clientes em kSOAP não é necessária a geração de *stubs*, o que reduz a complexidade das aplicações. Também não é necessária a existência de um WSDL do lado do servidor (em JSR 172 é necessário indicar a localização do WSDL para criar o *stub* ou então o processo tem que ser feito de forma manual).

Conclusão

Uma vez que a prioridade principal deste projecto é criar aplicações abrangentes ao maior número de dispositivos possível (conciliando obviamente com as outras prioridades), decidiu optar-se pelo kSOAP pois desta forma são compatíveis com qualquer dispositivo com MIDP 2.0. É também capaz de fazer aquilo que se pretende sem tarefas adicionais como geração de *stubs*.

3.4.3. Interface

Como se sabe, a interface de uma aplicação é preponderante no primeiro contacto que o utilizador tem com a mesma - uma aplicação com uma interface pobre pode vir a fracassar perante outra aplicação com menos funcionalidades mas com um aspecto mais atraente. Pretende-se também que o *layout*, além de ter um aspecto profissional e cativante, seja o mais uniformizado possível, ou seja, não varie muito entre as diferentes marcas/modelos de dispositivos existentes no mercado.

A API MIDP de alto nível do Java ME permite a inclusão de itens básicos como caixas de texto ou grupos de escolha, no entanto o aspecto dos mesmos pode deixar muito a desejar e varia muito de dispositivo para dispositivo, não se tendo controlo total sobre a forma como são apresentados. Foram realizados alguns testes usando estes elementos, tendo sido possível constatar a grande variação existente na forma como os mesmos são mostrados ao utilizador, consoante o dispositivo. De seguida é feita uma análise mais detalhada às API MIDP de alto nível e de baixo nível, assim como a soluções de terceiros encontradas para melhorar a interface de utilizador e torná-la mais uniforme e coerente independentemente do aparelho.

API MIDP de alto/baixo nível

A API de alto nível permite a implementação de aplicações simples onde não haja grande interesse em se ter um domínio absoluto sobre a forma como os elementos são apresentados no ecrã e onde não seja necessário saber quando determinadas teclas são pressionadas. Quando se realizaram os testes iniciais, a API MIDP de alto nível foi usada na definição de menus, campos de texto não editáveis, entre outros. Foi criado um protótipo da primeira aplicação com a correcta transição entre os diferentes menus de escolha da ocasião (relembra-se que a primeira aplicação a implementar consiste na sugestão de vinhos consoante uma ocasião escolhida). Os menus usados neste caso foram *choice groups* do tipo POPUP, disponíveis em MIDP 2.0, que são comparáveis às listas *dropdown* comuns. A interacção com o utilizador funcionava correctamente, no entanto notavam-se grandes variações na forma como os elementos eram apresentados de dispositivo para dispositivo. Na

Fig. 24 pode ver-se um exemplo da grande diferença de aparência existente entre diversas marcas/modelos (neste caso o emulador do WTK, o emulador do Nokia 6230i e o Nokia N80, respectivamente) quando se usa a API de alto nível.



Fig. 24: Diferenças ao nível do *choice group* inicial.

Estes três *screenshots* referem-se aos *choice groups* do tipo POPUP antes de serem seleccionados. Repare-se na diferença de aspecto entre eles, assim como relativamente aos comandos apresentados. Como se pode verificar, a diferença de aparência é abismal, mesmo entre modelos da mesma marca (neste caso Nokia N80 e Nokia 6230i). A API de alto nível foi por isso posta de parte.

A API de baixo nível confere ao programador mais controlo sobre o *layout* e sobre a forma como é feita a interacção com o utilizador. Permite desenhar no ecrã aquilo que se quiser com precisão ao nível dos pixels. Cada ecrã é um objecto da classe `Canvas` e o único método que é necessário implementar é o `paint(Graphics g)`. O argumento recebido é um objecto da classe `Graphics` que permite desenhar aquilo que se quiser no ecrã.

Uma vez que permite ter total controlo sobre a forma como os itens são apresentados, a implementação de classes de UI⁵⁸ sobre a API de baixo nível permitiria resolver os problemas da API de alto nível referidos anteriormente. Equacionou-se desenvolver de raiz um conjunto de classes que simulassem componentes como menus, botões ou caixas de texto, mas a implementação de uma biblioteca desse tipo desviaria dos objectivos do projecto e existem alternativas de distribuição livre disponíveis na *Web* que cumprem os requisitos pretendidos e que têm sido alvo de contínuos melhoramentos. Foram por isso pesquisadas soluções que usassem esta API para melhorar e unificar o aspecto das aplicações em Java ME. As mais relevantes encontradas foram a J4ME⁵⁹, a MWT⁶⁰ e a FIRE⁶¹.

J4ME

O J4ME foi pela primeira vez disponibilizado em Novembro de 2007 e tem sido melhorado ao longo do tempo (a última contribuição data de Maio de 2008). Permite o uso de

⁵⁸ User Interface.

⁵⁹ Java For Me.

⁶⁰ Micro Window Toolkit.

⁶¹ Flexible Interface Rendering Engine.

temas que alteram as cores das aplicações (tanto ao nível do fundo como das letras). A título meramente informativo, o J4ME além de dispor de *frameworks* direccionadas à interface com o utilizador, também dispõe de outras relacionadas com *logging* e GPS⁶², além de bibliotecas de funções úteis que não são incluídas em Java ME (algumas funções de cálculo matemático por exemplo). A *framework* de UI dispõe de todas as classes que se prevê serem necessárias na implementação das aplicações, como por exemplo as relativas a menus, submenus, campos e caixas de texto, imagens, alertas e comandos. Além disso os aspectos dos itens e a interacção das aplicações com o utilizador (como por exemplo as acções despoletadas a partir dos eventos associados a determinadas teclas) podem ser facilmente configurados a partir do código fonte.

Existe um fórum de suporte activo onde os criadores do J4ME (ou outros utilizadores da *framework*) respondem de forma rápida a perguntas efectuadas por programadores que estão a usar este produto.

Na Fig. 25 pode ver-se uma imagem comparativa de três *screenshots* retirados do emulador do WTK, do Nokia N80 e do emulador do Nokia 6230i (por esta ordem), correspondentes à etapa inicial de definição da ocasião.



Fig. 25: *Screenshots* do menu inicial da aplicação de sugestões de vinhos.

Como se pode verificar, o aspecto é muito mais uniforme do que no caso da API de alto nível. É de referir também que as letras no caso do primeiro *screenshot* aparecem com um tamanho reduzido devido ao facto de a imagem ter sido redimensionada de forma a se ajustar à altura das outras. Uma vez que foi esta a *framework* adoptada, no capítulo dedicado à implementação serão apresentados mais capturas de ecrã do emulador do WTK no seu tamanho real.

MWT

Pelos *screenshots* exemplificativos, a *framework* MWT também foi considerada. No entanto, o aspecto dos exemplos no Nokia N80 revelou deficiências que podem ser visualizadas na Fig. 26.

⁶² Global Positioning System.



Fig. 26: Screenshot de um exemplo do site do MWT num Nokia N80.

A última *release* também é relativamente antiga (Março de 2007), o que sugere que o desenvolvimento não foi continuado.

FIRE

Pelos *screenshots* apresentados, a *framework* FIRE também pareceu esteticamente apelativa, mas mais adequada ao desenvolvimento de jogos. Não foram encontrados exemplos de código (apenas algumas aplicações já compiladas para testar) e a documentação é escassa.

Conclusão

Foi adoptada a *framework* J4ME por ser a que, além de dar mais garantias relativamente à continuidade do melhoramento feito pelos seus criadores, também foi considerada esteticamente mais apelativa e mais adequada ao tipo de componentes que se pretendia incluir nas aplicações (menus, comandos, caixas de texto, campos de texto não editáveis, alertas e imagens), permitindo cumprir com o requisito estabelecido de usabilidade das aplicações. Assim, a arquitectura adoptada para os clientes é a esquematizada na Fig. 27.

Especificação

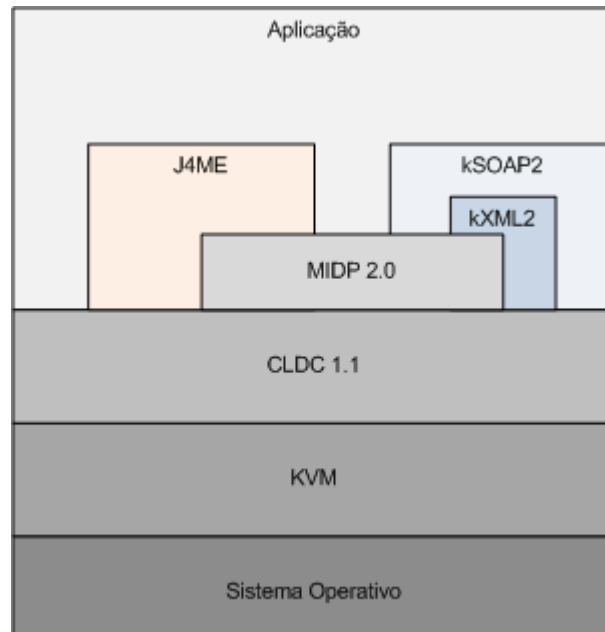


Fig. 27: Arquitectura adoptada em “Uma Ocasão, um Vinho” e “Pesquisa de Vinhos” (lado do cliente).

4. Implementação

Neste capítulo são descritos pormenores de implementação das aplicações e dos módulos mais relevantes do lado do servidor. São também incluídos diagramas de actividades mostrando o funcionamento e a interacção entre o utilizador, clientes e servidores.

4.1. Método de armazenamento local

Em MIDP não existe forma de armazenamento de dados de forma persistente em ficheiros (no sentido em que os dados persistam mesmo terminando a execução da MIDlet), já que os dispositivos por norma não expõem o sistema de ficheiros (caso o possuam) às aplicações MIDP.

A única forma de armazenamento persistente de dados do utilizador é no RMS⁶³, um gestor de bases de dados muito simples que consiste num conjunto de *record stores*. Cada *record store* funciona como uma tabela de duas colunas: um identificador numérico que funciona como chave primária e um campo de dados (*array* de *bytes*). As *record stores* são identificadas pelo seu nome que poderá ter entre 1 a 32 caracteres Unicode. Como referido na secção dedicada ao MIDP 2.0, os requisitos mínimos para um dispositivo que esteja conforme a especificação exigem que reserve 8 *kilobytes* para este tipo de armazenamento (partilhados por todas as aplicações Java ME). A maior parte da informação que interessa manter armazenada localmente é estática, não passível de alterações por parte do utilizador como sugestões genéricas de vinhos, nomes de castas, nomes e descrições de pontos de interesse, entre outras, de forma que não será necessário guardá-la em *record stores*. Estes dados serão mantidos no código das próprias aplicações em classes criadas para o efeito, já que não faria sentido ocupar o espaço reservado à memória não volátil com este tipo de informações estáticas que nem são partilhadas entre MIDlets. No entanto, existem algumas informações que têm a ver com as características do dispositivo ou preferências do utilizador que convém armazenar no RMS, que são as seguintes:

Aplicação “Uma Ocasão, um Vinho”

- informação sobre o suporte a imagens JPG.

Aplicação “Pesquisa de Vinhos”

- informação sobre o suporte a imagens JPG.
- ordenação preferida dos resultados de pesquisa.

Aplicação “Rotas do Vinho”

- informação sobre o suporte a imagens JPG.
- informação sobre o suporte a endereços Web extensos.

⁶³ Record Management System.

Todas as aplicações podem manipular imagens oriundas da rede (rótulos no caso das duas primeiras e fotografias e imagens de mapas no caso da terceira). Apesar de a especificação MIDP 2.0 exigir apenas que os dispositivos suportem imagens no formato PNG, muitos deles suportam também outros formatos como JPG. Devido às limitações no que diz respeito à largura de banda disponível e ao facto de a quantidade de dados transferidos poder estar a ser taxada, interessa minimizá-la. O formato JPG envolve normalmente uma maior compressão relativamente a PNG⁶⁴ por isso, caso o dispositivo o suporte, é vantajoso usar esse formato. O suporte ao mesmo só pode ser avaliado após a primeira execução das aplicações, podendo então o resultado ser armazenado em *record stores* para futuras execuções. No caso da terceira aplicação, importa também armazenar a capacidade do dispositivo de abrir endereços Web muito longos depois da primeira vez em que tenha tido a necessidade de os abrir, para saber se deve contactar o servidor do Infovini no sentido de pedir um URL ou a própria imagem do mapa de um itinerário. Na aplicação de pesquisa de vinhos interessa dar ao utilizador a oportunidade de decidir qual a ordenação dos resultados desejada e guardá-la para futuras execuções, sendo o RMS o único meio para o garantir.

Resumindo, na sua primeira execução as aplicações verificam se já existem dados armazenados nos *record stores* correspondentes. Se não existirem, ao longo da execução decidem sobre o que lá devem armazenar.

4.2. Características comuns

Existem algumas características comuns nas aplicações que foram desenvolvidas. De forma a não repetir as mesmas informações à medida que se aborda a implementação de cada uma delas, essas características são referidas apenas nesta secção.

Todas as aplicações envolvem eventualmente a transferência de imagens da rede. Uma vez que, como já foi referido, interessa que as imagens ocupem o mínimo possível, opta-se sempre que possível pela transferência de imagens no formato JPG (em vez de PNG, o único formato cujo suporte é obrigatório em MIDP 2.0). De forma a averiguar se um dispositivo é capaz de decodificar esse formato, é feito um teste na primeira execução de cada aplicação com uma imagem JPG de pequenas dimensões incluída no próprio ficheiro JAR. O resultado do teste é guardado num *record store* acedido das próximas vezes que a aplicação for iniciada, não havendo por isso necessidade de realizar mais testes.

Os tamanhos dos ecrãs são muito variáveis. Podem ser de 96x54 pixéis (dimensões mínimas em MIDP 2.0) assim como 352x416 (no caso do Nokia N80) ou 208x208 (no caso do Nokia 6230i), entre outros. Sentiu-se por isso a necessidade de adaptar o tamanho da fonte usada ao tamanho dos ecrãs. Em MIDP 2.0 podem usar-se três tamanhos de fontes (pequena, média e grande). Nas três aplicações criadas são usadas as fontes pequena ou média para os textos dos itens de menu e descrições consoante as dimensões do ecrã do dispositivo, avaliadas aquando do início da execução. Na Fig. 28 pode ver-se um diagrama de actividades mostrando o fluxo de acções comuns às três aplicações desenvolvidas.

⁶⁴ Em alguns testes efectuados o formato JPG fazia com que as imagens ocupassem 25% do tamanho das mesmas em PNG, com um nível de compressão que não permitia notar diferenças entre ambas.

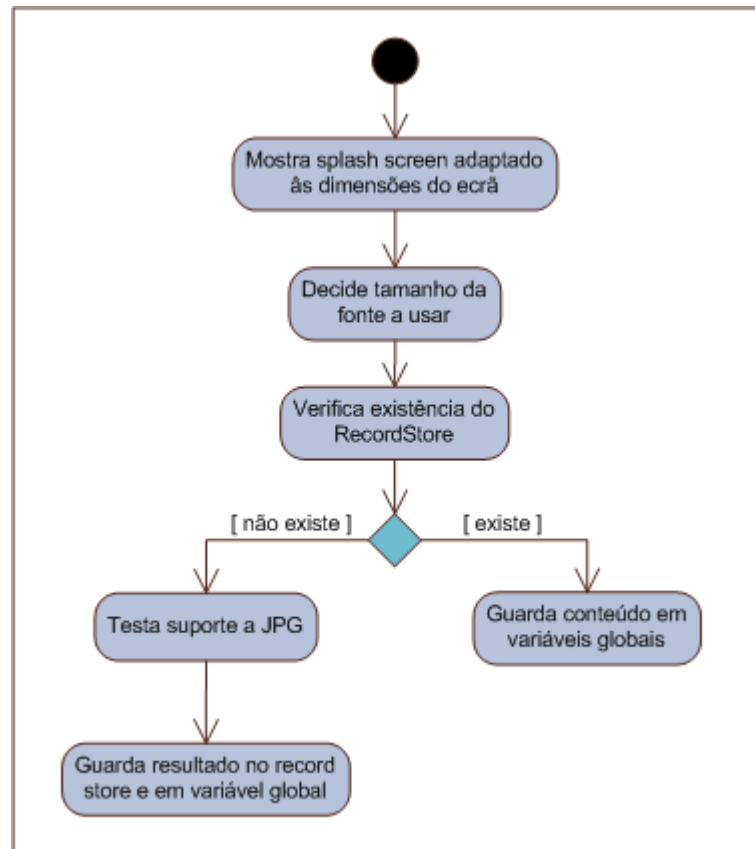


Fig. 28: Diagrama de actividades do início da execução das aplicações.

Em todas as aplicações existem menus e eventualmente submenus. De forma a tornar mais simples e intuitiva a transição entre eles, foram definidas teclas de atalho que permitem o deslocamento para os menus anterior e posterior (teclas de deslocamento para a esquerda e para a direita, respectivamente). Para isto foram criadas subclasses da classe `Menu` do J4ME que fazem o *override* dos métodos associados a eventos de teclas, dependendo do comportamento desejado para cada tecla. Se o aparelho disponibilizar um ecrã tátil, este também poderá ser utilizado para a selecção. As setas para a direita (triângulos) que aparecem nos *screenshots* indicam ao utilizador que existem mais opções em menus posteriores.

4.3. Servidor

4.3.1. Web Services

Para se ter uma ideia da afluência do público aos serviços disponibilizados, todos os acessos aos Web Services são registados na base de dados. Cada registo inclui o IP do dispositivo, a data, o *user agent* e o método ao qual se acedeu.

Para a aplicação “Uma Ocasião, um Vinho” existe apenas um método, responsável por retornar uma lista aleatória de vinhos com base nos códigos da região, tipo e casta assim como no número máximo de vinhos pedido.

```
function pesquisaVinhosPorCodigos($codCasta, $codTipo, $codRegiao,
$maxVinhos)
```

É invocado o método da lógica de negócio com os três códigos recebidos. O resultado obtido é depois baralhado usando uma função do PHP específica para o efeito. Posteriormente os campos dos primeiros `$maxVinhos` (ou menos no caso de não se atingir esse valor) elementos do *array* resultante são filtrados e tratados – campos que não interessa enviar são descartados, campos nulos ou só com espaços são substituídos por *string* vazia, espaços nos nomes dos ficheiros de rótulos são substituídos por “%20”, informações sobre colheitas que já apareçam no nome do vinho são retiradas de forma a evitar redundância e algumas *strings* são abreviadas de forma a ocuparem o mínimo de espaço no *display* dos dispositivos. O resultado é depois retornado ao dispositivo.

Para a aplicação “Pesquisa de Vinhos” são usados dois métodos, um para a pesquisa básica e outro para a pesquisa avançada. Tal como já foi referido, esta aplicação permite ao utilizador definir uma ordenação de resultados e fazer novos pedidos de forma a visualizar os resultados seguintes. Desta forma cada um dos dois métodos recebe, para além dos campos de pesquisa e do número máximo de resultados pretendido, um código relativo à ordenação dos resultados e outro relativo ao índice inicial a partir do qual estes devem ser devolvidos.

```
function pesquisaVinhosTexto($texto, $ordenacao, $indiceInicial,
$maxVinhos)
```

O método anterior é o relativo à pesquisa básica, recebendo como único critério de pesquisa uma *string* genérica, `$texto`. É chamada a função da lógica de negócio responsável por pesquisas genéricas, passando-lhe também o código de ordenação. O resultado obtido é depois filtrado e tratado de forma semelhante a `pesquisaVinhosPorCodigos`, exceptuando o uso da função responsável por baralhar o *array* e o facto de agora incluir na resposta o número total de vinhos encontrados.

O método responsável pela pesquisa avançada (`pesquisaVinhos`) difere da pesquisa básica fundamentalmente nos parâmetros recebidos (todos aqueles que podem ser especificados do lado do cliente) e na função da camada de lógica de negócio usada.

```
function pesquisaVinhos($nome, $produtor, $casta, $tipoVinho, $regiao,
$colheita, $ordenacao, $indiceInicial, $maxVinhos)
```

4.3.2. PhpThumb

Os rótulos não devem ser transferidos no seu tamanho original porque iriam na maior parte dos casos exceder as dimensões dos ecrãs dos dispositivos móveis, quando aquilo que se pretende é adaptar as imagens aos mesmos. O `PhpThumb` é uma ferramenta *open-source* que

é usada no portal Infovini para redimensionamento de imagens, na criação de miniaturas (*thumbnails*) por exemplo. Esta ferramenta permite especificar alguns parâmetros úteis no contexto da aplicação como por exemplo indicar (no próprio URL) as dimensões desejadas das imagens (em pixéis) e o formato para o qual se pretende converter (neste caso PNG ou JPG). Segue-se um exemplo de uma imagem que se pretende que tenha uma largura máxima de 200 pixéis e que seja retornada no formato PNG:

```
http://www.infovini.com/phpthumb/phpthumb.php?src=/imagens/rotulos/ro_380.jpg&w=200&f=png
```

ou, generalizando,

```
http://[URL_da_pasta_do_PhpThumb]/phpthumb.php?src=[path_da_imagem_original][parâmetros]
```

De forma a transferir uma imagem é necessário em primeiro lugar criar uma conexão (*HttpConnection*), abrir uma *input stream* e ler o seu conteúdo para um *byte array* que é depois usado na inicialização da imagem. Para se fazer a leitura de uma só vez, o *byte array* tem que ter sido inicializado com o tamanho da imagem a transferir. As respostas devolvidas pelo *PhpThumb* não contêm informação relativa ao tamanho das imagens (não é incluído o *header* “Content-Length”), de maneira que não é possível à partida saber qual o tamanho do *byte array*. Foi por isso necessário pensar num método diferente de receber as imagens. Existiam duas soluções para resolver o problema: uma passava por adaptar o código da função do lado da aplicação de forma a receber as imagens em *chunks* (em blocos de tamanho definido) onde cada *chunk* ia sendo adicionado ao *byte array* até os dados terminarem, ou adaptar o código do próprio *PhpThumb* de forma a incluir o Content-Length. Escolheu-se a segunda opção já que simplifica o código do lado da aplicação e reduz o nível de processamento do dispositivo, que desta forma poderá guardar a resposta recebida de uma só vez pois sabe à partida o espaço que deverá alocar.

O *PhpThumb* permite especificar uma altura ou uma largura máxima das imagens consoante se trate de uma imagem do tipo *landscape* (largura superior à altura) ou *portrait* (altura superior à largura). Uma vez que se pretende manter as relações entre largura e altura das imagens, se estas forem do tipo *landscape* interessa que o factor limitador seja a largura e se forem do tipo *portrait* interessa restringir a altura. Para isso são usadas as propriedades *wl* e *hp* (*weight landscape* e *height portrait*) ou *wp* e *hl* (*width portrait* e *height landscape*). Segue-se um exemplo:

```
http://[URL_da_pasta_do_PhpThumb]/phpthumb.php?src=[path_da_imagem_original]&wl=[imgWidth]&hp=[imgHeight][outros_parâmetros]
```

4.3.3. Procedimentos do lado do servidor para “Rotas do Vinho”

Como referido e justificado em 3.4.2.2, a arquitectura utilizada para adaptar a funcionalidade “Rotas do Vinho” difere das anteriores já que neste caso não é usado SOAP, sendo utilizado um conjunto de quatro procedimentos existentes do lado do servidor que retornam texto (que poderá corresponder a um URL do Google Static Maps ou a uma descrição de um local) ou imagens (fotografias ou mapas).

4.3.3.1. getMapaItinerario.php

Este procedimento é o responsável por responder aos pedidos relacionados com as imagens dos itinerários e representou o maior desafio ao nível de implementação do lado do servidor para a aplicação “Rotas do Vinho”. Recebe oito argumentos:

- `codItinerario`: o código do itinerário;
- `maxlargura`: a largura máxima da imagem;
- `maxaltura`: a altura máxima da imagem;
- `formato`: o formato da imagem ("png" ou "jpg");
- `zoom`: o `zoom` a aplicar à imagem;
- `dlat`: o deslocamento na latitude;
- `dlng`: o deslocamento na longitude;
- `resultado`: o tipo de retorno (0 para URL, 1 para imagem).

Existem quatro tabelas na base de dados que contêm informação relativa a itinerários/locais que são utilizadas ao longo da execução deste procedimento. Pode consultar-se um diagrama das mesmas na Fig. 29.

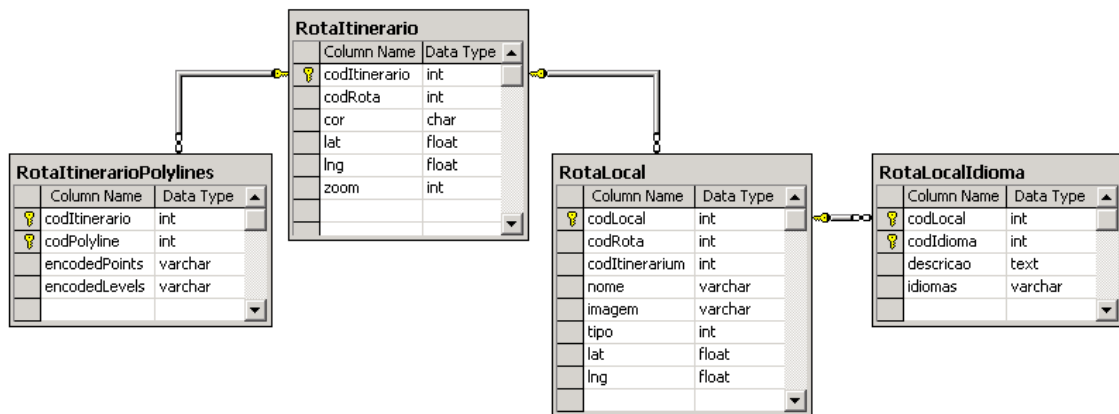


Fig. 29: Diagrama de tabelas relativas à aplicação "Rotas do Vinho".

A tabela `RotaItinerario` é usada para obter as coordenadas do centro do mapa, assim como o seu `zoom` inicial (`lat`, `lng` e `zoom`). O centro a utilizar é então calculado somando os deslocamentos recebidos por HTTP GET ao centro original.

De seguida são obtidos dois *arrays*, um contendo os códigos dos locais e o outro contendo os respectivos tipos, dependendo do itinerário. Dependendo da ordem do *array* de locais, são decididas as letras a usar nos marcadores. Dependendo dos tipos dos locais, são também decididas as suas cores. Os *arrays* de locais e dos respectivos tipos têm a mesma ordem daqueles que se encontram no código da aplicação, de forma a existir uma coerência entre as legendas dos locais e os respectivos mapas. A informação relativa às coordenadas de cada ponto de interesse encontra-se armazenada na tabela `RotaLocal`.

Com base nos parâmetros recebidos relativos às dimensões da imagem para apresentar no dispositivo, calcula os valores mínimos e máximos das latitudes e longitudes, para decidir quais os marcadores que serão incluídos. Os que estiverem fora da área de visualização são descartados da imagem do mapa.

Foi encontrado um problema que diz respeito ao tamanho dos endereços Web que o serviço do Google Static Maps aceita. A documentação do serviço não impõe limites, no entanto eles existem. Nos testes realizados era recebido um erro genérico de “Bad Request” sempre que se usavam URL superiores a cerca de 1900 caracteres. Apesar de a API restringir o número de marcadores e de pontos das *polylines* a 50 e de permitir níveis de precisão até às 6 casas decimais para as coordenadas, chegou-se à conclusão que o número máximo de caracteres excedia em muito o permitido se se chegasse perto dos limites oficialmente impostos. As mensagens de erro apareciam em dois itinerários com mais de 30 marcadores. Foram por isso adoptadas duas estratégias de redução do número de caracteres do URL:

- **redução da precisão das coordenadas dos *markers* nos níveis de *zoom* mais baixos (mais afastados):** os erros aconteciam nos níveis de *zoom* mais reduzidos porque nesses níveis estão presentes todos os marcadores que fazem parte do itinerário. Uma vez que quanto menor for o nível de *zoom*, menor é a precisão necessária, o procedimento foi feito de modo a adaptar o número de casas decimais das coordenadas ao nível de *zoom* pretendido. À medida que se vai aumentando o nível, parte dos marcadores vão sendo descartados, de maneira que se pode ir aumentando a precisão. Conseguiram-se assim reduções de quatro caracteres por marcador nos níveis iniciais, o que se traduz numa redução de 128 caracteres num dos itinerários que não podia ser visualizado e de 144 caracteres no outro.
- **omissão dos valores correspondentes às quantidades de vermelho e verde na definição da cor das *polylines*:** tendo definido que a cor dos percursos seria sempre o azul com 50% de transparência, não havia necessidade de se usar os quatro caracteres correspondentes aos códigos hexadecimais do vermelho e do verde que fazem parte do modelo de cores RGB. Desta forma passou a usar-se `0xff80` em vez de `0x0000ff80`, tendo-se conseguido uma redução de quatro caracteres por cada *polyline*.

As estratégias adoptadas permitiram reduzir o número de caracteres para valores aceites pelo serviço.

Na API “dinâmica”, os pontos a incluir na definição de uma *polyline* (especialmente naquelas que contêm um número elevado de pontos) podem ser em primeiro lugar codificados para um conjunto compactado de caracteres ASCII que depois é passado ao serviço, compactando-se assim o código e tornando-se o desenho da linha mais rápido. É permitido também especificar o nível máximo de *zoom* para cada segmento, tornando mais eficiente o processo de desenho de *polylines* em níveis de *zoom* mais baixos⁶⁵. De forma a adaptar a informação relativa a cada percurso ao contexto das aplicações para dispositivos móveis foi preciso em primeiro lugar descodificar as *polylines*, cujas *strings* se encontram na tabela `RotaItinerarioPolylines`. O método de descodificação usado faz o processo inverso ao descrito na página referente aos algoritmos de codificação e é da autoria de Ian Dees, estando disponível na Web num fórum aberto de partilha de ideias, código e ajuda para os serviços do Google Maps [Map07]. Uma vez descodificadas as *strings* das *polylines*, é necessário reduzi-las a no máximo 50 pontos, dados os limites impostos pela Google Static Maps API. O nível de redução depende do itinerário em causa. A Tabela 2 contém o número de pontos

⁶⁵ Os algoritmos de codificação podem ser consultados em [Goo08].

originalmente definidos para os percursos na API “dinâmica” e a redução requerida para que ficassem com o máximo de 50 pontos exigidos pela API estática.

Tabela 2: Reduções dos pontos das *polylines*.

Código do itinerário	Número de pontos total	Redução (%)
2	67	25.37
3	660	92.42
4	174	71.26
5	245	79.59
6	426	88.26
7	81	38.27
8	901	94.45
9	878	94.30
10	543	90.79
11	514	90.27
12	795	93.71
13	806	93.79
14	291	82.81
15	855	94.15
16	489	89.77
17	927	94.60
18	838	94.03
19	2612	98.08
20	1980	97.47
21	1987	97.48
22	195	74.35
23	151	66.88
24	350	85.71
25	1146	95.63
26	715	93.00
27	1022	95.10
28	311	83.92
29	929	94.61
30	656	92.37
31	405	87.65
32	781	93.59
33	977	94.88
34	1697	97.05
35	928	94.61
36	582	91.40
37	513	90.25
38	546	90.84
39	886	94.35
40	924	94.58
41	423	88.17
42	171	70.76

Como se pode verificar, são necessárias em alguns casos reduções da ordem dos 97-98%. Depois de alguns testes efectuados com um algoritmo de redução em passos iguais,

concluiu-se que reduções desta magnitude aconselham o uso de um algoritmo com uma estratégia de eliminação de pontos considerados menos relevantes em detrimento de outros considerados essenciais, de forma a manter as características dos caminhos. Adoptou-se por isso um algoritmo usado recorrentemente em aplicações comerciais de GIS, o de Douglas-Peucker [Clo00]. Este algoritmo foi proposto em 1973 e é até hoje considerado o melhor ao nível da preservação das características das poligonais originais [Clo00]. O algoritmo recebe como argumento um valor de tolerância com base no qual vai decidir quais os pontos a descartar. O seu funcionamento é recursivo, começando por calcular a distância do segmento de recta formado pelos pontos inicial e final da *polyline* ao ponto mais longínquo. Se esse ponto se encontrar mais distante do que a tolerância definida, mantém-no e repete o procedimento para os dois segmentos de recta resultantes. Se, pelo contrário, a sua distância à recta formada pelos dois pontos extremos for inferior à tolerância, elimina-o assim como a todos os pontos que estejam a uma distância intermédia. Está disponível na Web uma implementação *open-source* em PHP deste algoritmo da autoria de Anthony Cartmell, que nos testes efectuados funcionou correctamente e foi por isso adoptada [Fon].

Para os cálculos dos *arrays* das *polylines* são usados apenas os pontos que se encontram na área de visualização, de forma a aumentar-se a precisão das mesmas. São também incluídos os pontos que se situam imediatamente antes e depois da área de visualização, de forma a evitar quebras nas linhas. Na Fig. 30 pode ver-se uma imagem explicativa.

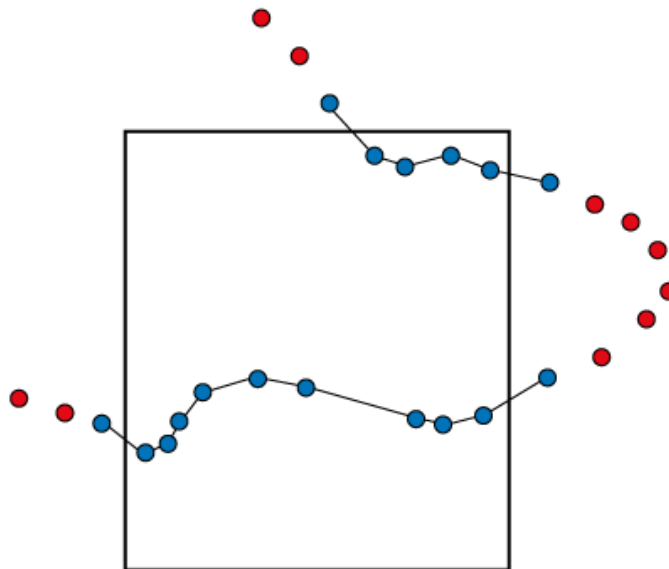


Fig. 30: Eliminação de pontos irrelevantes em `getItinerarioMapa.php`.

Considerando que a área de visualização é representada pelo rectângulo de bordas pretas, os pontos a vermelho podem ser descartados de forma a diminuir a redução necessária do *array* e desta forma aumentar a precisão das linhas. Tal como referido, os pontos imediatamente antes e depois da área de visualização são incluídos de maneira a não levar o utilizador a pensar que as *polylines* começam e terminam dentro da mesma.

Quanto maior é o número de pontos das *polylines* de um itinerário, maior é a tolerância que deve ser usada. Foram calculados os valores das tolerâncias para os níveis de

zoom iniciais de cada percurso de forma a fazer com que o algoritmo de Douglas-Peucker devolvesse um número de pontos igual ou próximo de 50. Uma vez que o cálculo das tolerâncias necessárias para diferentes níveis de *zoom* pode dar resultados ligeiramente superiores a 50, foi criado um outro algoritmo mais simples que retira pontos ao *array* resultante do algoritmo anterior uniformemente, com base num “passo” calculado em função do tamanho do *array* recebido e do tamanho pretendido. Este algoritmo funciona por isso como uma forma de assegurar que a *polyline* a usar contém sempre um máximo de 50 pontos.

Com base nas coordenadas, letras e cores associadas aos locais assim como no *array* de pontos resultante, no formato pretendido da imagem e no *zoom*, é criada a *string* correspondente ao URL da Google Static Maps API. Se a aplicação o tiver requerido (parâmetro *resultado*), o mesmo é enviado. Se não, é verificado o número de acessos directos já feitos nas últimas 24 horas ao serviço da Google. Se ainda não chegou ao limite imposto, é guardada na base de dados a data/hora do acesso (assim como o IP e o *user agent*, para fins estatísticos) e realizado o pedido. A imagem recebida é depois redireccionada para o dispositivo.

Quer se trate de um URL ou da própria imagem, é incluído na resposta um cabeçalho indicando o nível de *zoom*, informando a aplicação do valor do *zoom* inicial relativo ao itinerário pretendido, de forma que possa depois saber quais os níveis a aplicar à medida que o utilizador decide alterá-los. Se se tratar da própria imagem inclui também o “Content-Length” de forma a indicar à aplicação o tamanho do *byte array* a alocar.

4.3.3.2. **getMapaLocal.php**

Este procedimento é o responsável por devolver o URL do Google Static Maps relativo a um local isolado. Recebe nove argumentos:

- *codLocal*: o código do local.
- *maxlargura*: a largura máxima da imagem.
- *maxaltura*: a altura máxima da imagem.
- *letra*: a letra a usar no *marker*.
- *cor*: a cor a usar no *marker*.
- *formato*: o formato da imagem ("png" ou "jpg").
- *zoom*: o *zoom* a aplicar à imagem.
- *dlat*: o deslocamento na latitude.
- *dlng*: o deslocamento na longitude.

O funcionamento deste procedimento é bastante mais simples do que o do *getMapaItinerario* na medida em que não envolve cálculos relativos a *polylines* nem existe a preocupação com o tamanho do URL, já que além de não ser usada qualquer informação relativa a *polylines*, só é usado um *marker* (o requerido). Consulta a base de dados no sentido de obter as coordenadas do local, aplica o *zoom* pretendido (se não estiver especificado aplica um valor predefinido), calcula as coordenadas do centro do mapa baseando-se nos valores obtidos da base de dados e nos deslocamento requeridos, cria o URL baseado nesses valores e nos outros parâmetros e envia-o à aplicação, juntamente com o *header* indicando o nível de *zoom* aplicado (para no caso de ser a primeira chamada, a aplicação saber qual o seu valor).

4.3.3.3. `getImageLocal.php`

Este é o procedimento responsável por enviar uma fotografia de um local (as mesmas que aparecem nos mapas do portal Infovini quando se selecciona um ponto de interesse). Recebe cinco argumentos:

- `codLocal`: o código do local.
- `maxlargura`: o valor máximo em pixéis para a largura da imagem.
- `maxaltura`: o valor máximo em pixéis para a altura da imagem.
- `formato`: o formato da imagem a retornar (“jpg” ou “png”).
- `qualidade`: o índice de qualidade associado à compressão da imagem (opcional e só relevante no caso do formato ser JPG).

O procedimento contacta a base de dados no sentido de obter o nome do ficheiro de imagem associado ao local e usa o `PhpThumb` para a redimensionar em função dos parâmetros recebidos de largura, altura, formato e qualidade da mesma. Redirecciona-a depois para o dispositivo, incluindo o *header* “Content-Length” pelas mesmas razões enunciadas anteriormente na descrição de `getMapaItinerario.php`.

4.3.3.4. `getDescricaoLocal.php`

Recebe um único parâmetro relativo ao código do local. Contacta a base de dados no sentido de obter a descrição do mesmo, filtra o resultado de forma a não incluir algumas *tags* HTML que são usadas no portal e envia-o.

4.4. Aplicação “Uma Ocasião, um Vinho”

4.4.1. Definição da ocasião

A definição das ocasiões é feita com base em menus, com um menu por cada etapa. Desta forma o utilizador necessita apenas de utilizar as teclas de deslocamento (para cima e para baixo) e a tecla de selecção. De maneira a não sobrecarregar os menus (o que, consoante as dimensões do *display* do dispositivo, poderia impossibilitar o utilizador de ver todas as opções disponíveis no ecrã), optou por se fazer referência visual ao Infovini apenas num *splash screen* contendo um logótipo mostrado no arranque da aplicação. Pode ver-se uma sequência de capturas de ecrã obtidas durante a fase de definição da ocasião na Fig. 31.

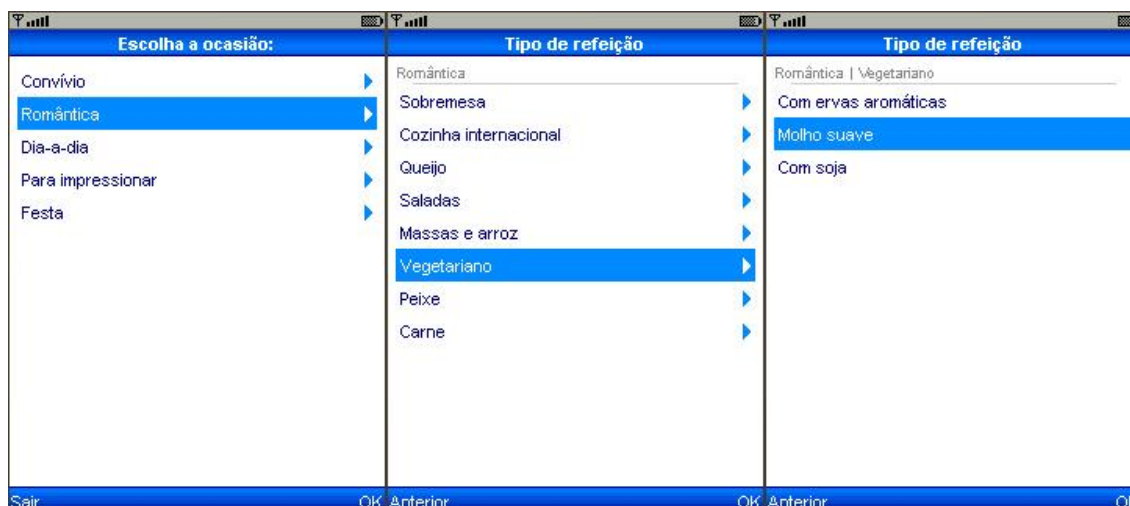


Fig. 31: Exemplo de fases na definição da ocasião na aplicação "Uma Ocasião, um Vinho" para dispositivos móveis.

À medida que a ocasião vai sendo definida, vai sendo dado *feedback* ao utilizador em relação às etapas anteriormente seleccionadas (em cima, a cinzento). Neste caso foi escolhida uma ocasião “Romântica” com um prato “Vegetariano” e o utilizador está prestes a escolher “Molho suave”.

4.4.2. Visualização de sugestões

As sugestões genéricas encontram-se armazenadas na própria aplicação, numa classe própria para o efeito. Foi criado um *script* em PHP que em função das sugestões presentes na base de dados, gerou o código Java que foi incluído na aplicação. De forma a minimizar o espaço ocupado, cada sugestão consiste num *array* de dezoito elementos⁶⁶ onde cada um dos elementos corresponde a um código. Existem 13 regiões, 5 tipos e 230 castas, sendo portanto a informação relativa aos nomes das castas a que ocupa mais espaço (aproximadamente 4 KB depois de compilar no nível máximo de *obfuscation*). É apresentado um *screenshot* de um menu com uma sugestão na Fig. 32.

⁶⁶ Já que são mostradas três sugestões por ocasião e cada sugestão é definida por seis campos (região, tipo, subtipo e três castas).

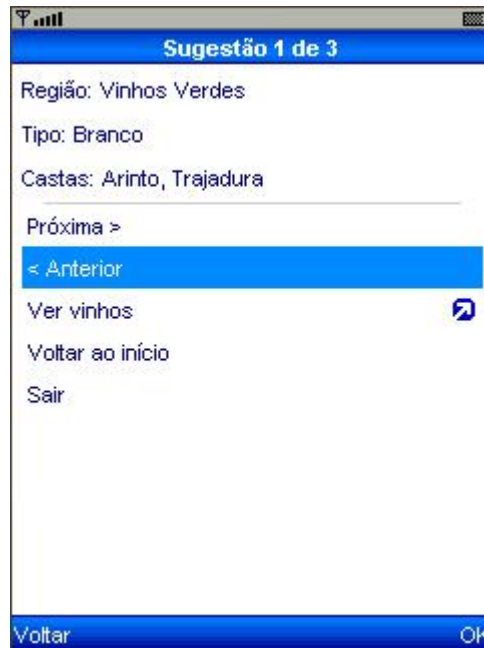


Fig. 32: Sugestão genérica na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.

Os itens “Próxima >” e “< Anterior” permitem ver outras sugestões para a ocasião definida. “Voltar ao início” permite regressar ao menu inicial e seleccionar “Sair” faz com que a execução termine. O item “Ver vinhos” encontra-se com um ícone vectorial especial para indicar ao utilizador que a sua escolha envolve uma ligação à rede. O ícone foi desenhado usando os métodos de baixo nível do MIDP 2.0 de desenho de círculos, rectângulos e linhas, tendo sido feitas algumas alterações ao código fonte do J4ME para o permitir.

4.4.3. Visualização de resultados

É pedido um máximo de trinta vinhos ao Web Service. Este valor é *hardcoded* em oposição a ser susceptível de alteração pelo utilizador já que se optou por não introduzir mais complexidade do ponto de vista da usabilidade. Os vinhos são apresentados numa primeira fase no modo de lista, podendo eventualmente ser feita uma truncagem do seu nome se o mesmo ocupar mais espaço que a largura do ecrã. Se houver espaço para isso e se existir informação relativa à idade ou colheita, um destes campos também é mostrado. Pode ver-se um exemplo de uma lista de resultados na Fig. 33.



Fig. 33: Lista de vinhos na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.

O fluxo de actividades relacionadas com a transferência de dados relativos a vinhos encontra-se resumido no diagrama da Fig. 34:

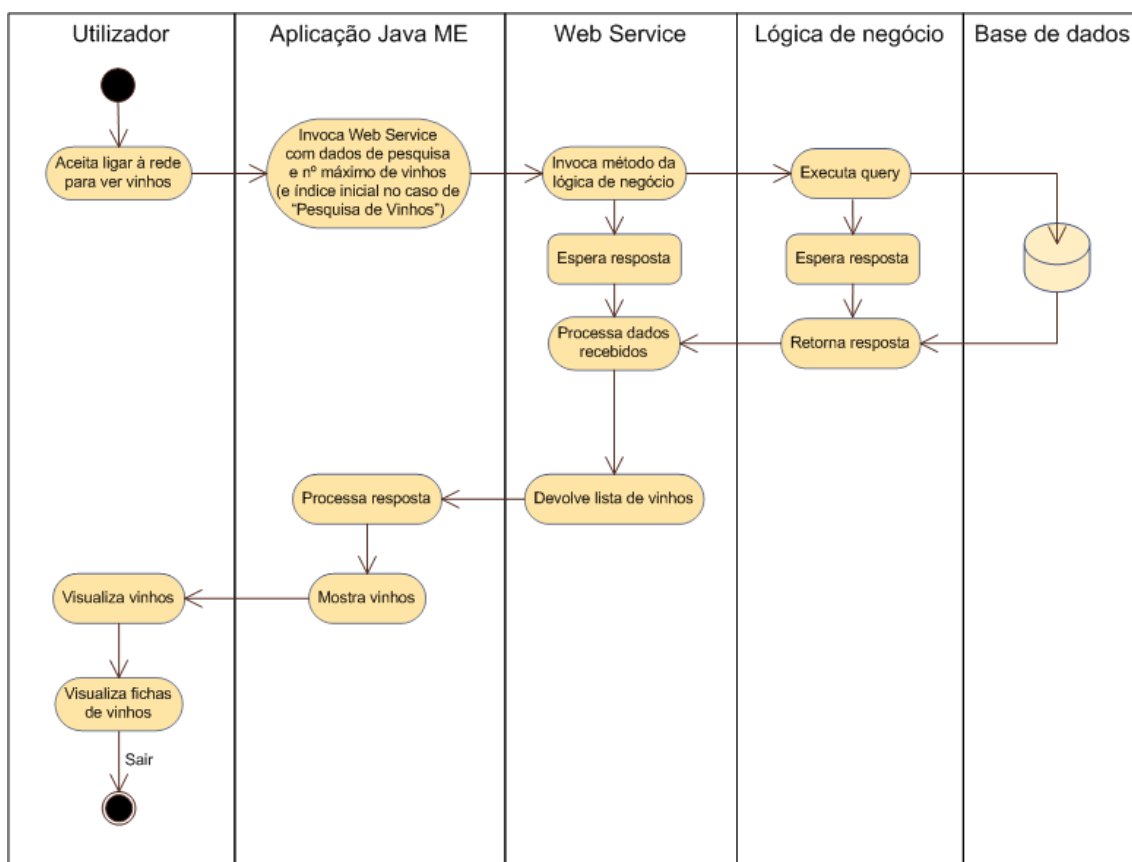


Fig. 34: Diagrama de actividades da transferência de vinhos nas aplicações "Uma Ocasão, um Vinho" e "Pesquisa de Vinhos".

Ao seleccionar um dos vinhos, é apresentada a sua ficha com todos os dados considerados relevantes (Fig. 35). O tamanho da letra varia neste caso, sendo utilizada a maior fonte disponível na API de baixo nível do Java ME para o nome e colheita/idade.



Fig. 35: Ficha de vinho na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.

Caso exista um rótulo associado ao vinho, tem-se acesso ao botão “Ver rótulo” (que também contém o ícone vectorial para indicar que vai haver um novo acesso à Internet). O redimensionamento do rótulo em função das características do ecrã é feito do lado do servidor pelo PhpThumb, com base nos parâmetros passados por HTTP GET pela aplicação relativos à largura e altura do ecrã em pixéis. Entre esses parâmetros é também incluído o formato desejado (PNG ou JPG). Na Fig. 36 pode ver-se uma captura de ecrã de um rótulo.

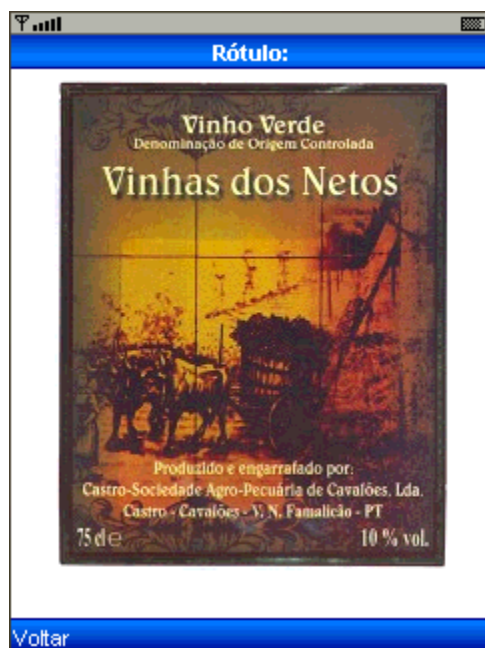


Fig. 36: Menu de rótulo na aplicação "Uma Ocasão, um Vinho" para dispositivos móveis.

O fluxo de actividades associado à transferência de um rótulo encontra-se resumido no diagrama da Fig. 37.

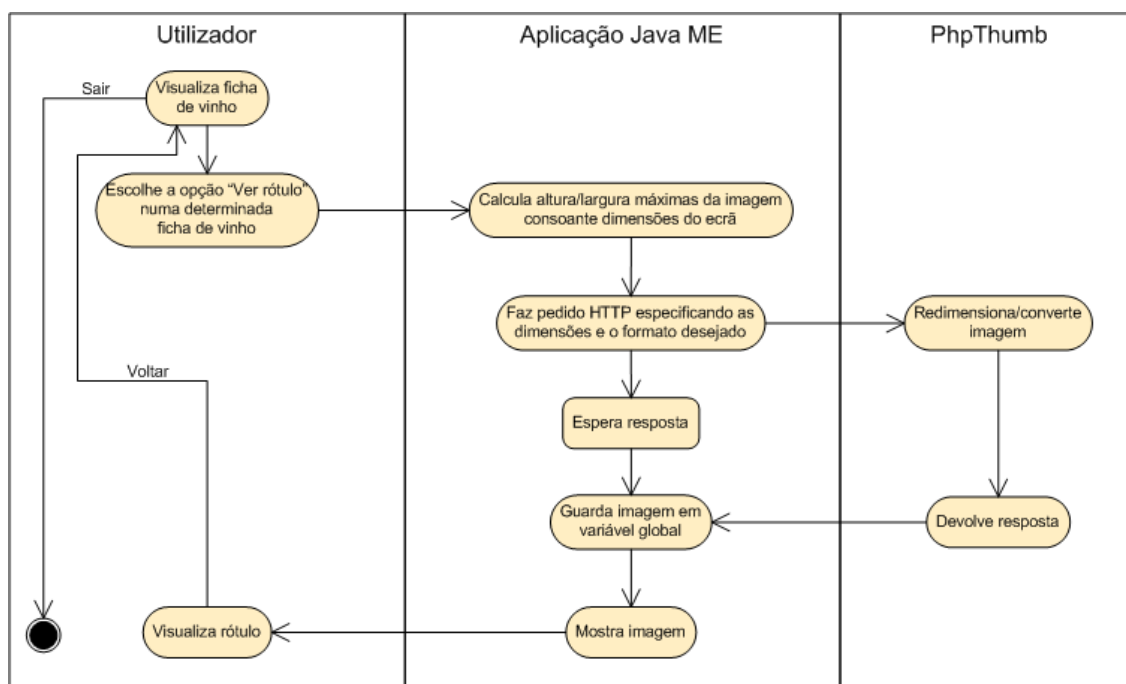


Fig. 37: Diagrama de actividades da transferência de um rótulo nas aplicações "Uma Ocasão, um Vinho" e "Pesquisa de Vinhos".

4.5. Aplicação "Pesquisa de Vinhos"

4.5.1. Pesquisa básica

Como planeado na fase de especificação, a aplicação de pesquisa de vinhos permite pesquisar com base num único campo de pesquisa genérico (que pode corresponder ao nome do vinho, nome do produtor, região, colheita, idade ou castas). Esta forma de pesquisa contém por isso uma única caixa de texto centrada verticalmente (Fig. 38). A transição para a pesquisa avançada é feita a partir de um `MenuItem` colocado logo abaixo e o comando do lado direito é usado para dar a ordem de pesquisa.

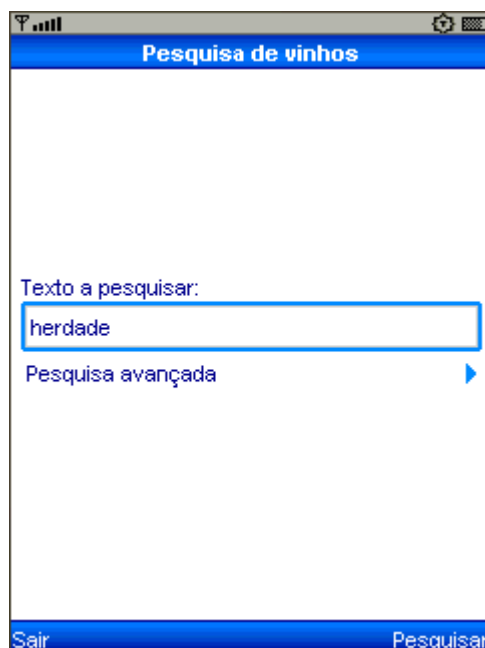


Fig. 38: Pesquisa básica da aplicação "Pesquisa de Vinhos" para dispositivos móveis.

4.5.2. Pesquisa avançada

A pesquisa avançada permite especificar os parâmetros de pesquisa. Os parâmetros que têm um conjunto fixo de opções (como tipo de vinho e a região) aparecem sob a forma de listas. O código fonte do J4ME foi alterado de modo a fazer aparecer as setas viradas para baixo e não para a direita, indicando que a selecção daquele item dá acesso a uma lista e não a um submenu. Na Fig. 39 é apresentada uma imagem com duas capturas de ecrã relativas ao menu de pesquisa avançada (variando a barra de deslocamento vertical).

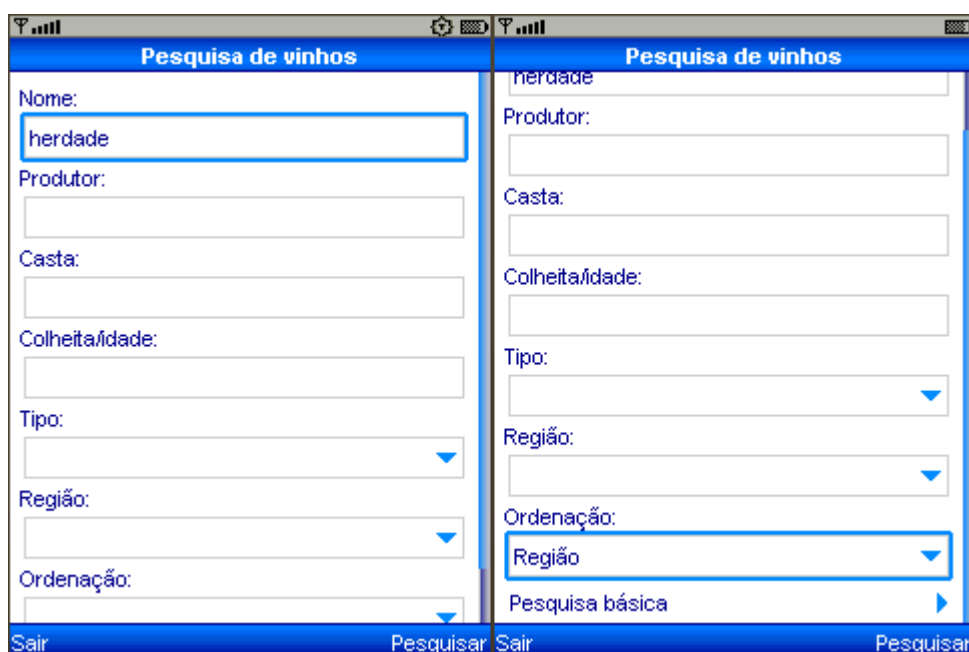


Fig. 39: Pesquisa avançada na aplicação "Pesquisa de Vinhos" para dispositivos móveis.

4.5.3. Visualização de resultados

A lista resultante das pesquisas é ligeiramente diferente da lista de “Uma Ocasão, um Vinho” na medida em que, tal como especificado, permite ao utilizador visualizar o número total de vinhos encontrado (que neste caso aparece na barra de título) assim como avançar para o próximo conjunto de resultados (a partir de um botão adicional no fim da lista de vinhos), fazendo novos pedidos ao Web Service com um índice inicial superior.



Fig. 40: Resultados das pesquisas básica e avançada na aplicação "Pesquisa de Vinhos" para dispositivos móveis.

Na Fig. 40, a primeira captura de ecrã corresponde à pesquisa básica pela palavra “Herdade” e a segunda captura corresponde à pesquisa avançada pela mesma palavra mas restringindo os resultados aos vinhos que a contêm no nome, daí o facto de se encontrarem bastantes mais resultados na primeira. Alguns nomes dos vinhos que não cabem na largura do ecrã são truncados. Esta truncagem é feita com base num método da classe `Font` da API de baixo nível do MIDP que devolve as dimensões em pixéis de uma *string* numa determinada fonte (`stringWidth(string nomeDoVinho)`). O resultado é então comparado com o resultado da invocação de um método da classe `Canvas` da API de baixo nível do MIDP que devolve a largura do Canvas em pixéis (`getWidth()`). A *string* é depois dimensionada de forma a ocupar a largura máxima do ecrã, sendo incluído o carácter “...” para indicar ao utilizador que o nome é maior do que o visível na lista. Ao fazer o dimensionamento depender do tamanho dos ecrãs assegura-se o cumprimento dos requisitos de flexibilidade.

As apresentações das fichas de vinhos e dos rótulos são iguais às da aplicação “Uma Ocasão, um Vinho”. Os diagramas de actividades genéricos relacionados com a obtenção dos vinhos e dos rótulos também são os mesmos.

4.6. Aplicação “Rotas dos Vinhos”

4.6.1. Definição da região e do itinerário

A definição da região e do itinerário correspondente é feita com base em objectos Menu. Na Fig. 41 pode ver-se uma imagem com duas capturas de ecrã relativas ao processo de escolha da rota e do itinerário.

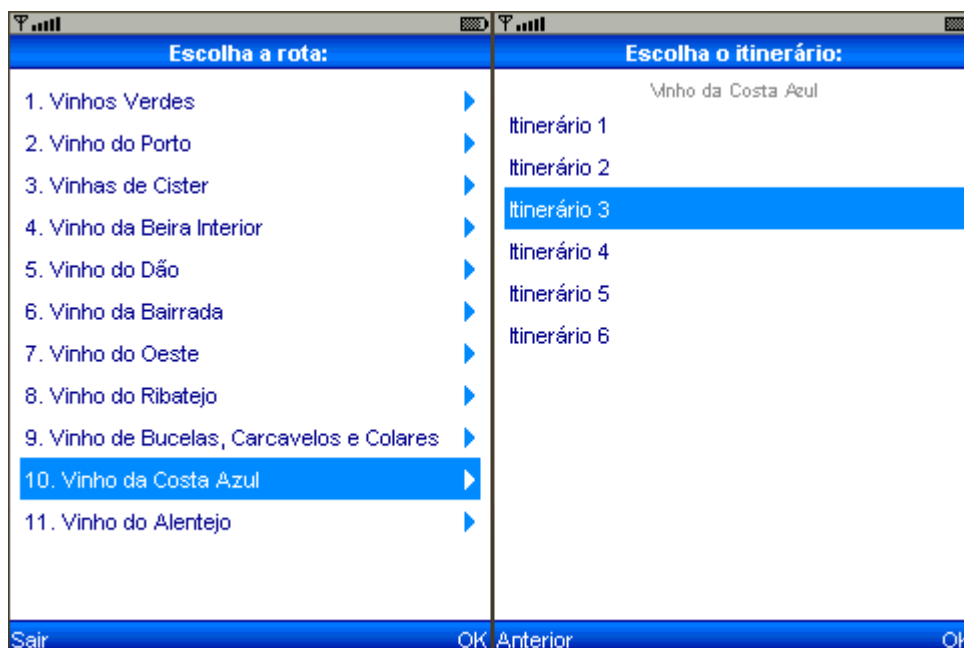


Fig. 41: Definição da rota e do itinerário na aplicação "Rotas do Vinho" para dispositivos móveis.

A partir do momento em que escolhe o itinerário, se o utilizador o desejar, é transferido o mapa do mesmo e mostrado. Se negar a ligação, é mostrado apenas o menu contendo a lista de pontos de interesse do itinerário escolhido, que dá acesso aos contactos de alguns deles (das quintas) sem necessidade de ligação à rede.

4.6.2. Mapa de itinerário

O primeiro mapa de itinerário tem normalmente um nível de *zoom* baixo e dependente do itinerário pedido, permitindo ao utilizador ter uma noção abrangente de todo o percurso (Fig. 42).

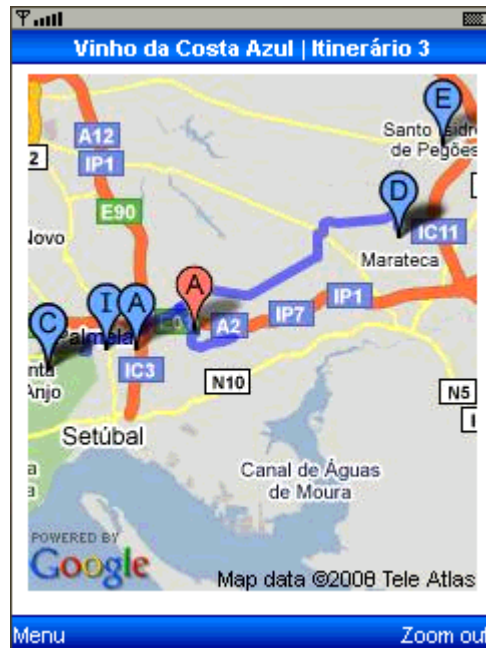


Fig. 42: Mapa de itinerário na aplicação "Rotas do Vinho" para dispositivos móveis.

Como já foi referido, alguns dispositivos mais antigos não são capazes de abrir endereços Web com um elevado número de caracteres, de forma que é guardada no *record store* da aplicação a capacidade de os abrir. Se conseguir, é pedido ao servidor o URL da Google Static Maps API que depois é usado para contactar o serviço directamente. Se não, o pedido ao servidor indica que se pretende imediatamente o mapa, que eventualmente será retornado (caso o servidor conclua que ainda não excedeu o limite das 1000 visualizações imposto pelo serviço da Google). O menu do mapa de itinerário dá acesso directo à legenda do mesmo (usando o comando “Menu”) para que o utilizador possa rapidamente saber ao que se refere cada *marker*. As teclas de deslocamento permitem navegar no mapa, pedindo ao servidor o URL da imagem (ou a própria imagem) com o seu centro deslocado na latitude ou na longitude em relação ao centro inicialmente usado. O comando associado à tecla da direita (“Zoom out”) permite diminuições do nível de *zoom*, enquanto que a tecla do meio permite aumentá-lo.

Uma vez que a Google Static Maps API permite apenas o uso dos 26 caracteres do alfabeto e alguns dos mapas continham mais do que 26 marcadores, decidi fazer-se uma diferenciação por cores além da diferenciação por letras. Existem quatro tipos distintos de locais, sendo associada uma cor diferente aos marcadores de cada um deles, tal como no portal Infovini (azul para quintas, vermelho para povoações, castanho para monumentos e verde para parques naturais).

4.6.3. Lista de pontos de interesse

A lista de pontos de interesse (que no fundo funciona como uma legenda do mapa de itinerário) mostra os nomes dos locais em cores diferentes, consoante o seu tipo (Fig. 43). Os objectos `MenuOption` da *framework* J4ME só podiam inicialmente estar associados a um tema por cada menu. O código fonte teve por isso de ser alterado de forma a permitir associar temas (classes derivadas de `Theme`) diferentes a cada um dos botões.



Fig. 43: Screenshots relativos a uma legenda de um mapa na aplicação "Rotas do Vinho".

Os primeiros dois itens dão acesso aos dois menus anteriores. Os itens seguintes correspondem a quintas (a azul) e a povoações (a vermelho). O mapa escolhido como exemplo não contém monumentos nem parques naturais (que apareceriam a castanho e a verde respectivamente). Com esta diferenciação de cores, e uma vez que a Google Static Maps API permite apenas o uso de uma letra por marcador, pode ver-se facilmente a associação entre um *marker* e o nome do local respectivo.

4.6.4. Ficha de ponto de interesse

Como definido e justificado na fase de especificação, as descrições associadas às quintas encontram-se armazenadas no próprio código da aplicação. Se o utilizador pretender visualizar informações de qualquer outro tipo de locais, a aplicação terá que as transferir da rede fazendo uso de `getDescricaoLocal.php`, que retorna a descrição do local presente na base de dados a partir do seu código. Na Fig. 44 pode ver-se um exemplo de um ficha relativa a uma quinta.

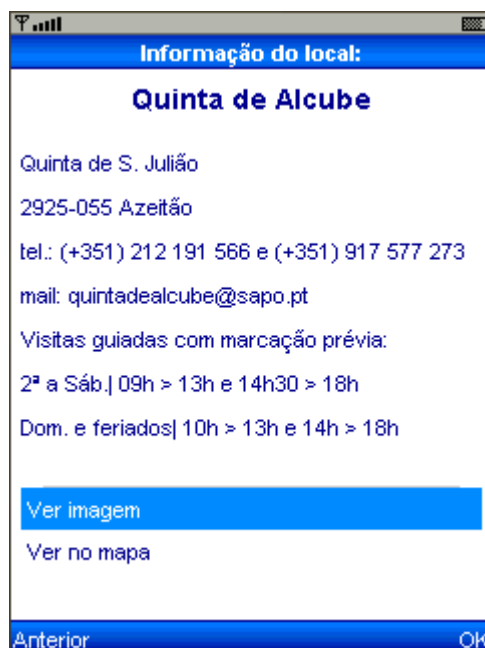


Fig. 44: Ficha de local na aplicação "Rotas do Vinho" para dispositivos móveis.

Por baixo da descrição associada o utilizador tem acesso a dois botões, um para ver a fotografia do local (se existir) e outro para ver o local isoladamente no mapa. Na Fig. 45 pode ver-se uma imagem de um local semelhante à que aparece no portal quando o mesmo é seleccionado.

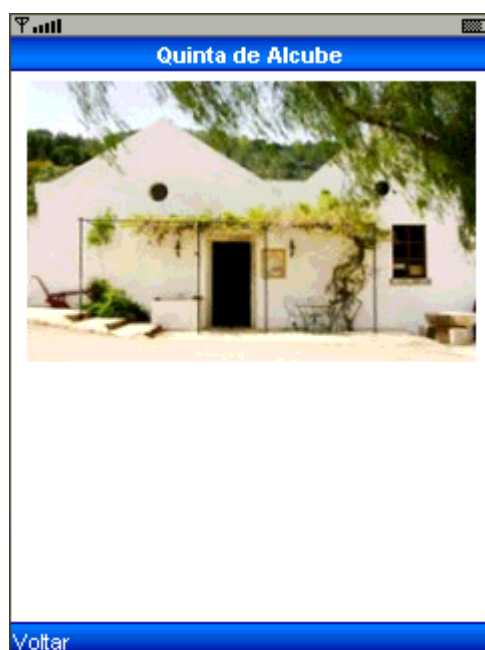


Fig. 45: Imagem de um local na aplicação "Rotas do Vinho" para dispositivos móveis.

A classe derivada de Menu usada para a opção referente à visualização de um local isoladamente no mapa (sem outros pontos de interesse ou *polylines*) é a mesma que é usada para ver os mapas de itinerários (diferindo a imagem, o título e o comando do lado esquerdo) de forma que o comportamento associado a cada evento de teclado é o mesmo

(nomeadamente deslocamentos horizontais e verticais e alterações ao nível de *zoom*). Pode visualizar-se um exemplo de um mapa deste tipo na Fig. 46.

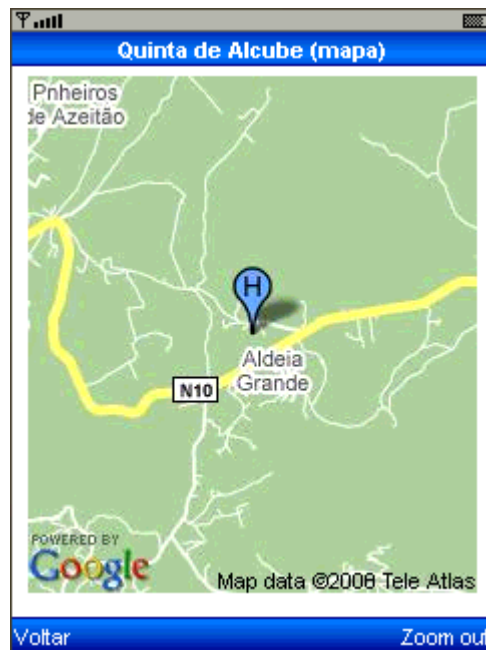


Fig. 46: Mapa de local na aplicação "Rotas do Vinho" para dispositivos móveis.

5. Conclusões

Os objectivos do projecto foram alcançados, tendo-se adaptado com sucesso as três funcionalidades incluídas no planeamento. Os casos de uso foram implementados e os requisitos não funcionais estipulados foram cumpridos. O Java ME, pelas razões referidas no capítulo 2, é actualmente a tecnologia de implementação que dá mais garantias de portabilidade em dispositivos móveis. A *framework* escolhida para a interface de utilizador (J4ME), ligeiramente alterada de forma a se adaptar às necessidades das aplicações, teve um papel preponderante no cumprimento do requisito de usabilidade já que permitiu a criação de interfaces simples e apelativas e também a introdução de redundância nas transições entre diferentes menus de forma a tornar a navegação intuitiva. A arquitectura adoptada baseada em SOAP no que diz respeito às duas funcionalidades que envolvem transferência de dados sobre vinhos cumpre com o requisito de interoperabilidade e abre as portas ao desenvolvimento de novas aplicações que manipulem este tipo de informações. O NuSOAP e o kSOAP permitiram implementar de forma satisfatória o servidor e os clientes de Web Services, tendo a curva de aprendizagem no que respeita à utilização destas bibliotecas sido curto muito devido à documentação existente e à popularidade das mesmas. O aproveitamento de determinadas características de alguns dispositivos de forma a minimizar a quantidade de dados a transferir da rede, conjugado com a adaptação dos conteúdos às dimensões dos ecrãs cumprem com o requisito de flexibilidade estabelecido inicialmente.

Apesar das dificuldades anteriormente reportadas encontradas no decurso do desenvolvimento da aplicação “Rotas do Vinho” no que diz respeito às limitações de tamanho dos endereços Web para a Google Static Maps API, o calendário estabelecido para a implementação da mesma foi cumprido uma vez que se sabia à partida que esta aplicação poderia representar um maior desafio que as anteriores e se definiu um prazo mais alargado para o seu desenvolvimento.

O carácter inovador do trabalho desenvolvido contribui de forma muito positiva para a missão do Infovini de promoção e divulgação do sector vitivinícola, colocando-o na senda do progresso no que respeita à apresentação de conteúdos em dispositivos móveis.

5.1. Perspectivas de desenvolvimento futuro

A adaptação dos conteúdos do portal Infovini a dispositivos móveis não é um processo estanque. O primeiro relatório intermédio a ser elaborado, disponível no Apêndice B, indica um conjunto de outras funcionalidades de menor prioridade mas também passíveis de adaptação, nomeadamente as referentes a notícias e eventos relacionados com o sector vitivinícola. A criação de uma versão do portal optimizada para suportes móveis é também um dos objectivos da equipa de desenvolvimento. De realçar também que o próprio portal está em constante desenvolvimento, o que significa que poderão surgir novas funcionalidades passíveis de adaptação.

Em relação às funcionalidades adaptadas no âmbito do projecto, prevê-se a inclusão de imagens nas etapas de definição da rota e do itinerário da aplicação “Rotas do Vinho”. O

objectivo é substituir a forma de selecção baseada numa lista por uma selecção baseada em imagens das rotas e dos itinerários, de maneira a dar um melhor *feedback* visual ao utilizador das opções existentes que desta forma poderá vê-las num mapa. Prevê-se também a inclusão de uma pesquisa de pontos de interesse nessa aplicação. Uma vez que os nomes dos pontos de interesse já se encontram armazenados no próprio código, a visualização dos resultados na forma de listagem não envolveria a transferência de dados do servidor, assim como a visualização de descrições associadas a quintas. Tal como acontece actualmente, a apresentação de imagens e mapas de locais, assim como de descrições de povoações, monumentos ou parques naturais envolveria o uso dos procedimentos PHP já criados do lado do servidor. Assim, a adição de um módulo de pesquisas envolveria apenas desenvolvimento do lado da aplicação.

O INEGI é uma instituição que se afirma muito pelo carácter inovador dos seus projectos. Desta forma, a equipa de Sistemas de Informação onde o autor se inseriu, responsável pelo desenvolvimento e manutenção do portal Infovini e imbuída no espírito da empresa, planeia agora o desenvolvimento de aplicações para um dos dispositivos móveis mais em voga actualmente, o iPhone. Apesar de a Sun Microsystems estar já a desenvolver uma JVM para o iPhone OS baseada no Java ME, considera-se o desenvolvimento de pelo menos uma aplicação (“Uma Ocasão, um Vinho”) para este aparelho a partir do SDK disponibilizado pela Apple em Março. A percentagem de mercado à qual se destinaria esta versão da aplicação seria obviamente muito mais reduzida do que aquela à qual se destinam as aplicações Java ME desenvolvidas no âmbito do projecto, no entanto a associação do portal ao carácter inovador do iPhone seria mais um passo importante na afirmação do Infovini como o portal de referência no que diz respeito a actividades de inovação relacionadas com o sector.

5.2. Reacções

Os produtos desenvolvidos já foram apresentados em dois *workshops* realizados no âmbito da RCM⁶⁷, um Centro de Excelência que pretende “agregar, mobilizar e promover o desenvolvimento de competências e promover o desenvolvimento de produtos inovadores com potencial no mercado internacional na área do suporte das TIC à Mobilidade” [RCM07]. Nessas apresentações encontravam-se presentes vários *stakeholders* de instituições ligadas ao sector vitivinícola (tais como o IVV e do IVDP) que deram um *feedback* bastante positivo relativamente ao trabalho desenvolvido e se mostraram interessados em futuros desenvolvimentos.

⁶⁷ Rede de Competência em Mobilidade.

Referências

- [Can] Canalys. , 2008. canalys.com. Available at <http://www.canalys.com/pr/2008/r2008021.pdf>
- [CEn06] C. E. Ortiz. , 2006, Mar.. Sun Developer Network (SDN). Available at <http://developers.sun.com/mobility/midp/articles/webservices/>
- [Clo00] C. Júnior, "Simplificação de Poligonais," 3(13):28-30, 2000.
- [Coy02] F. P. Coyle, *XML, Web Services and the Data Revolution*. Addison-Wesley, 2002.
- [Eri04] E. Giguere. , 2004, . Sun Developer Network (SDN). Available at <http://developers.sun.com/mobility/midp/articles/databasermes/>
- [Fon] Fonant. Fonant. Available at http://www.fonant.com/demos/douglas_peucker/algorithm
- [Goo08] Google. , 2008. Google Code. Available at <http://code.google.com/apis/maps/documentation/polylinealgorithm.html>
- [Goo081] Google. , 2008. Google Groups. Available at <http://groups.google.com/>
- [GSM08] GSMarena. GSMarena.com - GSM reviews, opinions, votes, manuals, ringtones and more.... Available at <http://www.gsmarena.com>, last accessed on Jun. 2008
- [Hri05] Hrissan. , 2005, Jan.. CodeProject. Free source code and programming help. Available at http://www.codeproject.com/KB/mobile/Symbian_OS_design_faults.aspx
- [IVD04] IVDP, IP. , 2004. Instituto dos Vinhos do Douro e Porto. Available at <http://www.ivdp.pt/>
- [IVV] IVV - Instituto da Vinha e do Vinho. Available at <http://www.ivv.min-agricultura.pt/>
- [JBe08] JBenchmark. , 2008. Jbenchmark. Available at <http://www.jbenchmark.com/>
- [Jea04] J.-L. David. , 2004, Mar.. XML.com: XML From the Inside Out. Available at <http://www.xml.com/pub/a/ws/2004/03/24/phpws.html?page=1>
- [Jor07] J. Cardoso, *Java para telemóveis MIDP 2.0*. Porto: FEUP edições, 2007.
- [Map07] Mapki. , 2007, . Mapki. Available at <http://mapki.com/index.php?title=Polyline Decode>
- [Mob08] Mobile Phones UK. , 2008. Mobile Phones UK. Available at <http://www.mobile-phones-uk.org.uk/gprs.htm>
- [RCM07] RCM - Rede de Competência em Mobilidade. , 2007. RCM. Available at <http://rcm.inescporto.pt/a-rcm/rcm.html>
- [Sco04] S. Nichol. , 2004. Scott Nichol Consulting. Available at <http://www.scottnichol.com/nusoapintro.htm>
- [Wor] World Wide Web Consortium. World Wide Web Consortium - Web Standards. Available at <http://www.w3.org/TR/soap/>

Apêndice A: Planeamento do projecto

ID	Task Name	Start	Finish	Duration	Task Name	Fev 2008		Mar 2008				Abr 2008				Mai 2008				Jun 2008									
						17-2	24-2	2-3	9-3	16-3	23-3	30-3	6-4	13-4	20-4	27-4	4-5	11-5	18-5	25-5	1-6	8-6	15-6	22-6	29-6				
1	Análise do portal Infovini	18-02-2008	20-02-2008	3d	Análise do portal Infovini																								
2	Brainstorming	21-02-2008	21-02-2008	1d	Brainstorming																								
3	Elaboração do relatório intermédio I (funcionalidades)	21-02-2008	22-02-2008	2d	Elaboração do relatório intermédio I (funcionalidades)																								
4	Estado da arte, identificação de tecnologias, testes	25-02-2008	14-03-2008	15d	Estado da arte, identificação de tecnologias, testes																								
5	Elaboração do relatório intermédio II (tecnologias)	10-03-2008	14-03-2008	5d	Elaboração do relatório intermédio II (tecnologias)																								
6	Implementação e testes de "Uma Ocasão, um Vinho"	17-03-2008	11-04-2008	20d	Implementação e testes de "Uma Ocasão, um Vinho"																								
7	Implementação e testes de "Pesquisa de Vinhos"	14-04-2008	30-04-2008	13d	Implementação e testes de "Pesquisa de Vinhos"																								
8	Implementação e testes de "Rotas dos Vinhos"	01-05-2008	13-06-2008	32d	Implementação e testes de "Rotas dos Vinhos"																								
9	Elaboração do relatório final	16-06-2008	04-07-2008	15d	Elaboração do relatório final																								

Apêndice B: Relatório Intermediário – Análise do portal Infovini

Análise do portal Infovini

Estudo de funcionalidades a disponibilizar no contexto dos dispositivos
móveis



Relatório intermédio realizado no INEGI/Mercatura no âmbito do projecto de
estágio proposto à FEUP relativo à adaptação de funcionalidades do portal Infovini a
dispositivos móveis

Nuno Miguel Sanches Ferreira de Almeida

Fevereiro de 2008

1. Introdução

Este relatório resulta de uma análise às diferentes áreas do portal Infovini e visa definir as secções mais relevantes a serem disponibilizadas em dispositivos móveis. As conclusões patentes neste documento são o resultado de uma discussão prévia com os elementos da equipa encarregue pela elaboração e actualização do portal.

Cada funcionalidade é descrita num pequeno resumo e posteriormente é avaliada a relevância e modo como será efectuada a sua disponibilização no contexto dos dispositivos móveis. É também definida uma importância/prioridade na seguinte escala:

- 0:** não necessária - não se vê utilidade na disponibilização da mesma;
- 1:** possível - pode ser interessante disponibilizá-la mas não é prioritária;
- 2:** relevante - significa que a sua disponibilização a dispositivos móveis é prioritária.

A navegação no portal Infovini é feita a partir de um menu horizontal feito em Flash. Cada *item* do menu contém *subitems* que dão acesso às diferentes páginas. Desta forma, o próximo capítulo deste relatório está organizado por secções e subsecções, seguindo a estrutura do portal.

Por fim, no terceiro e último capítulo é feita uma ordenação decrescente por prioridade das funcionalidades a implementar.

2. Menus

2.1. Beber

2.1.1. "Uma Ocasião, um Vinho"

Com esta funcionalidade o utilizador recebe uma lista de sugestões acerca dos vinhos mais apropriados para uma determinada situação. Essa situação é uma combinação entre a ocasião (romântica, festa, convívio, entre outras), o tipo de refeição (carne, peixe, saladas, entre outros) e eventuais subtipos (no primeiro caso tem-se por exemplo carne de caça, branca e vermelha e escolhendo esta última poderá ainda indicar-se se é assada, grelhada ou condimentada).

Esta opção considera-se útil a utilizadores de dispositivos móveis já que é interessante que em qualquer lugar possam receber sugestões de vinhos para a situação em que se encontram. Durante uma convivência/reunião com pessoas conhecidas (ou mesmo um pouco antes) é muitas vezes útil, principalmente se alguns dos elementos forem apreciadores, ter uma ideia do tipo, casta e região do vinho que mais se adequa à situação.

No portal Infovini, a visualização das etapas que definem uma ocasião é feita a partir de interrogações à base de dados sucessivas, à medida que se avança. No caso da aplicação para dispositivos móveis, uma vez que importa minimizar as ligações à rede, a definição da ocasião seria feita localmente, tendo a própria aplicação informação sobre as diferentes etapas. A ligação à rede seria efectuada apenas para retornar as sugestões, no momento em que a ocasião já estivesse definida.

Grau de importância: 2.

2.1.2. "Pesquisa de vinhos"

Permite pesquisar vinhos consoante um nome, produtor, casta, tipo de vinho, região, colheita/idade e preço.

Os resultados são disponibilizados como uma lista de hiperligações para as páginas dos vinhos, que contêm várias informações sobre os mesmos assim como imagens (que poderão ser da garrafa, do rótulo e/ou do contra-rótulo).

Esta opção foi considerada interessante já que pode ser de grande utilidade dispôr de um suporte móvel, um catálogo que permita pesquisar por vinhos e visualizar algumas das suas informações em qualquer lugar. Num dispositivo móvel pretende-se que seja possível escolher um vinho da lista e visualizar essas informações (incluindo a imagem do rótulo, considerada a mais relevante das três para fácil identificação de um vinho numa prateleira). Como perspectiva futura de melhoramento, pondera-se a hipótese de implementar um mecanismo de *bookmarking* que permita ao utilizador manter uma lista de vinhos favoritos cujas informações poderá consultar sem necessidade de novas pesquisas.

Grau de importância: 2.

2.1.3. "Descubra o seu vinho"

Indica o vinho que eventualmente mais se adequa à personalidade do utilizador. Esta sugestão é mostrada depois do preenchimento de um questionário por parte do utilizador.

Inicialmente considerou-se interessante a sua disponibilização apenas a título de curiosidade já que não se prevê nenhuma situação especial em que seja absolutamente necessária. No entanto, após discussão com a equipa chegou-se à conclusão que a sua disponibilização no contexto de uma aplicação para dispositivos móveis não tem interesse já que além de funcionar apenas como uma curiosidade, em princípio seria usada apenas uma vez.

Grau de importância: 0.

2.2. Viajar

2.2.1. "Rotas do Vinho"

Sugere itinerários a turistas interessados em conhecer adegas, vinhas e tradições das regiões vitivinícolas depois do utilizador indicar a rota pretendida (uma determinada região de Portugal). Actualmente, dependendo da rota escolhida, pode sugerir de 2 a 8 itinerários. Um itinerário é um percurso entre vários pontos de interesse. Cada itinerário é mostrado como uma imagem do Google Maps dando ao utilizador a possibilidade de fazer *zoom* ou deslocar-se no mapa.

Considera-se uma funcionalidade relevante já que poderá ser interessante a consulta de percursos quando não se tem acesso a um computador com internet, por exemplo quando se está a viajar e se pretende saber o que se pode encontrar na região. Pretende-se que numa primeira fase os itinerários sejam mostrados numa lista e ao escolher cada um se possa ver as mesmas imagens do Google Maps no dispositivo móvel, como no portal. No entanto, esta solução está dependente da análise das funcionalidades do Google Maps para dispositivos móveis. Se não fôr viável implementar esta ideia, considera-se a hipótese de mostrar os itinerários em formato de texto, com a ordenação dos pontos de interesse igual à dos percursos e com informação disponível na base de dados do Infovini sobre os mesmos.

Grau de importância: 2.

2.2.2. "Enoturismo"

Disponibiliza uma lista de hiperligações relacionadas com alojamento, gastronomia, património e natureza consoante os campos de pesquisa inseridos pelo utilizador (pode por exemplo mostrar uma lista de hotéis ou restaurantes numa determinada localidade). A lista é devolvida por um Web Service do Instituto de Turismo de Portugal e as respectivas hiperligações (que permitem saber mais sobre cada item) levam o utilizador a páginas alojadas por essa instituição.

Esta opção foi considerada interessante pelas mesmas razões da funcionalidade descrita em 2.2.1. já que poderia ser útil num contexto de turismo se num determinado momento não se tem acesso a um computador e se precisa, com alguma urgência, de informações relacionadas com alojamento por exemplo. No entanto, após debate com a equipa, considerou-se que no âmbito do trabalho desenvolvido no portal Infovini não tem interesse já que, depois da análise da documentação do Web Service disponibilizado pelo Instituto de Turismo de Portugal, chegou-se à conclusão que o único retorno das pesquisas por locais de interesse é apenas um URL para a página alojada pelo referido instituto e as informações sobre cada um deles são mantidas pelo Instituto de Turismo de Portugal, não

havendo acesso às mesmas a não ser que se visualize a página Web do ponto de interesse. Assim, apesar de ser interessante para o utilizador, a disponibilização desta funcionalidade por parte de uma aplicação ligada ao portal Infovini sairia do âmbito das responsabilidades da equipa.

Grau de importância: 0.

2.2.3. "Portugal"

Página estática que disponibiliza informações genéricas e curiosidades sobre o nosso país, não se considerando a sua disponibilização útil no contexto dos dispositivos móveis.

Grau de importância: 0.

2.3. Conhecer

A partir deste menu pode aceder-se a páginas com informações sobre regiões, tipos de vinho, processos envolvidos na elaboração do mesmo, rótulos, apreciação de um vinho e glossário. Também será possível no futuro fazer perguntas relacionadas com o tema e obter respostas.

As páginas que se acedem a partir deste menu são estáticas e essencialmente informativas não se considerando particularmente úteis no contexto de uma aplicação para dispositivos móveis.

Grau de importância das páginas acedidas a partir deste menu: 0.

2.4. Produzir

À semelhança do menú "Conhecer", é uma secção fundamentalmente informativa e estática exceptuando a página "Produtores", onde se pode pesquisar por produtores e enólogos fornecendo alguns critérios de pesquisa como nome, tipo de vinho ou tipo de resultado pretendido (produtores ou enólogos). Não se considera útil a pesquisa directa por produtores/enólogos a partir de uma aplicação para dispositivos móveis.

Grau de importância das páginas acedidas a partir deste menu: 0.

2.5. Vender

2.5.1. "Distribuidores"

Permite pesquisar por distribuidores fornecendo uma palavra-chave, um país e uma localidade. Os resultados aparecem numa lista com hiperligações para as páginas dos distribuidores (páginas do portal Infovini com informações relevantes como os contactos).

A sua disponibilização futura para dispositivos móveis não se considera útil já que não se prevêem situações em que não se tem a possibilidade de usar um computador comum e se pretende aceder a esta funcionalidade por motivos de lazer ou com urgência.

Grau de importância: 0.

2.5.2. "Promover"

Consiste numa página estática com informações sobre instituições empenhadas em divulgar e promover o vinho português, de forma que não se considera útil a sua disponibilização em aplicações para dispositivos móveis.

Grau de importância: 0.

2.5.3. "Estatísticas"

É uma página estática com gráficos que mostram os níveis de produção, importação, exportação e consumo anuais de vinhos, informações que não se consideram úteis no contexto dos dispositivos móveis.

Grau de importância: 0.

2.6. Comunidade

Dá acesso a um fórum de discussão onde utilizadores poderão partilhar opiniões sobre assuntos relacionados com a indústria vitivinícola ou sobre o próprio portal, não havendo por isso interesse na sua disponibilização a uma aplicação de um dispositivo móvel.

Grau de importância: 0.

2.7. Actualidades

2.7.1. "Eventos"

Permite, escolhido um determinado semestre, visualizar os eventos que se realizaram ou que se vão realizar no mesmo. É possível filtrar os resultados por tipo de evento (prova, feira, lançamento, jantar, exposição, conferência ou concurso) e região (norte ou sul). Os resultados são mostrados num calendário com indicação dos dias em que se vai realizar, local e título do evento. Ao clicar sobre este título pode ver-se, quando disponível, uma pequena descrição do local e/ou um resumo do evento.

Considera-se uma funcionalidade interessante para ser disponibilizada já que pode haver a necessidade de, num contexto de turismo, estando fora de casa e sem acesso a um computador pessoal com acesso à internet, um utilizador saber que eventos estão agendados para uma determinada região. Após debate com a equipa responsável pelo Infovini, concluiu-se que a disponibilização da mesma faria mais sentido num contexto de sincronização com a agenda do dispositivo e não no âmbito de uma aplicação. A sua abrangência fica assim limitada a dispositivos com agenda e possibilidade de sincronização.

Grau de importância: 2, num contexto de um mecanismo de sincronização com as agendas dos dispositivos.

2.7.2. "Notícias"

Faz uma listagem das últimas notícias (ordenadas decrescentemente por data) relacionadas com o sector vinícola.

É uma funcionalidade viável de ser disponibilizada já que, a título de curiosidade pode querer saber-se num dado momento as últimas notícias relacionadas com o sector, não sendo

no entanto uma opção prioritária. Na sequência da discussão com o grupo responsável pelo portal concluiu-se que a sua disponibilização fará mais sentido na forma de *RSS Feed*. O utilizador pode assim saber quando há actualizações nas notícias e lê-las a partir do seu cliente de RSS. A sua abrangência fica assim restringida a dispositivos com aplicações que suportem este mecanismo, mas esta restrição compreende-se já que não faria sentido a sua disponibilização no âmbito de uma aplicação. Além disto, os mecanismos de Feeds têm-se generalizado nos últimos anos, prevendo-se que os dispositivos móveis tendam a suportá-lo cada vez mais e que dentro em breve esta seja uma característica comum nos mesmos.

Grau de importância: 1.

2.7.3. "Artigos de opinião"

Semelhante à secção "Notícias" mas com artigos de opinião sobre o sector, encontrados em jornais ou revistas.

É uma funcionalidade passível de ser disponibilizada mas não prioritária já que os artigos de opinião são bastante esporádicos (menos de 30 desde Setembro de 2006 até à data de escrita deste relatório) e além de não haver urgência em consultá-los, normalmente envolvem muito texto e serão lidos em ecrãs de maiores dimensões. Após a sessão de *brainstorming* com o grupo decidiu optar-se também aqui (tal como em 2.7.2.) pela sua disponibilização através de *RSS feeds*.

Grau de importância: 1.

2.8. Recursos

Permite pesquisar por livros relacionados com o sector vinícola, fornecendo título, autor, editora, ISBN ou todos. Os resultados são mostrados numa lista com essas informações e uma imagem da capa (se disponível).

Inicialmente foi-lhe atribuído grau de importância 1 já que se considerou que esta funcionalidade poderia ser útil num contexto de turismo, por exemplo a alguém que esteja em viagem e queira num determinado momento adquirir uma publicação que permita saber um pouco mais sobre o assunto (se pesquisar por título). No entanto ficou decidido após discussão com os elementos responsáveis pela elaboração e pelos conteúdos do portal que não é considerada útil no contexto dos dispositivos móveis já que a pesquisa por livros sobre o sector deverá resultar de um processo mais aprofundado (a aquisição de uma publicação sobre o tema em princípio não será feita com base numa pesquisa tão rápida e superficial). Além disto, os critérios disponíveis para pesquisa limitariam o utilizador já que se o mesmo não tivesse uma ideia predefinida e quisesse procurar livros sobre um determinado assunto relacionado com a indústria vinícola, o único critério que lhe poderia interessar seria o título e mesmo assim seria pouco útil.

Grau de importância: 0.

3. Optimização do portal

Apesar de não ser ainda prioritário, pondera-se no futuro optimizar o portal para *browsers* de dispositivos móveis. Prevê-se que esta optimização tenha duas vertentes:

- Criação de uma versão muito simplificada do portal, adaptada a dimensões de ecrã reduzidas e com um número também reduzido de imagens;
- Alteração do portal existente de forma a estar de acordo com as normas de concepção de sites do W3C que, entre outras coisas, exigem que a navegação se faça mesmo quando não existe suporte para JavaScript ou CSS. O visitante que acesse a partir de dispositivos móveis numa primeira fase seria direccionado para a versão simplificada, mas caso entendesse que o seu *browser* era capaz de suportar a versão completa, poderia acedê-la facilmente a partir de uma hiperligação. A análise das características do dispositivo seria feita com base nos pedidos HTTP, que contêm informações como o sistema operativo e o *browser* usado no *header* User-Agent.

4. Conclusões

Depois do estudo do portal Infovini e da reunião com os elementos responsáveis pela elaboração e conteúdos do mesmo, concluiu-se que as funcionalidades a implementar no contexto dos dispositivos móveis são, por ordem decrescente de importância:

- 2.1.1. "Uma ocasião, um vinho" (grau 2), como aplicação;
- 2.1.2. "Pesquisa de vinhos" (grau 2), como aplicação;
- 2.2.1. "Rotas do vinho" (grau 2), como aplicação;
- 2.7.1. "Eventos" (grau 2), num mecanismo de sincronização com agendas;
- 3. Optimização do portal;
- 2.7.2. "Notícias" (grau 1), por *RSS Feeds*;
- 2.7.3. "Artigos de opinião" (grau 1), por *RSS Feeds*.

De entre as funcionalidades marcadas com o grau 2, considerou-se que as primeiras três são mais importantes como funcionalidades necessárias num momento muito específico, já que "Eventos" envolve visualizar uma listagem de eventos que poderia também ter sido consultada previamente. No âmbito do projecto de fim de curso a realizar pelo autor contempla-se a adaptação das três funcionalidades mais prioritárias: "Uma Ocasião, um Vinho", "Pesquisa de Vinhos" e "Rotas do Vinho".

"Notícias" encontra-se à frente de "Artigos de opinião" pelo facto de ter uma actualização mais frequente. Os artigos de opinião são normalmente mais extensos e mais aprofundados nos assuntos que abordam e a sua leitura faz por isso mais sentido num ecrã de maiores dimensões.