UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

REPRESENTAÇÃO DE PONTOS DE INTERESSE SOBRE MAPAS

Bruno Miguel Gândara Leitão Paiva

MESTRADO EM ENGENHARIA INFORMÁTICA
Sistemas de Informação
2009

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

REPRESENTAÇÃO DE PONTOS DE INTERESSE SOBRE MAPAS

Bruno Miguel Gândara Leitão Paiva

PROJECTO

Trabalho orientado pela Profª. Drª Ana Paula Boler Cláudio

MESTRADO EM ENGENHARIA INFORMÁTICA
Sistemas de Informação
2009

Agradecimentos

À Professora Doutora Ana Paula Cláudio, pela sua orientação, conhecimentos, entusiasmo, constante dedicação, disponibilidade e apoio durante a realização do trabalho, essenciais para o seu sucesso.

Trabalhar com a minha orientadora, permitiu-me ganhar e desenvolver mais conhecimentos não só em informática, mas também noutras áreas.

À Professora Doutora Cristina Catita pelo fornecimento de mapas na área de Lisboa, sem o qual não seria possível terminar o projecto e ao Professor Doutor Paulo Urbano pela sua ajuda na procura dos mesmos.

Os meus agradecimentos à FCT (Fundação para a Ciência e Tecnologia) através do projecto PTDC/EIA/69765/2006 por ter financiado o trabalho de investigação apresentado neste relatório.

À minha família pela sua ajuda e apoio incondicional durante toda a minha fase académica, e por sempre acreditarem em mim e nas minhas capacidades.

A todos os meus amigos e a todo o pessoal do Labmag pela sua ajuda, ideias e ambiente que forneceram ao longo do curso.

Aos meus pais, à minha avó, avô e à minha irmã.

Resumo

Os mapas geográficos são amplamente utilizados na Internet para visualizar vários tipos de dados geográficos.

Concebemos uma ferramenta, VisWide, que permite ao utilizador expressar o seu interesse sobre determinadas categorias de pontos de interesse que podem ser representados sobre um mapa, de um modo inteligível.

É usada uma função de grau de interesse que, com base nos valores introduzidos pelo utilizador na interface desta ferramenta, calcula um valor numérico por cada ponto de interesse armazenado na base de dados. Os pontos cujo valor esteja abaixo de um limiar definido pelo utilizador não são representados, os restantes têm representações gráficas que reflectem a sua importância.

A aplicação utiliza mapas vectoriais porque estes apresentam algumas características mais vantajosas em relação aos mapas em formato raster. Nomeadamente, os ficheiros são mais pequenos, não se perde qualidade quando se efectuam operações de zoom e suportam a adição de camadas.

Quanto à representação dos pontos de interesse concebemos um conjunto de estratégias que integrámos na ferramenta VisWide e que tentam dar resposta, fundamentalmente às seguintes dificuldades :

- Os símbolos relativos aos pontos de interesse podem ocultar zonas do mapa que o utilizador necessita visualizar;
- Apesar da aplicação da função de grau de interesse, os símbolos podem ser em número significativo em determinadas zonas do mapa, tornando díficil obter informação útil de forma rápida e simples.

Palavras-chave: Visualização, Pontos de Interesse, Função de Grau de Interesse, SVG.

Abstract

Geographical maps are widely used on the World Wide Web to visualize various kinds of spatial related data.

The main goal is to conceive a tool, VisWide, which allows users to express their interest on certain categories of points of interest, which can be represented on a map, in an intelligible way.

It uses a degree of interest function that, based on the values introduced by the user in the interface of this tool, determines a numeric value for each point of interest stored in the database. The points, whose values are underneath a certain threshold defined by the user, aren't represented and the remaining ones have graphical representations that reflect their importance.

The application uses vectorial maps because they have more advantageous features compared to maps in raster format. Particularly, the files are smaller, they don't lose quality when doing zoom operations and support the addition of layers.

As for the representation of points of interest we have fundamentally in consideration the following aspects:

- The symbols that are relative to the points of interest may conceal some parts of the map.
- Even though we use the degree of interest function there can still be a significant number of symbols in some areas of the map, making it difficult to read information.

Keywords: Visualization, Points of Interest, Degree of Interest Function, SVG.

Conteúdo

Capítulo 1 Introdução 15

1.1 Motivação e Objectivos 15

1.2 Contexto e Contribuições do Trabalho 17

1.3 Organização do documento 18

Capítulo 2 Trabalho Relacionado 19

2.1 O Projecto "Visualização de Informação Geo-Referenciada em Dispositivos Móveis" 19

2.2 O Projecto "Pesquisas Interactivas para Informação Geo-referenciada em Dispositivos Móveis" 24

Capítulo 3 Trabalho Realizado 27

3.1 SVG (Scalable Vector Graphics) 28

3.1.1 Especificação do SVG 30

3.1.2 SVG User Agents 30

3.1.3 Estrutura do SVG 31

3.1.4 SVG vs Outros Formatos 34

3.1.5 Futuro do SVG 35

3.2 GeoServer 36

3.2.1 Web Map Service (wms) 37

3.2.2 Web Feature Service (wfs) 37

3.3 Aplicação VisWide 38

3.3.1 DOM 41

3.3.2 Arquitectura do Sistema 42

3.3.3 Linguagens Server-side 43

3.3.4 Símbolos 44

3.3.5 Coordenadas 45

3.3.6 Interface 47

3.3.7 Estratégias de Representação de Pontos de Interesse 50

3.3.8 Função de Grau de Interesse 60

Capítulo 4 Conclusão e Perspectivas Futuras 62

4.1 Conclusão 62
4.2 Perspectivas Futuras 63
Bibliografia 66

Lista de Figuras

Figura 2.1 – Representação de pontos de interesse sobre um mapa no sistema MoviSys [Pombinho08] 19

Figura 2.2 – Operador de selecção [Edwardes05] 23

Figura 2.3 – Operador de simplificação [Edwardes05] 23

Figura 2.4 – Operador de agregação [Edwardes05] 23

Figura 2.5 – Operador de tipificação [Edwardes05] 23

Figura 2.6 – Operador de deslocamento [Edwardes05] 23

Figura 2.7 – Exemplo de um mecanismo baseado em checkBoxes para a escolha de múltiplos valores [Vaz08] 25

Figura 2.8 – Áreas da Interface do Protótipo MoViSys com utilização de tabs [Vaz08] 26

Figura 3.1 – Interface do VisWide 27

Figura 3.2 – Menu ampliado do VisWide 27

Figura 3.3 – Imagem raster [Bauer02] 28

Figura 3.4 – Imagem vectorial [Bauer02] 28

Figura 3.5 – Descrição de um código de uma simples elipse 31

Figura 3.6 – Imagem da elipse descrita na figura 3.5 32

Figura 3.7 – Openlayers com uma imagem da layer rios do Reino Unido 36

Figura 3.8 – Mapas com várias camadas acedendo ao Google Maps 39

Figura 3.9 – Um ficheiro SVG dividido em 9 mais pequenos [Clip] 40

Figura 3.10 – Exemplo de um simples DOM numa página web [Neumann01] 41

Figura 3.11 – Etapa prévia para obtenção do mapa 42

Figura 3.12 – Arquitectura utilizada no protótipo desenvolvido 43

Figura 3.13 – Exemplo de Símbolos (Restaurante, Museu, Estação de Serviço, Hotel) [Carto] 44

Figura 3.14 – Zonas UTM na Europa [url-Wikipedia] 46

Figura 3.15 – Exemplo de interface para a categoria Hotel com o menu ajuda 48

Figura 3.16 – Exemplo Interface para Monumentos..... 49
Figura 3.17 – Resultado da pesquisa da aplicação VisWide..... 49
Figura 3.18 – Minimização 52
Figura 3.19 – Minimização em conjunto 53
Figura 3.20 – Separação..... 54
Figura 3.21 – Tonalidades da mesma cor 55
Figura 3.22 – Agregação..... 56
Figura 3.23 – Efeito *Dock-Like*..... 57
Figura 3.24 – Efeito *Zoom* 58
Figura 3.25 – Efeito *Fade-in*/*Fade-out*..... 59
Figura 3.26 – Adaptar o tamanho dos símbolos de acordo com o seu interesse..... 60
Figura 4.1 – Mapa (a) e Interface (b) do VisWide com razão de aspecto < 1 63
Figura 4.2 – Mapa (a) e Interface (b) do VisWide com razão de aspecto > 1 63
Figura 4.3 – Ideia para cria um efeito 2.5D através de um cubo 64

Lista de Tabelas

Tabela 3.1 – Elementos do SVG 32

Tabela 3.2 – Descrição dos atributos L e M do elemento path do SVG [url-W3Path] 33

Tabela 3.3 – Diferentes formatos vectoriais na Internet [Neumann01] 34

Tabela 3.4 – SVG vs GIF vs PNG vs PDF vs FLASH [Bauer02], [url-Devarticles] 35

Lista de Acrónimos

AJAX	Asynchronous Javascript And XML
CSS	Cascading Style Sheets
DOM	Document Object Model
DTD	Document Type Definition
GIF	Graphics Interchange Format
GIS	Geographic Information System
GML	Geography Markup Language
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JPEG	Joint Photographic Experts Group
KML	Keyhole Markup Language
MIME	Multipurpose Internet Mail Extensions
OGC	Open GIS Consortium
PDF	Portable Document Format
PNG	Portable Network Graphics
PHP	PHP: Hypertext Preprocessor
SQL	Standard Query Language
SVF	Simple Vector Format
SVG	Scalable Vector Graphics
SWF	ShockWave Flash
URL	Uniform Resource Locator
WFS	Web Feature Service
WMS	Web Map Service
X3D	Extensible 3D Graphics
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language for Transformation

Capítulo 1

Introdução

1.1 Motivação e Objectivos

A visualização de informação geo-referenciada é usada, actualmente, em diversos contextos, alguns dos quais passamos a exemplificar. No âmbito do turismo, existem inúmeras aplicações disponíveis no mercado, mas existem outras áreas de aplicação como, por exemplo, na História em que são visualizados, num mapa, os locais onde ocorreram eventos relevantes; em Geografia, em que se visualiza a distribuição da população no globo ou em sub-regiões deste; na Geologia, para termos conhecimento do padrão das áreas de terreno; no Urbanismo, para identificarmos não só zonas de edificação, bem como o crescimento da população mundial; na Ecologia e na Agricultura, de modo a que se possam visualizar as zonas do coberto da vegetação de uma floresta e os seus respectivos mapas, ou a ocorrência de incêndios neste tipo de ambientes; na Meteorologia e estudos climatéricos, para nos informarmos dos diferentes tipos de cataclismos naturais que podem ocorrer na Terra como furacões, terramotos, ciclones, tempestades de neve, tsunamis ou outros; na Saúde, mais concretamente, na dispersão de enfermidades, como por exemplo a Gripe A; na Defesa, para visualizar estratégias de protecção e na Protecção Civil com a finalidade de prevenir riscos colectivos inerentes a situações de acidente grave ou catástrofe.

Por tudo o que foi exposto, existe uma grande motivação em querer melhorar, actualizar e desenvolver novas técnicas que ofereçam ao utilizador formas de visualizar informação geo-referenciada fiáveis, fáceis de interpretar e que destaquem a informação que o utilizador considere mais interessante.

O objectivo deste projecto é desenvolver um protótipo para computadores de secretária que permita visualizar informação geo-referenciada tirando partido das potencialidades deste tipo de computadores, sobretudo do maior espaço de ecrã, em comparação com o espaço de que dispõe os dispositivos móveis, do vasto conjunto de cores, da maior resolução e do maior poder de processamento e memória.

Este trabalho foi realizado no âmbito do Projecto de Engenharia Informática e teve início em Outubro de 2008. Houve que seguir algumas linhas directoras já definidas e uma vez que se enquadrou num projecto científico em curso financiado pela FCT, "Visualização de Informação Geo-Referenciada" [url-VisGeoRef].

Pretendo com este projecto, explorar estratégias de representação de pontos de interesse sob mapas. A linguagem escolhida foi o SVG e tentou-se tirar partido de todas as potencialidades oferecidas por esta linguagem, de maneira a que o utilizador possa extrair de forma fácil a informação que lhe é efectivamente útil ou interessante.

Tal como noutros trabalhos, do mesmo projecto, que precederam este, foram utilizadas técnicas de gestão de múltiplas representações e de filtragem, para obter imagens inteligíveis como resultado de pesquisas interactivas efectuadas pelo utilizador.

Neste aspecto o SVG tem um grande papel a desenrolar em termos de grafismo, cores e design. Contudo foram surgindo algumas barreiras pelo caminho, já que o SVG por ser uma tecnologia bastante recente, não se encontra implementada em todos os browsers que existem na Internet. Para alguns, por exemplo o Internet Explorer, o mais utilizado no momento, todavia não é permitido visualizar gráficos SVG, sendo necessário descarregar o plug-in da Adobe [url-Adobe].

Estes problemas de compatibilidade dificultaram o trabalho, na medida em que alguns gráficos em SVG, poderão apresentar certas limitações em determinados browsers. Alguns poderão não carregar no Mozilla-Firefox, outros poderão aparecer com animações distorcidas.

1.2 Contexto e Contribuições do Trabalho

Este trabalho foi realizado no ano lectivo 2008/2009, no âmbito do Mestrado em Engenharia de Informática da Faculdade de Ciências da Universidade de Lisboa (FCUL). O trabalho apresentado foi realizado no contexto do projecto Visualization of Geo-referenced Information, que tem como coordenadora a Professora Doutora Maria Beatriz Carmo e que faz parte de um dos grupos de investigação do Departamento de Informática, LabMag. O trabalho foi proposto e orientado pela Professora Ana Paula Cláudio.

Durante este trabalho foi escrita uma contribuição para a comunidade científica na forma de poster. Foi submetido no 17º Encontro Português de Computação Gráfica, a decorrer nos dias 29 e 30 de Outubro de 2009, na Universidade da Beira Interior, Covilhã. O título do poster é Estratégias para a Representação de Pontos de Interesse sobre Mapas e foi aceite para apresentação no 17º EPCG [Paiva09].

1.3 Organização do documento

Este documento está organizado da seguinte forma:

- Capítulo 1 – Introdução

  Neste capítulo são apresentadas as motivações e objectivos da tese, as contribuições feitas durante o trabalho e a organização deste documento.

- Capítulo 2 – Trabalho Relacionado

  Neste capítulo apresenta-se o resultado do estudo efectuado sobre o desenvolvimento do projecto até à data de início deste trabalho.

- Capítulo 3 – Trabalho Realizado

  Neste capítulo é apresentado o trabalho desenvolvido, incluíndo a aplicação VisWide.

- Capítulo 4 – Conclusão e Perspectivas Futuras

  São tiradas conclusões e apresentadas perspectivas para o futuro, incluindo melhorias ao trabalho e objectivos delineados como trabalho futuro.

Capítulo 2

Trabalho Relacionado

Na fase inicial deste trabalho houve necessidade de compreender o trabalho desenvolvido anteriormente no projecto. Nas duas secções que se seguem resumem-se as duas componentes fundamentais: O protótipo MoviSys, desenvolvido pelo Mestre Paulo Pombinho e um segundo protótipo desenvolvido pela Mestre Ana Margarida Vaz.

Ambos os protótipos foram desenvolvidos durante os respectivos trabalhos de tese.

2.1 O Projecto "Visualização de Informação Geo-Referenciada em Dispositivos Móveis"

O protótipo MoViSys [Pombinho08] permite visualizar informação geo-referenciada em dispositivos móveis, usando mapas do Google Maps Services, e assinalando sobre estes, pontos de interesse de acordo com as preferências expressas pelo utilizador (monumentos, hotéis, etc). (Figura 2.1)

Figura 2.1 – Representação de pontos de interesse sobre um mapa no sistema MoviSys [Pombinho08]

Existem outros trabalhos na mesma área, nomeadamente o descrito em [Freitas05], mas o trabalho desenvolvido no protótipo MoViSys é distinto, no sentido em que este último é feito para dispositivos móveis e a abordagem seguida no que diz respeito à função de grau de interesse também é distinta, como veremos adiante.

Em [Pombinho08] efectuou-se uma pesquisa de vários conceitos na área da Geo-Visualização, sobretudo, em relação a dispositivos móveis o que nos deixa algum espaço para desenvolver projectos com computadores *desktop*.

Foram expostas várias técnicas no trabalho e estudadas várias aplicações como o Google Maps, Yahoo Maps, e Virtual Maps. As tarefas que permitem a um utilizador aceder à informação e, deste modo, facilitar-lhe a vida são: Selecção, Sniffing, Visualização remota e Pesquisa. Esta última suporta quatro tipos de questões: Localização, Proximidade, Navegação e Evento.

Para visualizar a informação há três tipos de dados que é necessário obter: as coordenadas da posição que o utilizador deseja visualizar, a imagem do mapa que contenha estas mesmas coordenadas e a informação sobre os pontos de interesse existentes nessa área. Estes dados são usados para obter a imagem final, apresentada ao utilizador. A obtenção de coordenadas é feita através de satélites, um conjunto de quatro satélites que fornecem um sinal com as coordenadas desejadas. Em relação à obtenção de mapas, poder-se-iam escolher mapas vectoriais ou de tipo raster, tendo a escolha recaído nos mapas do tipo raster. Quanto aos pontos de interesse foram utilizados vários e de diversos níveis de complexidade, consoante o grau de detalhe que se pretende na aplicação.

Foram indicados vários exemplos de sistemas, entre eles, o Google Maps e o Google Earth, o Yahoo!Maps e o Live Search Maps da Microsoft para aplicações de secretária. O TomTom Navigator, o sistema Navigon e o NDrive para aplicações de navegação e depois outras como o WebPark, GiModig, MAGDA, e finalmente o Google Maps mobile.

Foram expostas várias funções de grau de interesse, como por exemplo, a função de George Furnas (olho-de-peixe), na qual existe um balanço entre os detalhes locais e contexto global. Esta técnica é adequada para quando há uma grande quantidade de informação. A função de grau de interesse é definida deste modo:

$$DOI_{fisheye}(x|y) = API(x) - D(x,y)$$

onde $DOI_{fisheye}$ corresponde ao interesse num dado objecto x, sendo que o ponto de foco é y; API(x) corresponde à importância *a priori* de x e D(x,y) é a distância entre x e

o ponto de foco y. Assim, o interesse num dado ponto cresce com a importância *a priori* e diminui quanto mais afastado estiver do foco.

Uma outra função de grau de interesse é a utilizada em [Martins02]. Esta função foi criada para corrigir dois problemas, já que a função de Furnas é aplicada a bases de dados relacionais. Para eliminar a dificuldade de atribuir uma importância *a priori* a cada um dos seus elementos, e eliminar o problema da definição do ponto foco, que pode não ser um elemento da estrutura no caso de uma base de dados relacional, foram reformulados os conceitos de ponto foco, distância ao foco e importância *a priori*. A função de grau de interesse redefinida é:

$$\text{DOI}\left(O_i, O_f, O_p\right) = f\left(API\left(O_i, O_f\right), \text{Dist}\left(O_i, O_f\right)\right)$$

onde *Oi* designa cada um dos objectos a visualizar e *f* é uma função que tem de ser crescente no primeiro termo e decrescente no segundo.

Foi incluído, na aplicação, um mecanismo de filtragem baseado em critérios semânticos, com vista à redução do volume de informação visualizado, de acordo com os interesses dos utilizadores, permitindo deste modo, uma melhor selecção da informação suprimindo a informação menos relevante.

Apesar da função de Furnas ter sido a função de grau de interesse inicialmente utilizada no MoViSys, foi-se tornando cada vez mais semelhante à função de Keim que foi usada no sistema VisDB [Keim94], com o objectivo de facilitar o processo de especificação das interrogações, através da utilização de técnicas de visualização que permitam ao utilizador obter mais informação sobre os resultados das suas interrogações.

Foi mencionado o trabalho de Reichenbacher [Reichenbacher04], uma *framework* conceptual para serviços de geo-visualização, que assegura uma comunicação eficiente e a usabilidade deste tipo de serviços através de sistemas de adaptação ao utilizador.

No protótipo MoViSys são usadas duas funções de grau de interesse: uma mais simples e uma outra versão estendida.

Para definir a função de grau interesse foi usada a seguinte fórmula:

$$DOI\ (\rho_j) = \frac{\sum_{i=i}^{k} UI(a_i, \rho_{ji})}{k} \in [0,1]$$

onde K corresponde aos diferentes atributos, como por exemplo o ar condicionado ou serviço de quarto, ou o número de estrelas que um hotel possui etc, o conjunto A representa

os K valores de atributos seleccionados pelo utilizador e o conjunto P os n pontos de interesse existentes.

A função UI($a_{i,} p_{ji}$) é dada pela fórmula:

$$UI(a_{i,} p_{ji}) = 1 - Dist\ (a_{i,} p_{ji}) \times w_i,\ w_i \in [0,1]$$

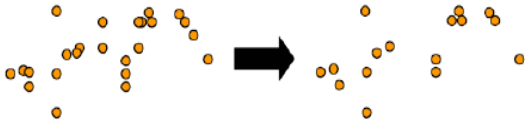
Neste caso w_i será a importância escolhida pelo utilizador para o atributo e a Dist($a_{i,} p_{ji}$) será 0 no caso de $a_i = p_{ji}$ e 1 no caso de $a_i \neq p_{ji}$

Para a normalização da função de distância geográfica na forma simples, é necessário ter em conta que os valores máximos e mínimos utilizados variam consoante a posição geográfica de interesse escolhida pelo utilizador. Assim, caso a posição de interesse para o utilizador esteja dentro da área visível no ecrã, os valores utilizados para a normalização serão as coordenadas máximas e mínimas da área visível. Caso a posição de interesse esteja fora da área visível, as coordenadas máximas e mínimas utilizadas serão os extremos de um rectângulo centrado na área visível, e expandido para incluir a posição de interesse. De referir, no entanto, que o tamanho deste rectângulo não pode nunca ser inferior à área visível, de modo a impedir que pontos de interesse visíveis fiquem fora da área de normalização.

A versão estendida foi feita tendo em conta as diferentes importâncias que cada categoria pode ter, para que na procura de duas categorias diferentes, uma não se sobreponha à outra. Foi feita, assim, uma extensão à fórmula da função de grau de interesse, adicionando-se um peso às categorias. Outro problema que existe na função simples é o de esta apenas permitir a selecção de um único valor em cada atributo, de modo que foram estendidas as funções de distância.

Foram analisados vários operadores de generalização, um tópico muito importante na geo-referenciação, já que num único espaço é possível que existam, por exemplo, várias áreas comerciais, o que origina a sobreposição de símbolos. Por exemplo, o sistema MetaCarta diminui o número de símbolos representados e, assim, aumenta a legibilidade da imagem.

Foram utilizadas vários operadores de generalização , como o operador de selecção, de simplificação, de agregação, o de tipificação e por último, o de deslocamento.

	**Figura 2.2 – Operador de selecção [Edwardes05]**
	**Figura 2.3 – Operador de simplificação [Edwardes05]**
	**Figura 2.4 – Operador de agregação [Edwardes05]**
	**Figura 2.5 – Operador de tipificação [Edwardes05]**
	**Figura 2.6 – Operador de deslocamento [Edwardes05]**

O primeiro operador identifica quais os objectos que devem ser representados, omitindo assim conflitos. Em relação ao operador simplificação, é similar ao primeiro, mas com a diferença de procurar um subconjunto de objectos que seja próximo do conjunto inteiro. A agregação permite substituir um ou mais símbolos do mesmo tipo, por um outro, diminuindo assim o número de símbolos no ecrã. Quanto ao de tipificação, difere do de agregação, pelo facto de utilizar as relações espaciais entre os objectos. É utilizado quando não existe uma relação semântica entre si, pertencendo a categorias distintas. O último operador afasta os símbolos quando existe uma grande sobreposição destes (por exemplo, quando se efectua uma desagregação).

Os resultados obtidos nesta área de pesquisa demonstram que a combinação dos operadores de generalização com a utilização de funções de filtragem, permite aos

utilizadores visualizar um mapa mais claro e nítido, o que lhe permite mais facilmente compreender o significado da informação visualizada.

Para ajudar a compreender a informação apresentada foi também desenvolvida a simbologia. Os símbolos foram subdivididos em individuais e em agregados, tendo cada um dos seus símbolos individuais, um detalhe diferente e, para os agregados, uma ou várias categorias. No protótipo MoViSyS a cor usada para todos os símbolos foi o azul, de modo a serem facilmente detectáveis, e foram usadas várias técnicas para compensar a falta de espaço nos dispositivos móveis, como, por exemplo, o efeito de pilha, a criação de um símbolo composto pelas quatro categorias mais numerosas e um ícone '+' no símbolo que contenha o melhor ponto de interesse de cada categoria. De referir, que existe um modo 'andar' que torna os símbolos maiores quando o utilizador se encontra em movimento.

Na interface, foi ainda incorporada uma caixa de informação, que apresenta os detalhes sobre cada ponto de interesse.

No final do projecto, foi realizado um plano de avaliação, composto por vários testes, que permitiram avaliar o protótipo do sistema MoViSys. O primeiro teste foi um pré-questionário, composto por um conjunto de perguntas sobre o perfil dos utilizadores e a sua experiência na utilização deste tipo de dispositivos. O segundo teste foi baseado num guião que tinha como objectivo a realização de um conjunto de tarefas seguido de um conjunto de perguntas de forma a obter as opiniões por parte dos utilizadores. Estes testes levaram a concluir que a função de interesse e os operadores de generalização são considerados úteis e os utilizadores preferem utilizar a aplicação com as funcionalidades activas.

2.2 O Projecto "Pesquisas Interactivas para Informação Geo-referenciada em Dispositivos Móveis"

No projecto [Vaz08], procede-se ao desenvolvimento de uma interface para efectuar pesquisas interactivas de informação geo-referenciada em dispositivos móveis.

O utilizador através desta interface visualiza os pontos de interesse que estão contidos numa base de dados, e esta informação é actualizada de acordo com a solicitação do utilizador.

Foram utilizadas técnicas de visualização de forma a organizar várias representações com diferentes níveis de detalhe, a posição dos símbolos e também a identificação das características que definem as representações simbólicas.

Como este projecto foi desenvolvido para dispositivos móveis, foram implementadas várias técnicas de gestão de espaço de informação. Algumas das aproximações usadas nesta área, são a *Broad*, *Narrow*, *MessageBox* (caixas de mensagens), *Tabs* (separadores) e menus *pull-down* e *pop-up*.

Figura 2.7 – Exemplo de um mecanismo baseado em checkBoxes para a escolha de múltiplos valores [Vaz08]

Foi também empregue uma função de grau de interesse de modo a reduzir o número de símbolos apresentados no ecrã. Neste aspecto, o projecto dá a possibilidade ao utilizador de definir o número máximo de pontos de interesse a visualizar, permitindo ao mesmo tempo configurar o tamanho de células da grelha (o mapa está subdividido segundo uma grelha com células rectangulares) a utilizar pelos operadores de generalização, e o número máximo de pontos de interesse apresentados em cada célula.

É, importante, notar que foram usadas várias 'Query devices' que permitiram utilizar pesquisas dinâmicas de modo a agilizar a definição e a análise da informação geo-referenciada desejada pelo utilizador.

Como parte final deste trabalho foi feita uma avaliação do protótipo, ou seja, foram apresentados testes de usabilidade, com o intuito de avaliar a interface do sistema MoViSys, assim como os resultados dos mesmos. Estes testes foram constituídos por observação directa do utilizador ao explorar o protótipo, através de um questionário, Protocolo "Pensar em voz alta (*Thinking Aloud Protocol*) em que foi pedido um parecer ao utilizador de todos os seus pensamentos à medida que ia observando a interface,

Cognitive walkthroughs, isto é, um guião de tarefas que o utilizador teve de realizar e finalmente um método *contextual inquiry* que consiste numa entrevista informal de forma a identificar as opiniões subjectivas sobre o protótipo. Os testes foram bastante satisfatórios, evidenciando uma facilidade de aprendizagem, utilização e interacção por parte do utilizador.

Figura 2.8 – Áreas da Interface do Protótipo MoViSys com utilização de tabs [Vaz08]

Capítulo 3

Trabalho Realizado

Neste capítulo é descrito o protótipo VisWide e as suas funcionalidades, assim como as tecnologias usadas no seu desenvolvimento. A secção 3.1 é dedicada à linguagem SVG, a secção 3.2 é relativa ao servidor de mapas GeoServer e a secção 3.3 descreve o protótipo VisWide. As figuras 3.1 e 3.2 mostram a interface do protótipo.

Figura 3.1 – Interface do VisWide

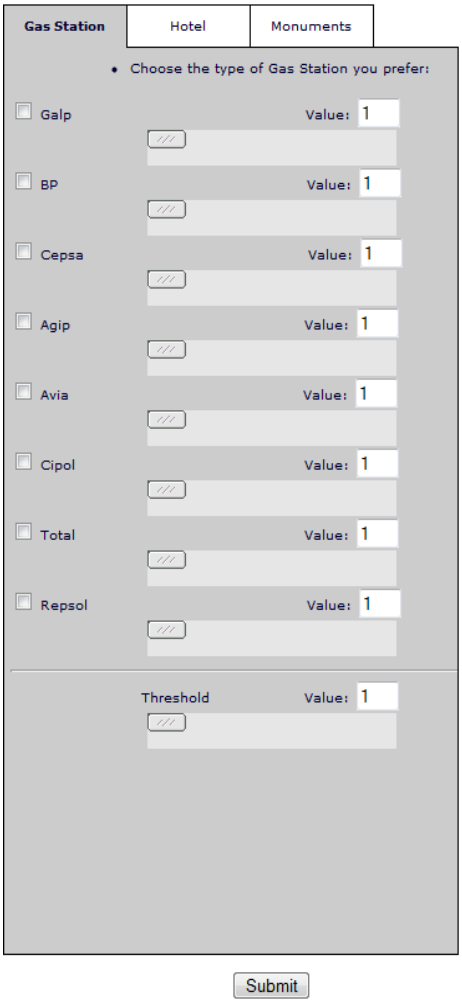


Figura 3.2 – Menu ampliado do VisWide

3.1 SVG (Scalable Vector Graphics)

O Scalable Vector Graphics (SVG) é descrito por ter um papel importante a desenrolar no projecto pois permite representar todos os elementos produzidos por gráficos e mapas cartográficos. Para além do X3D (padrão aberto para distribuir conteúdo 3D), é uma das principais linguagens da família do XML, vocacionada para descrever gráficos vectoriais de duas dimensões [url-W3]. Permite aos programadores criar interfaces interactivas para documentos de XML de vários tipos. É utilizado, principalmente, para fazer gráficos de páginas da internet e pode substituir o Flash [url-Xul].

Actualmente, a maior parte das páginas na Internet são baseadas em imagens raster (ex: JPEG, PNG, BMP, GIF) que consistem em imagens de grande tamanho, sem uma boa qualidade de escalabilidade (zoom) e extensibilidade, isto é durante a implementação é necessário ter em consideração um desenvolvimento futuro de modo a que possamos manipular novas actualizações.

Figura 3.3 – Imagem raster [Bauer02]

Figura 3.4 – Imagem vectorial [Bauer02]

Paul Presco utiliza a página do Google para ilustrar a importância do SVG [Presco03]. Segundo este autor, esta página poderia ser mais rápida se utilizasse gráficos vectoriais. Vamos tomar em conta a famosa página da internet: Google. É famosa não só pelas suas pesquisas, mas também pela sua rapidez a carregar. Contudo, esta página poderia ser mais rápida se utilizasse gráficos vectoriais [Presco03]. A página do Google consiste basicamente em imagens e num documento HTML. Se usasse SVG, poderia usar simplesmente um único documento XHTML/SVG, o que permitiria o descarregamento numa única transacção de rede, em vez de várias. O código do lado do cliente poderia manipular e transformar os gráficos e texto juntos. Após termos realizado a pesquisa que queríamos, o Google leva-nos para uma outra página com os resultados dessa pesquisa e aparece uma imagem igual à que havia na página principal, mas com um tamanho diferente. Como as imagens raster têm uma fraca escalabilidade, o que acontece é que é carregado um gráfico completamente diferente do anterior.

Apesar da imagem grande ser guardada em cache, a versão mais pequena é descarregada a partir do zero. Com o SVG, a imagem seria apenas reutilizada e redimensionada para o tamanho correcto. Por vezes, o Google utiliza também imagens correspondentes a épocas festivas. Ao utilizarmos SVG, teríamos somente que adicionar outras camadas ou superposições, sem ter que usar novos gráficos, o que levaria a um aumento de desempenho.

Se o SVG pode melhorar grandemente a eficiência do Google, também poderia melhorar o acesso a sites com estruturas mais complexas. Teríamos um melhoramento de desempenho significativo por toda a Internet [Presco03].

As principais características do SVG são:

- Gráficos de alta qualidade com controlo e estrutura precisa.
- Gráficos gerados dinamicamente através de dados XML em tempo-real
- Gráficos interactivos usando XSLT, JavaScript, DOM, Java, Visual Basic, etc.
- Gráficos integrados com outros membros da família XML
- Gráficos facilmente personalizados a todos os utilizadores
- Gráficos automaticamente optimizados para dispositivos móveis, PDAs, etc.
- Gráficos facilmente actualizados (desde que o *design* esteja separado do conteúdo)
- Gráficos que permitem a aplicação de zoom sem distorção
- Gráficos que possibilitam internacionalização e localização (uso de muitas linguagens) ou seja são facilmente migrados para diferentes linguagens
- Gráficos em formato de texto legíveis ao humano.
- Totalmente compatível com CSS

Além destas características temos como vantagem o facto de ser livre de normas, ou seja, é dada a possibilidade de escolher um plug-in e, quando ocorrem erros no código são indicados o número de linha e de coluna correspondentes.

O SVG é uma linguagem interactiva que permite trabalhar com outras linguagens como Java, C++, Perl e PHP, mas é com o JavaScript que é mais frequentemente utilizada. Permite o uso de animações através da alteração de posicionamento, efeitos de texto e cor. Os documentos SVG têm um número de elementos estruturais, que apesar de não especificarem um efeito gráfico, fornecem um sentido de organização, ou seja é possível agrupá-los e fornecem também interactividade.

Por tudo isto, o SVG é uma excelente ferramenta de trabalho para criar interfaces visuais interactivas e extensíveis, ou seja com capacidade de crescer através da adição de novos componentes.

3.1.1 Especificação do SVG

No SVG e em qualquer outro documento XML, é necessário adicionar uma declaração DOCTYPE. Na linguagem SVG existem cinco tipos diferentes de especificação para declarar o DOCTYPE: o SVG 1.0 - DTD, o SVG 1.1 Full - DTD, o SVG 1.1 Basic - DTD, o SVG 1.1 Tiny - DTD e o SVG 1.2 Tiny - DTD. Além do DOCTYPE é, também necessário especificar o tipo MIME, de maneira a que o *user agent* (que veremos mais adiante), reconheça o SVG.

O MIME foi originalmente criado para configurar o tipo de dados, num anexo de um correio electrónico [Wenz02]. O tipo MIME do SVG é image/svg+xml. Em termos de configuração é bastante simples, pois basta adicionar ao ficheiro mime.types a seguinte entrada: image/svg+xml. Se o servidor web não é configurado para enviar automaticamente o tipo MIME, ou se estamos a utilizar ficheiros SVG com a extensão PHP, que é o caso deste projecto, então necessitamos de dizer ao *user agent* qual o tipo de dados que está a receber. Torna-se, também, necessário incluir a declaração DOCTYPE para ficar de acordo com a especificação XML e, assim, validar o nosso documento.

O SVG DTD actual encontra-se em SVG W3C, assim como o *default namespace* do SVG e o *xlink namespace,* de modo a podermos fazer ligações tanto externas como internas [url-W3]. Como o SVG é XML, necessitamos igualmente de uma declaração XML.

3.1.2 SVG User Agents

Um *user agent* é uma aplicação que exibe conteúdo da web, por exemplo, um *browser* [Argerich02]. Por vezes, este *user agent* necessita de um plug-in para permitir a visualização da informação. Existem SVG *user agents*, tanto nativo como *plug-in,* mas no caso deste projecto, utilizamos um *browser* com um agente nativo.

- **Adobe SVG Viewer Plug-in**

  É o *plug-in* de SVG mais famoso que existe e funciona em várias plataformas como Linux, Windows e Macintosh, funcionando também sem dificuldades com Internet Explorer 4 ou superior, Real Player 8,

Netscape 4.5-4.77, Mozilla, Opera 5.x, Google Chrome, Apple Safari. As versões que existem actualmente são ASV3 e ASV6. É baseado em Java

- **Webkit/KSVG Plug-in**

  Funciona como *plug-in* ou visualizador autónomo.

- **Mozilla Firefox**

  Suporte SVG incluído no browser.

- **Opera**

  Suporte SVG incluído no browser.

- **Outros**

  Novas implementações SVG estão continuamente a aparecer, incluindo visualizadores, editores, exportadores, conversores e geradores quer para dispositivos móveis quer para *Desktop*.

3.1.3 Estrutura do SVG

Através do SVG, temos inúmeras possibilidades para a criação de símbolos. As capacidades do SVG podem estender-se à criação de imagens, de textos e através destas, dar forma a novas imagens e textos através de animações e efeitos, mas também oferecem interactividade, organização e controlo [Geroimenko05]. Temos, portanto, a oportunidade de criar formas básicas como linhas, rectângulos, círculos, elipses, polígonos, textos de vários tamanhos e aspectos, imagens, invocar atributos de camada e opacidade. Alguns destes elementos básicos têm possiblidade de receber parâmetros suplementares, como cantos redondos.

A figura 3.5 apresenta um código de uma simples elipse.

```
<?xml version="1.0" standalone="no"?>   //declaração xml
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics
/SVG/1.1/DTD/svg11.dtd"> //declaração DOCTYPE que identifica o DTD
<svg width="90%" height="90%" xmlns="http://www.w3.org/2000/svg">
<ellipse cx="300" cy="150" rx="200" ry="80" style="fill:rgb(200,100,50);"/>
</svg>
```

Figura 3.5 – Descrição de um código de uma simples elipse

A figura 3.6 mostra a imagem resultante do código apresentado.

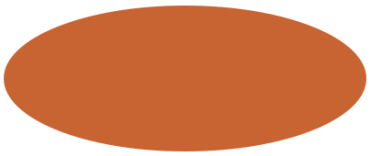


Figura 3.6 – Imagem da elipse descrita na figura 3.5

Quando se cria um objecto em SVG é necessário definir vários atributos de acordo com a finalidade que desejamos aplicar-lhe. Na tabela 3.1 mostramos os atributos que podemos encontrar no SVG.

id	Identificador único para referenciar o objecto
x, y	Posição onde se encontra
xlink:href	Hiperligação de onde o objecto vai herdar os seus atributos. É, usualmente, utilizado para ir buscar imagens a uma determinada localização (local ou na *web*)
fill	Preencher o objecto com uma determinada cor
opacity	Dar um determinado valor de opacidade ao objecto
style	Permite adicionar mais informação como o *stroke* e o *fill*, tipo de letra, etc...
height, width	Tamanho de um objecto
g	Agrupar um conjunto de objectos, o que permite realizar transformações para vários objectos como se fosse somente para um único
path	Usado para desenhar formas mais avançadas, combinando arcos e linhas. É um dos elementos mais avançados e versáteis do SVG
transform	Utilizado para efectuar rotações, translacções e/ou aumento/diminuição de objectos
image	Inclui dentro de um ficheiro SVG imagens vectoriais ou raster
a	Cria ligações nas imagens SVG que funcionam exactamente como ligações HTML

Tabela 3.1 – Elementos do SVG

O elemento mais importante é o path, que é ao mesmo tempo o mais flexível [Neumann03].

Sobre o path, é importante referir que o mapa de Lisboa usado neste trabalho é todo feito através deste elemento, e como tal é tudo especificado dentro do atributo "d". Este atributo contém comandos de desenho, que podem consistir em ordens de mover ("m") e linha ("l"), entre outras como curva, arco e *closepath*. Estes comandos são enviados para uma "caneta virtual" que vai, então, realizar os contornos do mapa, dando forma ao nosso mapa de Lisboa. Existem outros comandos, mas os principais encontram-se nestes dois, visto que no mapa apenas queremos indicar de onde, ou até onde é que vai uma rua, ou avenida, etc.

A tabela 3.2 dá uma descrição pormenorizada de dois dos atributos mais usados do elemento path.

Comando	**Nome**	**Parâmetros**	**Description**
M(absoluto) **m** (relativo)	moveto	(x y)+	Inicia um novo sub-path numa dada coordenada (x,y). **m** neste caso significa relativo à posição do ultimo comando que foi desenhado (também chamado "ponto corrente"). Enquanto que **M** diz respeito a coordenadas absolutas
L(absolute) **l** (relative)	lineto	(x y)+	Desenha uma linha desde o "ponto corrente" para a dada coordenada (x,y) que se torna então o novo ponto corrente. Tal como no comando anterior **L** indica coordenadas absolutas e **l** coordenadas relativas. Pode ser usado um conjunto de coordenadas para formar um polígono. O ponto corrente será o ultimo conjunto de coordenadas fornecidas.

Tabela 3.2 – Descrição dos atributos L e M do elemento path do SVG [url-W3Path]

3.1.4 SVG vs Outros Formatos

Desde 1993 que têm sido realizados esforços para estabelecer um formato baseado em vectores. Tudo começou com o SVF (Simple Vector Format), passando por outros como Macromedia Flash e PDF, além de Java-Applets. Nenhum destes é totalmente apropriado para requisitos cartográficos, o que explica em parte o facto de nos dias de hoje, existerem servidores-mapas com uma fraca qualidade em termos de (carto-)grafia. [Neumann01]. Como exemplo de outros formatos, temos também o Postscript e o SWF.

Existem duas alternativas ao SVG para distribuir gráficos vectoriais a clientes web: Macromedia Flash e WebCGM (recomendação W3C), contudo nenhum supera a funcionalidade e flexibilidade fornecida pelo SVG [Winter01].

De seguida, é mostrada uma tabela onde se podem encontrar diferentes formatos gráficos e suas especificidades.

Formato	Módulos de Visualização	Usabilidade	Nível de Interactividade	Formato interno
SVG	browser/plugin	raro (novo)	4	ascii
DWF	plugin/applet	raro	2	binário
Flash	plugin	frequente	3	binário
PDF	plugin	frequente	1	binário/ascii
SVF	plugin	desactualizado	1	binário
PGML	a	a	3	ascii
WebCGM	browser/plugin	raro	2	binário
HGML	a	a	1	ascii
DrawML	a	a	0	binário
VML	browser	raro	1	ascii
Java2D b	applet	raro (novo)	4	binário
ActiveX b	browser	frequente	4	binário

a: formato específicado. mas não implementado

b: não é um formato gráfico, contudo é uma biblioteca de gráficos para os programadores

1: zoom, camadas, ligações nos objectos — 2: scripts externos acedendo a gráficos

3: animações — 4: Controlo total nos objectos e animações

Tabela 3.3 – Diferentes formatos vectoriais na Internet [Neumann01]

Na tabela que se encontra abaixo, podemos ver a comparação entre SVG e outros formatos gráficos [Bauer02].

	SVG	GIF	PNG	PDF	FLASH
Vectorial	Sim	-	-	Sim	Sim
Transparência Canal Alfa	Sim	-	Sim	Sim	Sim
Precisão ao Pixel no esboço	Sim	Sim	Sim	Sim	Sim
Recursos Partilhados através de múltiplos ficheiros	Sim	-	-	-	-
Standard Aberto	Sim	-	Sim	-	-
Filtro de efeitos	Sim	-	-	-	-
Localização de erro	Sim	-	-	-	-
Extensibilidade	Sim	-	-	-	-
Animação	Sim	Sim	-	-	Sim
Interacção	Sim	-	-	Sim	Sim
Ficheiro comprimido	Sim	Sim	Sim	Sim	Sim

Tabela 3.4 – SVG vs GIF vs PNG vs PDF vs FLASH
[Bauer02], [url-Devarticles]

3.1.5 Futuro do SVG

O SVG é uma linguagem gráfica que de acordo com alguns autores terá um importante papel a desenrolar na próxima geração de Internet, em termos de visualização de informação e criação de interfaces gráficas [Geroimenko05].

Encontra-se a ser desenvolvido e suportado por todas as grandes organizações e companhias de software e grafismo, que estão relacionadas com a *web*: Adobe, Apple, Autodesk, Bit-Flash, Corel, HP, IBM, ILOG, Inso, Kodak, Macromedia, Microsoft, Netscape, Oasis, Open Text, Oxford University, Quark, RAL, Sun-Microsystems, W3C e Xerox. Este facto garante que no futuro será dado um largo suporte, não só no que diz respeito a desenvolvimento de importação e exportação de filtros, bem como a conversores e visualizadores. Dado que o SVG está bastante bem documentado e é um *standard* XML, é possível gerar e converter, facilmente, os nossos próprios *scripts* e programas. É também esperado que o SVG seja brevemente suportado pelos *softwares*

GIS (Sistemas de informação geográfica), e devido à sua aplicação em projectos de *web,* espera-se que obtenha grande importância de um modo geral no formato de grafismo. Depois do Adobe Illustrator e PDF, o SVG é um dos poucos formatos gráficos ASCII bem documentados e comparado com o PDF oferece, decididamente, mais possibilidades [Neumann03]. Espera-se também que, num futuro próximo, as versões dos *browsers* tenham já os interpretadores SVG implementados como *standard,* o que facilitaria em grande parte a sua visualização aos utilizadores da internet.

3.2 GeoServer

O GeoServer é um servidor *Open Source* desenvolvido em Java, que permite aos utilizadores partilhar e editar informação geo-espacial. O GeoServer tem evoluído para facilitar a conectividade com serviços como o Google Earth e Nasa World Wind, Google Maps e Windows Live Local. GeoServer é a implementação referenciada da Open Geospatial Consortium Web Feature Service e também permite implementar especificações Web Map Service e Web Coverage Service [url-GeoServer].

Uma das suas características é ler ficheiros expressos numa grande variedade de formatos incluíndo PostGis, Oracle Spatial, ArcSDE, DB2, MySQL, Shapefiles, GeoTIFF, GTOPO30, ECW, MrSID e JPEG2000. Neste projecto, foram utilizados mapas em formato shapefile. Através de alguns protocolos, é possível ler informações nestes formatos e produzir ficheiros KML, GML, Shapefile, GeoRSS, PDF, GeoJSON, JPEG, GIF, SVG, PNG etc [url-Wikipedia].

Além disso, é possível editar dados através do WFS transactional profile.

O OpenLayers é um cliente (também *Open Source*), que permite visualizar o mapa e as suas camadas como se pode verificar na figura 3.7.

Figura 3.7 – Openlayers com uma imagem da layer rios do Reino Unido

3.2.1 Web Map Service (wms)

É um protocolo que permite realizar pedidos de imagens geradas a partir de dados geográficos. Portanto, tem como objectivo poder colocar um mapa num ficheiro ou numa página da internet. Estes mapas são normalmente produzidos em formatos como PNG, GIF, JPEG ou também como SVG. Interactua com o seu cliente através do protocolo HTTP.

Este protocolo especifica um número diferente de pedidos, dois dos quais são necessários a qualquer servidor WMS:

- **GetCapabilities –** retorna os parâmetros acerca do WMS e as camadas disponíveis
- **GetMap –** retorna a imagem do mapa, fornecendo os parâmetros necessários.

Os restantes tipos de pedidos que o WMS fornece são opcionais e incluem:

- **GetFeatureInfo –** retorna informação acerca de características particulares
- **DescribeLayer –** retorna um XML com a descrição de uma camada do mapa
- **GetLegendGraphic –** retorna uma legenda (imagem/ícone) para a camada pedida, com etiquetas.

3.2.2 Web Feature Service (wfs)

É um protocolo que permite realizar pedidos de dados geográficos (vectores). A especificação WFS define interfaces para descrever manipulação de dados, de características geográficas. As operações relacionadas com manipulação de dados incluem a habilidade de criar uma instância, de actualizá-la, de apagá-la e realizar uma interrogação para receber a instância. WFS descreve operações de transformação de dados e interrogações. O cliente gera um pedido e publica-o num servidor usando HTTP. Este servidor executa, então, o pedido. A especificação WFS utiliza HTTP como um distribuidor apesar de não ser um requisito rígido.

Existem as seguintes operações que devem ser implementadas quando se escreve WFS:

- **GetCapabilities –** esta interrogação determina as opções disponíveis**.**
- **DescribeFeatureLayer –** retorna o esquema XML que permite ao cliente WFS analisar os resultados.
- **GetFeature –** permite realizar a interrogação. Parâmetros como a *bounding box* e outros filtros, devem ser passados e o serviço WFS retorna então um GML com os resultados que contêm todos os atributos e toda a geometria.

3.3 Aplicação VisWide

No início do projecto, foram efectuadas leituras de documentos de modo a compreender o estado de desenvolvimento do projecto de Informação Geo-Referenciada. A partir destas leituras, foram estudados outros artigos cada vez mais específicos, passando a investigar-se artigos, nomeadamente sobre SVG. Após a realização de um estudo nesta matéria, deu-se início à realização de pequenos exemplos de maneira a pôr em prática os conhecimentos adquiridos, e as ideias que se gostariam de ver implementadas. Prosseguiu-se, então, na investigação de servidores de mapa, como o SUAS Map Server e o GeoServer. O servidor adoptado foi o GeoServer por termos considerado que é fácil de usar e de configurar, gera mapas em formato SVG, permite adicionar outras camadas, e trabalhar com protocolos como *Web Map Service* (WMS) *Web Feature Service* (WFS) *Open Geospatial Consortium* (OGC) e *Web Coverage Service* (WCS), entre outros.

Concluiu-se que através do GeoServer, podemos trabalhar com clientes como o MapBuilder e o OpenLayers. O MapBuilder é mais robusto e tem mais recursos, porém é mais complexo e tem uma curva de aprendizagem maior. Quanto ao OpenLayers, é bem mais simples, e no que diz respeito à integração com o GeoServer, pode dizer-se que estão ambos ao mesmo nível, pois basta apenas passar o caminho do WMS e as camadas desejadas. A partir deste ponto, é possível seleccionar a camada no mapa que pretendemos visualizar, e até aceder a mapas do Google Maps, Yahoo Maps e Bing Maps da Microsoft.

Figura 3.8 – Mapas com várias camadas acedendo ao Google Maps

Em relação aos mapas, considerando que os mapas raster são fáceis de obter, Reichenbacher [Reichenbacher02] argumenta, porém, que uma base vectorial tem várias vantagens em comparação com o formato raster, como um tamanho de ficheiro menor e maior flexibilidade em respeito a modificação dinâmica do mapa e a adaptação do mesmo a diferentes contextos. É, também, defendido que o SVG tem sido descoberto por uma grande variedade de projectos como um formato gráfico vectorial adequado para a implementação de mapas em servidores web, assim como em dispositivos móveis [Reichenbacher02], [Pavlicko04, Peterson04]. Como aspectos positivos, temos também a característica da escalabilidade em que nenhuma informação se perde durante o zoom e o facto dos objectos num mapa baseado em SVG, como por exemplo, texto, se manterem legíveis.

Adicionalmente, é possível modificar dinamicamente os mapas SVG em quase todos os aspectos imagináveis, como por exemplo, adicionar ou eliminar objectos, assim como camadas de mapas. Isto permite uma fácil implementação de mapas altamente adaptáveis [Patalaviciute05]. Para poder dispor destas funcionalidades é necessário usar o SVG com o DOM (Document Object Model), que é referido na subsecção seguinte.

As linguagens de programação utilizadas neste projecto foram SVG, JavaScript, HTML, PHP, SQL e JAVA.

Ao longo deste trabalho foi formalizado um conjunto de ideias, em torno do SVG, em relação à utilização da cor, da transparência e animação de símbolos SVG. Existe a possibilidade de mudar a cor dos símbolos, colocar em primeiro plano um determinado símbolo depois de o seleccionar numa agregação de símbolos; utilizar a transparência, ou seja, colocar mais visíveis os símbolos mais importantes, e também aplicar *Fade in*/*Fade out* que funciona de modo análogo ao da transparência. Uma outra ideia que surgiu foi dar maior ênfase ao desenho dos símbolos de modo a que sejam claros e concisos. Neste aspecto, gostaríamos de associar uma animação aos símbolos de acordo com a importância do símbolo. Um exemplo para colocar a funcionar esta ideia seria criar uma funcionalidade de animação nos símbolos, de modo a que apareçam os que têm um maior valor da função de grau de interesse com um nível de movimento médio, com uma função de grau de interesse intermédio, com um nível de movimento pequeno e os restantes sem movimento, ou como alternativa utilizar as cores para fazer essa distinção.

Podem, também, ser implementadas outras técnicas como o efeito 'Dock-Like', 'Layers', 'Scaled', e outras úteis para, por exemplo, dividir o SVG mapa em mosaicos regulares de forma a tornar o mapa menos pesado para a aplicação, como se pode ver na figura 3.9.

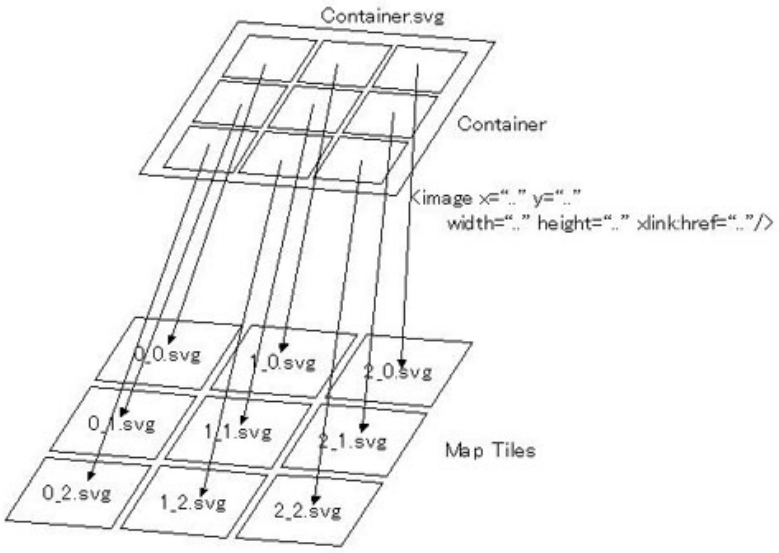


Figura 3.9 – Um ficheiro SVG dividido em 9 mais pequenos [Clip]

Outras ideias foram também pensadas. Por exemplo, quando existe uma grande concentração de pontos de interesse no mesmo sítio, ao clicar num deles, separamo-los, ou ao sobrepor o rato num símbolo aumentar o seu tamanho para o dobro para se visualizar melhor. Eventualmente, numa fase final é possível criar ícones, em forma de quadrado, associados aos símbolos, como existem no canto superior direito das janelas do Windows, de modo a poder fechá-los ou minimizá-los.

Os símbolos SVG que representam os pontos de interesse são desenhados sobre o mapa após cálculo dos valores respectivos da função de grau de interesse. Esta função de grau de interesse será explicada mais em pormenor numa das secções seguintes.

3.3.1 DOM

O DOM (Document Object Model) é uma especificação da W3C, independente de plataforma e linguagem, que permite alterar e editar dinamicamente a estrutura, conteúdo e estilo de um documento electrónico [url-W3]. Fornece um identificador único e acesso (por exemplo document.getElementById('idname')) a todos os objectos da página web, incluíndo tabelas, imagens, objectos gráficos etc. Representa a base para um trabalho eficiente com elementos de uma página da web, como é o caso de conteúdos cartográficos [Neumann03].

Existe uma ordem hierárquica segundo a qual é possível identificar e modificar cada objecto de um ficheiro SVG. A figura 3.10 ilustra a hierarquia.

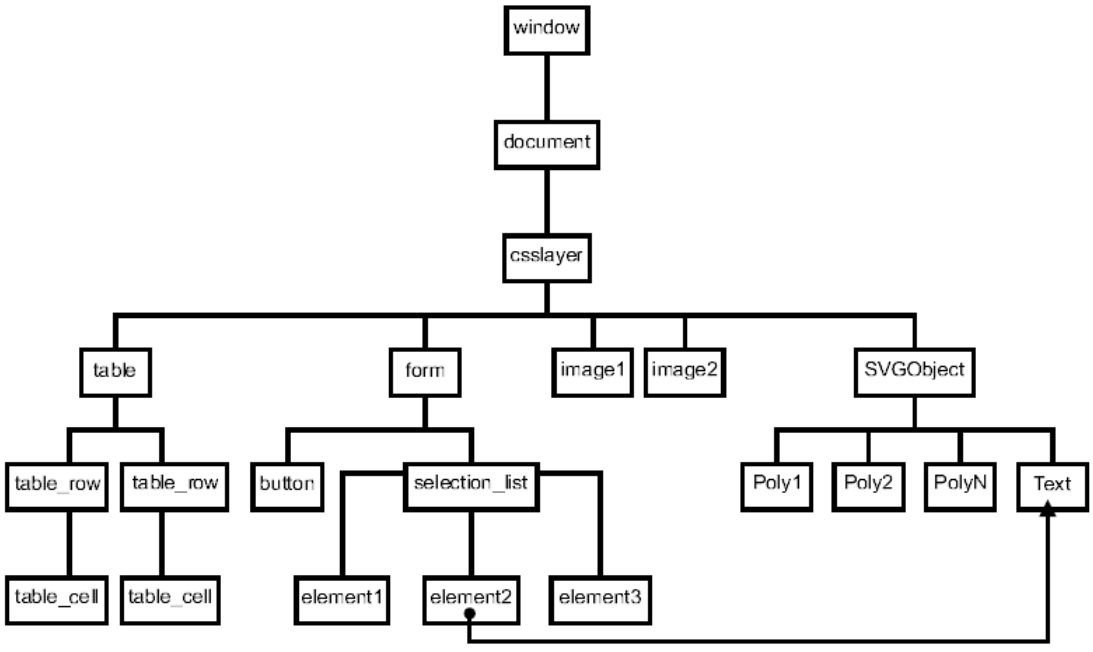


Figura 3.10 – Exemplo de um simples DOM numa página web [Neumann01]

Existe, portanto, uma hierarquia que permite estabelecer comunicação com cada um dos elementos. Neste caso, o elemento "text" dos gráficos SVG pode aceder o valor the "element2" na "selection-list" do elemento "form" e desenhar o texto no ficheiro SVG.

A integração do SVG com o DOM significa que os elementos do SVG podem ser controlados e modificados pelo Javascript. Isto é, conseguimos realizar animações no SVG invocando funções do Javascript.

3.3.2 Arquitectura do Sistema

A arquitectura do sistema utilizada neste projecto segue o modelo cliente / servidor, e é composta por uma grande variedade de componentes, entre os quais foi utilizado o servidor Apache com suporte para PHP.

Para este caso, foi instalado um servidor independente de plataforma, um software livre denominado XAMPP. Para obtenção do mapa e sua manipulação foi utilizado o GeoServer, que tem como objectivo produzir o mapa em SVG. O acesso à base de dados, que armazena toda a informação relativa aos pontos de interesse foi implementado usando o Microsoft SQL Server 2005.

A interface gráfica foi desenvolvida em SVG e HTML, tendo sido também utilizadas outras linguagens de programação como o PHP, o Javascript, o Java e o SQL. Para visualização do trabalho foi utilizado o *browser* Google Chrome e o *browser* Safari com o *plug-in* do SVG.

Numa etapa prévia do trabalho, quando obtivemos o mapa de Lisboa no formato shapefile tivemos que utilizar o software Geoserver de modo a convertê-lo para SVG, através do *web map service*.

Figura 3.11 – Etapa prévia para obtenção do mapa

Após esta primeira etapa começámos a desenvolver o nosso protótipo em que combinámos num ficheiro as linguagens SVG e HTML. A sua execução permite lançar a aplicação VisWide. Um segundo ficheiro integrando as linguagens SVG, PHP, SQL e Javascript foi desenvolvido e é executado a partir do momento em que o utilizador pressiona o botão *submit* na janela inicial da aplicação (figura 3.2). Este ficheiro desencadeia algumas tarefas como executar *queries* à base de dados e calcular a função de grau de interesse invocando classes programadas na linguagem JAVA. Finalmente é visualizado no ecrã o resultado final da procura do utilizador.

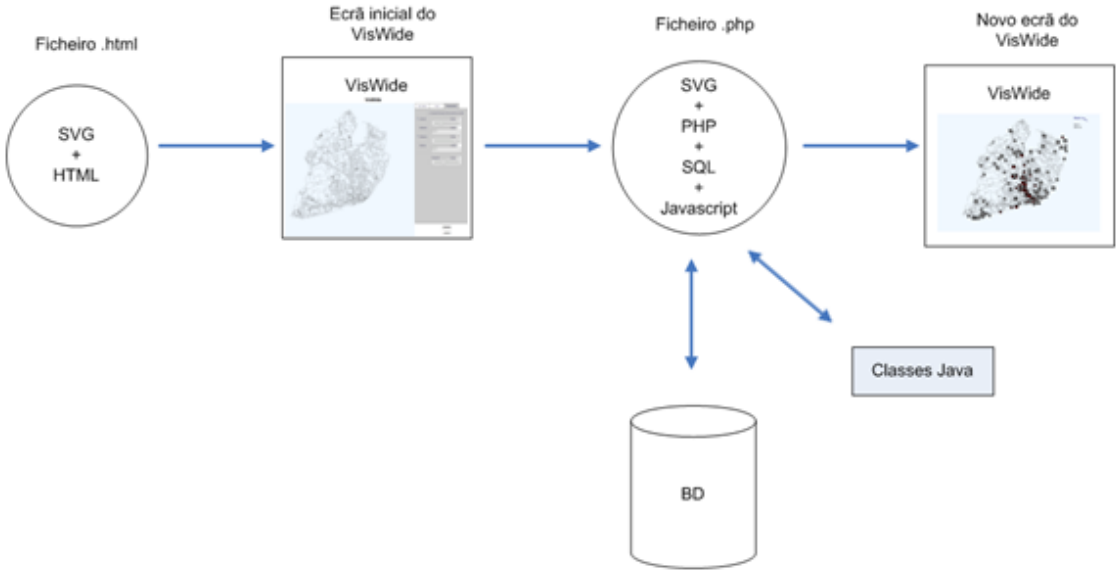


Figura 3.12 – Arquitectura utilizada no protótipo desenvolvido

3.3.3 Linguagens Server-side

Uma das potencialidades de que tirámos partido neste projecto é o facto do SVG permitir utilizar *scripts server-side*, o que nos possibilita o armazenamento de dados em ficheiros e a procura de informação em bases de dados. No entanto, existe como desvantagem a transferência de informação entre o lado do cliente (JavaScript) e o lado do servidor (PHP). Como solução para este problema, existe a possibilidade de utilizar a linguagem AJAX, para passar os dados [Shoemaker06] e/ou utilizar os *forms* na linguagem HTML.

Contudo, as tentivas para integrar o AJAX e o SVG não foram bem sucedidas. Portanto optou-se por usar forms HTML.

No caso do PHP, linguagem server-side, é utilizada a função header, que permite visualizar a informação no browser, e o content-type, que contém o tipo de informação MIME, que como já foi referido anteriormente, é image/svg+xml. Normalmente, o PHP utiliza text/html, mas isto impediria o browser de exibir os dados correctamente.

3.3.4 Símbolos

O papel de um símbolo é contribuir para comunicar informação de uma forma mais rápida e fácil. O símbolo é uma representação de um objecto, ideia ou acção [url-Símbolos].

Como foi dito antes, os símbolos além de facilmente vísiveis e relevantes devem ser de fácil compreensão, auto-explicativos e proporcionar uma melhor leitura do mapa, ou seja, o utilizador ao efectuar uma pesquisa, deve posteriormente compreender correctamente e facilmente o significado da sua simbologia.

Figura 3.13 – Exemplo de Símbolos (Restaurante, Museu, Estação de Serviço, Hotel) [Carto]

Um aspecto que tivemos em conta, foi a utilização de cores com um grande contraste para que o utilizador interprete mais facilmente a informação que lhe é dada. Por este motivo os símbolos possuem um tom preto como cor de fundo que não é utilizado nas cores do mapa e o branco para desenhar os símbolos. Para diferenciar a importância dos símbolos pode-se então alterar o branco para um tom vermelhado com um grau distinto de saturação. Como alternativa podemos também diferenciar os símbolos aumentando o tamanho de um determinado símbolo, para que este se destaque melhor dos restantes.

Procuramos, também, utilizar símbolos que fazem parte do dia-a-dia da pessoa de forma a que seja informação facilmente identificável.

3.3.5 Coordenadas

Como este projecto é feito para computadores *desktop*, e não para dispositivos móveis, não foi utilizado nenhum aparelho GPS de modo que o sistema de coordenadas utilizado não foi o geográfico (latitude e longitude). Ao utilizarmos um mapa em SVG é necessário converter as coordenadas para o Sistema Universal Transverso de Mercartor (UTM), já que o SVG utiliza um sistema de coordenadas cartesiano.

É importante referir que a fonte dos dados, dos pontos de interesse, com que trabalhamos foi obtida através do Google e como as coordenadas se encontram no sistema de coordenadas Google, foi necessário proceder à conversão das coordenadas dos pontos de interesse. Desta maneira foram recalculadas tanto a latitude, como a longitude com base nestas fórnulas :

Latitude_nova = latitude / 1000000
Longitude_nova = - (360000000 - (longitude/ 1000000))

O Sistema Universal Transverso de Mercartor resulta na composição de 60 fusos distintos que representam a superfície da Terra. Cada fuso tem a amplitude de 6º de longitude, sendo que a zona 1 localiza-se entre a longitude 180 a 174º O. Em relação às latitudes, o sistema é limitado pelos paralelos 84º N e 80º S, onde as deformações ainda não são significativas. Para latitudes maiores, já é utilizada a projecção Estereográfica Polar Universal.

A capital de Portugal encontra-se na zona 29 S (Hemisfério Norte) deste sistema, o que significa que terá uma latitude positiva e uma longitude negativa.

Uma posição na Terra é referenciada no sistema UTM pela sua zona UTM e o par de coordenadas (x,y). O X é a distância projectada da posição do centro meridiano, enquanto que o Y é a distância projectada a partir do equador. O ponto de origem de cada zona UTM é a intersecção do equador e a zona central meridional. De maneira a evitar lidar com números negativos, o meridional central de cada zona recebe um falso X valor de 500,000 metros, de maneira que tudo a oeste do meridional central terá um X menor que 500,000 metros. Por exemplo, X varia entre 167,000 metros até 833,000 metros no equador. No hemisfério norte, as posições Y são medidas a partir do equador, que tem um valor inicial de 0 metros e um máximo de aproximadamente 9,328,000 metros no paralelo 84. No hemisfério Sul, o Y decresce à medida que vamos para baixo do equador, em que é dado um falso Y de 10,000,000 metros de modo a que nenhum ponto dentro da zona tenho um valor negativo.

Para dar um exemplo, a Faculdade de Ciências encontra-se na posição Latitude 38.756, Longitude -9.157. Encontra-se portanto, na zona 29 e a posição X é 486358 metros, e a Y é 4289711 metros. Existem dois pontos na Terra com estas coordenadas.

Um no hemisfério norte e outro no hemisfério sul. Para não existir ambiguidade, deve usar-se uma de duas convenções:

1. Adicionar o hemisfério "N" ou "S" ao número da zona, ou seja "29 S 486358 4289711". Assim temos informação necessária para definir cada posição unicamente.
2. Adicionar a zona da grelha, por exemplo "29 T 486358 4289711". Neste caso, pode haver ambiguidades visto que a zona S se encontra no hemisfério norte.

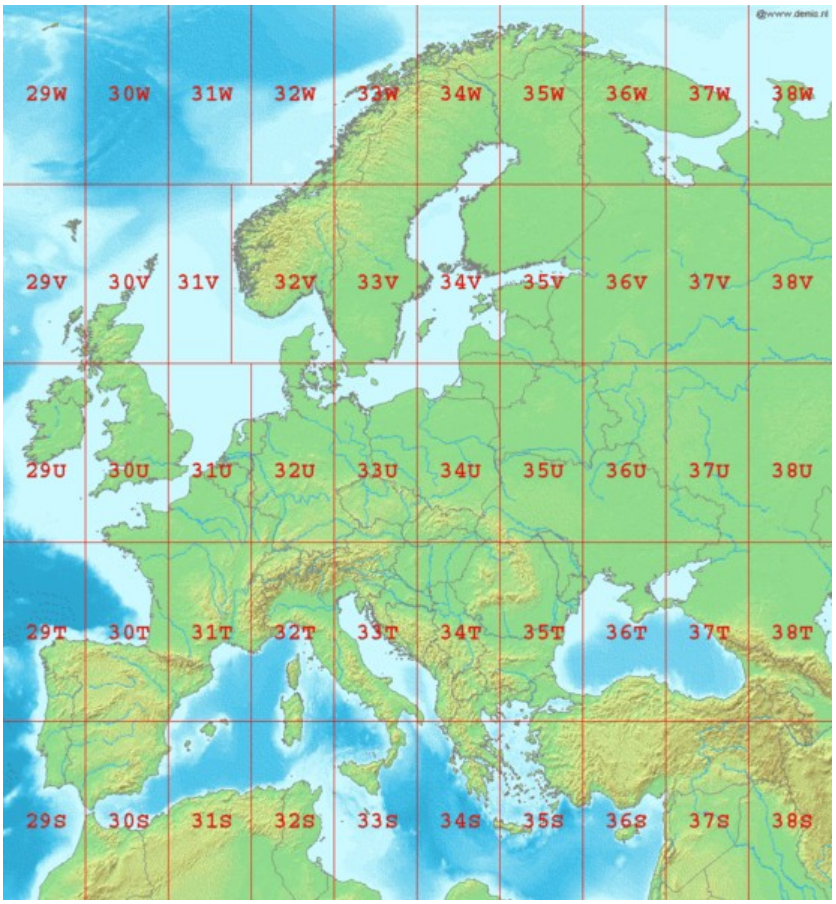


Figura 3.14 – Zonas UTM na Europa [url-Wikipedia]

Para não haver dúvidas deve escrever-se Norte ou Sul sempre que queremos indicar o hemisfério. Para este trabalho, foi utilizada a primeira convenção. Numa classe Java do trabalho, são dados dois argumentos, uma latitude e longitude, para então ser retornado as coordenadas X,Y.

Através destes últimos dados podemos finalmente adicionar os pontos de interesse armazenados na nossa base de dados

3.3.6 Interface

A interface do VisWide foi desenvolvida em HTML devido ao problema de compatibilidades que as linguagens Server-side trouxeram ao projecto. No início foi feita uma tentativa para fazer tudo em SVG, mas devido aos problemas que havia relacionados com o AJAX, passou-se a utilizar HTML. A solução para este problema passou por criar um *form* e assim enviar toda a informação do html para o ficheiro SVG, como, por exemplo, o valor de importância de cada atributo e valor verdadeiro/falso de cada atributo.

Um menu tem um problema: - Ocupa muito espaço [Alan Dix03]. Por esta razão foram adicionadas abas à interface do VisWide, relativas às seguintes categorias: Hotéis, Estações de Serviço e Monumentos, uma aba para cada categoria. Cada uma dessas abas contêm vários atributos, que podem ser seleccionados através de *checkboxes,* uns *sliders* para poder eleger o grau de importância de cada um dos atributos e um último *slider* para indicar o limiar de importância. A informação está assim organizada em várias categorias, e cada categoria pode ter vários atributos.

Estas opções irão então ser enviadas para o ficheiro de imagem SVG através do *form* como foi dito anteriormente, e posteriormente enviado para um ficheiro JAVA para calcular a função de grau de interesse. Finalmente, com o valor desta função calculado, serão mostrados os pontos de interesse no mapa. Foi, também, criada uma ligação "Back Html" de modo ao utilizador, após visualizar o mapa, retroceder à interface para então criar uma nova pesquisa. Para embeber o SVG no HTML foi utilizado um método denominado iframe. Para além deste, existem outros dois: o embed e o object. A razão pela qual foi utilizado o iframe é porque tem como vantagem funcionar na maioria dos *browsers*.

As figuras 3.15 e 3.16 contêm imagens da interface.

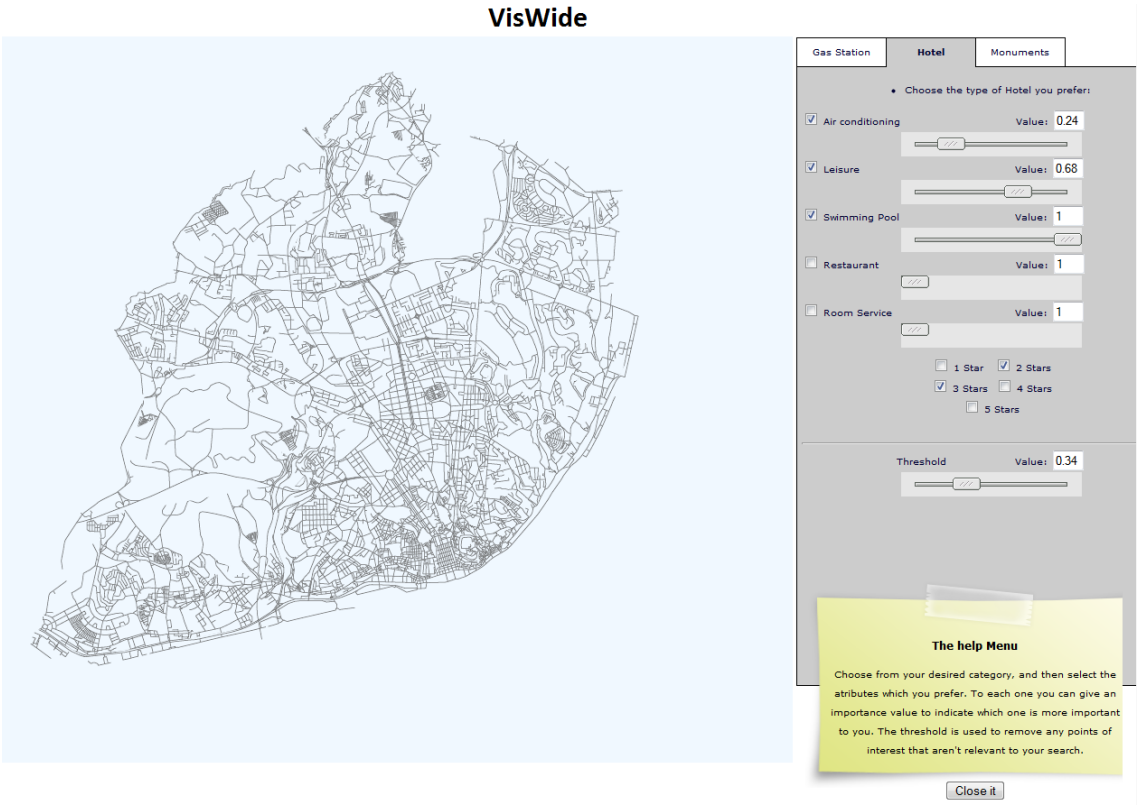


Figura 3.15 – Exemplo de interface para a categoria Hotel com o menu ajuda

Na figura 3.15 mostra-se um exemplo de uma pesquisa de hotéis. Neste caso, procuramos pontos de interesse do tipo hotéis com os atributos Ar Condicionado, Lazer e Piscina e com um grau de importância 0.24, 0.68 e 1, respectivamente. Em relação ao número de estrelas foi indicado, na pesquisa, duas e três estrelas. Finalmente no *threshold* colocamos um nível mínimo de 0.34. Como é possível verificar foi adicionado um botão de help, cuja selecção desencadeia o aparecimento de um quadro de ajuda (canto inferior direito da figura 3.15). Foi colocada esta funcionalidade para fornecer ajuda ao utilizador em caso de dúvida de funcionamento. Deste modo, lendo uma breve descrição, o utilizador pode-se orientar e facilmente seleccionar o que pretende.

É de referir que toda a interface foi feita em inglês. A concepção da interface foi pensada de maneira a ser simples, fácil de usar, adaptada ao conhecimento e experiência do utilizador. Com as abas, consideramos que se torna fácil trocar de menu de maneira a fazer a seleccção de todos os atributos antes de clicar no *Submit*.

Figura 3.16 – Exemplo Interface para Monumentos

A figura 3.16 ilustra uma utilização em que se pretende um ponto de interesse da categoria Monumento com atributos do tipo Cultural e Religioso, com os valores de interesse 0.27 e 0.85 respectivamente e finalmente com um *threshold* de 0.37. Após efectuar a seleccção, basta carregar no botão *Submit*, obtendo o resultado ilustrado na figura 3.17.

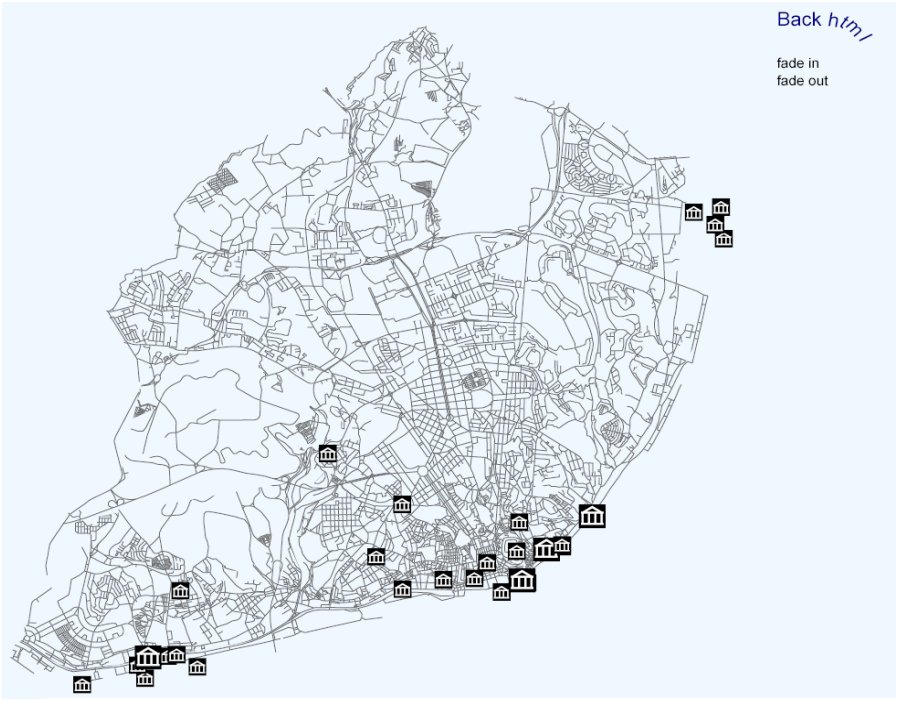


Figura 3.17 – Resultado da pesquisa da aplicação VisWide

3.3.7 Estratégias de Representação de Pontos de Interesse

Para tratar a sobreposição excessiva de pontos de interesse numa mesma área do mapa adoptámos as seguintes estratégias [Paiva09] :

- tornar totalmente transparentes todos os símbolos visíveis e efectuar a operação inversa desta;
- minimizar/maximizar símbolos individuais (figura 3.18);
- minimizar/maximizar símbolos em conjunto (figura 3.19);
- separar/reunir um conjunto de símbolos (figura 3.20);
- aplicar diferentes tonalidades da mesma cor (figura 3.21);
- agregação, ou seja colocar um símbolo em cima de outro (figura 3.22);
- aplicar um efeito de aumento progressivo de zoom a símbolos que se encontram bastante próximos (figura 3.23);
- efectuar um zoom a um determinado símbolo (figura 3.24);
- usar vários níveis de transparência, explorando efeitos do tipo *fade-in*/*fade-out* (figura 3.25);
- adaptar o tamanho dos símbolos representados em conformidade com o valor da função grau de interesse (figura 3.26).

Para implementar estas estratégias em SVG, recorremos à linguagem ECMAScript, uma linguagem de programação baseada em scripts e que permite criar animações e interfaces interactivas dentro do SVG.

Ao combinarmos SVG com ECMAScript temos a base para a criação do JavaScript, tendo assim a oportunidade de gerar uma grande variedade de animações e efeitos para explorar, que vamos utilizando à medida que o utilizador visualiza uma zona do mapa em que ficam ocultos determinados pontos de interesse por sobreposição de outros.

Para exemplificar a implementação de uma estratégia neste trabalho foi escolhida a estratégia minimização. Apresentamos seguidamente um excerto de código em Javascript e SVG.

```
//Javascript
  function minimizar(evt) {
  SVGDoc = evt.getTarget().getOwnerDocument(); var oi = evt.target;
  if (isEven(paridade)) {
     var node = svgDocument.getElementById('minin');
     pic=SVGDoc.createElementNS(svgNS,"image");
     pic.setAttribute ("x", -92500); pic.setAttribute ("y", -99200);
     pic.setAttribute ("width", 999); pic.setAttribute ("height", 999);
     pic.setAttributeNS (null, "id", "minincruz");
     pic.addEventListener("click",minimizar,false);
     pic.setAttributeNS(xlinkNS,"href","desenho3.svg");
     node.parentNode.replaceChild(pic,node);
     var qElement = document.getElementById('desaparece');
     qElement.parentNode.removeChild(qElement);      paridade++;
     }  else {
  var crum = document.getElementById('minincruz');
  pic=SVGDoc.createElementNS(svgNS,"image");
  pic.setAttribute ("x", -92500);  pic.setAttribute ("y", -99200);
  pic.setAttribute ("width", 999); pic.setAttribute ("height", 999);
  pic.setAttributeNS (null, "id", "minin");
  pic.addEventListener("click",minimizar,false);
  pic.setAttributeNS(xlinkNS,"href","desenho1.svg");
  crum.parentNode.appendChild(pic);
  picx=SVGDoc.createElementNS(svgNS,"image");
  picx.setAttribute ("x", -92920); picx.setAttribute ("y", -99065);
  picx.setAttribute ("width", 999); picx.setAttribute ("height", 999);
  picx.setAttributeNS (null, "id", "desaparece");
  picx.setAttributeNS(xlinkNS,"href","aiga_hotel_information.svg");
  crum.parentNode.replaceChild(picx,crum);   paridade++;
```

```
    }
}
//SVG
<g id="imagemx">
  <image id="desaparece" x="-92920" y="-99065"  width="999" height="999" xlink:href="aiga_hotel_information2.svg" />
<image id="minin" x="-92500" y="-99200" width="150" height="150" xlink:href="desenho1.svg" onclick="minimizar(evt)"/>
</g>
```

Na figura 3.18 podemos ver, no mapa, o resultado final da execução do código transcrito.

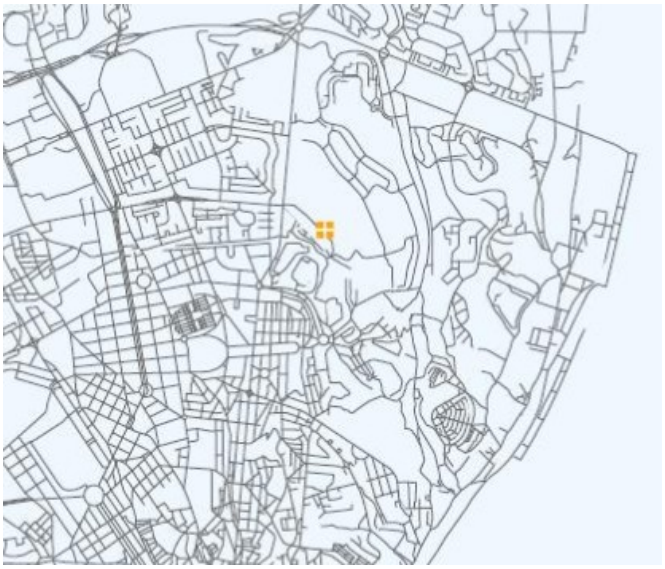
(a)

(b)

Figura 3.18 – Minimização

Na figura 3.19 é feita uma demonstração de como esta mesma técnica pode ser utilizada para um conjunto de símbolos.

(a)



(b)

Figura 3.19 – Minimização em conjunto

Em relação à técnica de separação, mostra-se um exemplo com três símbolos (figura 3.20). Esta técnica tem como grande vantagem a visualização de todos os pontos de interesse com uma grande clareza quando se procede à sua separação, e ao mesmo

tempo o utilizador continua com o conhecimento de qual a correcta posição do símbolo. Isto é feito através de uma seta que indica onde se encontra o ponto de interesse. Para efectuar a separação dos pontos de interesse, basta clicar num deles e para juntá-los, faz-se o mesmo procedimento.

Na figura 3.20, pode-se ver como funciona esta técnica.

(a)

(b)

Figura 3.20 – Separação

Nestas duas estratégias, foram utilizados um EventTarget, nomeadamente o addEventListener para adicionar eventos, fornecendo como parâmentro um objecto e, assim, criar um novo símbolo.

Foram exploradas a dimensão e a cor, em vários níveis de saturação, dos símbolos gráficos e a geração de animações usando variações dinâmicas de níveis de transparência.

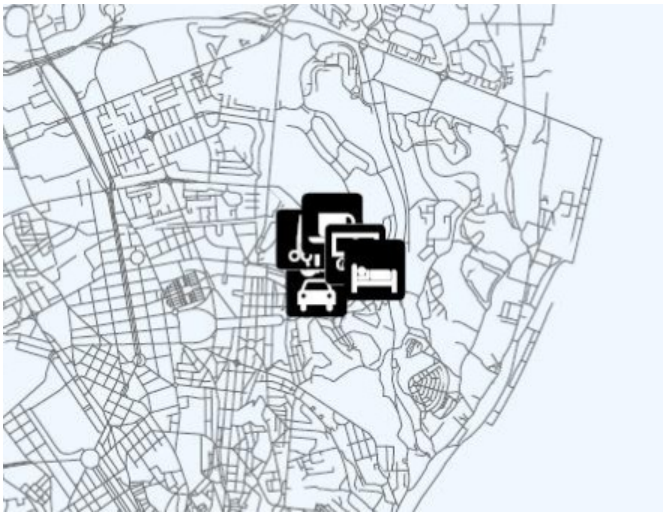
Optámos por variar apenas três valores nestas características, isto é, usamos o mesmo símbolo em três tamanhos diferentes, e em três tonalidades da mesma cor (Figura 3.21).

Figura 3.21 – Tonalidades da mesma cor

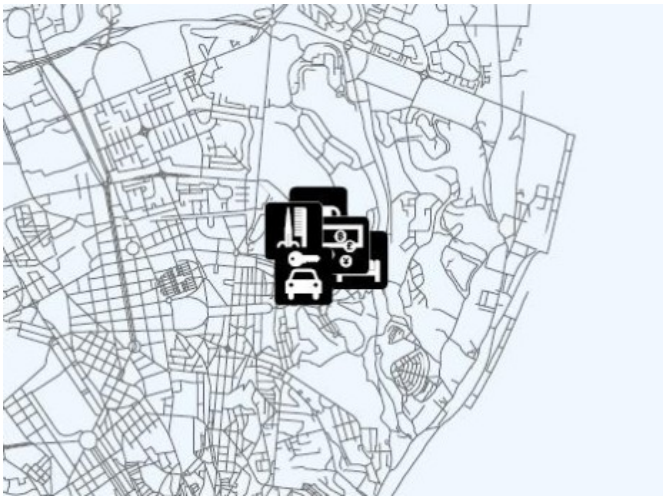
Destacamos, ainda, mais duas estratégias que nos permitem lidar com situações em que ocorrem diversos símbolos muito próximos geograficamente. A agregação (Figura 3.22) permite ao utilizador visualizar melhor os símbolos do conjunto, escolhendo um ponto de interesse de cada vez com o rato, de maneira a surgirem no topo da agregação. Sempre que se carrega num símbolo, colocamo-lo acima de todos os outros, e se o carregarmos de novo, vai voltar para baixo de todos os outros.

Esta técnica tem um resultado especialmente bom, visto que é bastante fácil de utilizar e de concretizar, no entanto quando existem muitos pontos de interesse na mesma zona, há a necessidade de clicar várias vezes até encontrarmos o ponto de interesse desejado.

Na figura 3.22 exemplifica-se o efeito de agregação.



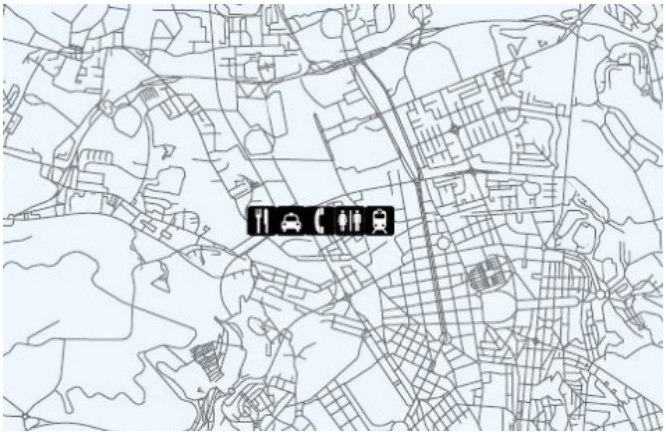
(a)



(b)

Figura 3.22 – Agregação

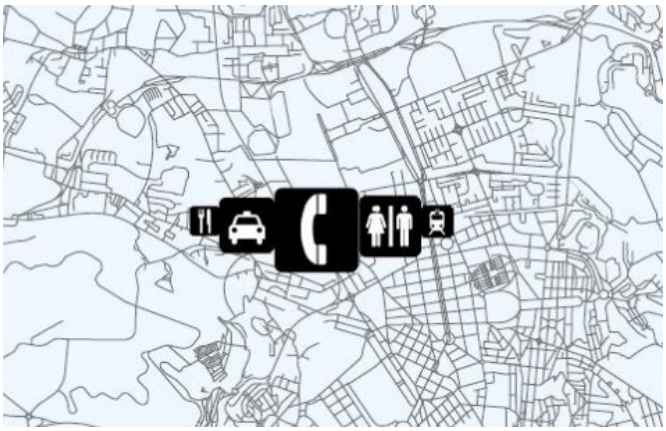
O efeito *Dock-Like* foi também implementado (Figura 3.23), e permite efectuar um zoom que realça o ponto de interesse sobre o qual o rato se encontra.



(a)

No exemplo mostrado na figura 3.23 (a), colocámos o rato em cima do símbolo do telefone que se encontra na posição do meio. Como é possível verificar, aplicou-se um zoom grande no símbolo do telefone, um zoom intermédio nos símbolos à sua direita e à sua esquerda e nos mais afastados um zoom pequeno. Foi aplicado um zoom simétrico para todos os símbolos, mas esta técnica tem uma pequena desvantagem que é o facto dos símbolos serem todos colocados lado a lado, o que implica uma grande ocupação dos pontos de interesse no mapa.

Na imagem 3.23 (b) é possível ver o efeito final do zoom.



(b)

Figura 3.23 – Efeito *Dock-Like*

Como outras técnicas suplementares, temos a possibilidade de aplicar um determinado zoom assim que se coloca o rato em cima do ponto de interesse, com o intuito de visualizá-lo mais claramente. Para voltar ao normal, basta retirar o rato de cima do ponto de interesse.

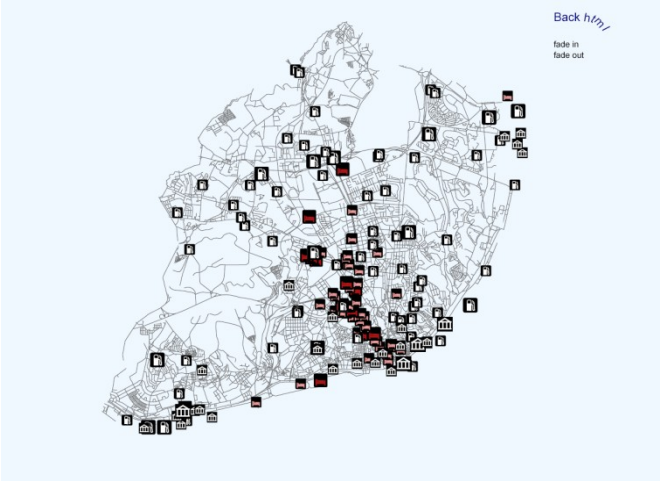
(a)

(b)

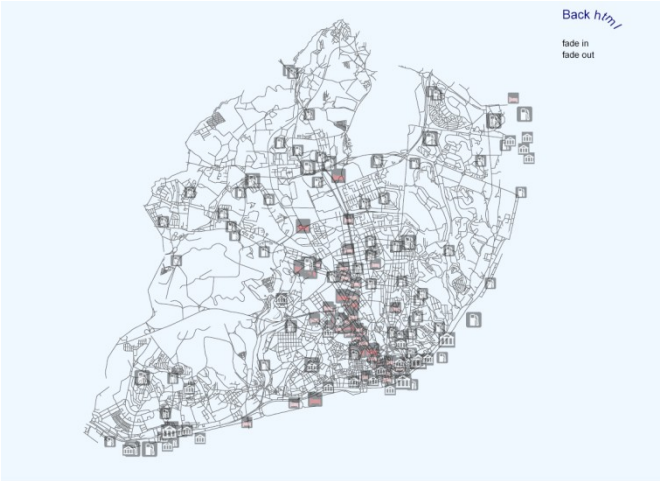
Figura 3.24 – Efeito *Zoom*

Os efeitos de fade-in e fade-out podem ser utilizados como orientação no mapa, pois, por vezes, pode-se tornar dificil a visualização do que se encontra por detrás do

símbolo. Esta última técnica utiliza-se basicamente para manter um nível alto de transparência, em vez de fazer desaparecer completamente o símbolo.



(a)



(b)

Figura 3.25 – Efeito *Fade-in*/*Fade-out*

Como última estratégia decidimos aplicar uma diferenciação dos símbolos em conformidade com o valor da função grau de interesse. Se o ponto de interesse é suficientemente relevante para aparecer no mapa, verificamos se o seu interesse é superior à soma do *threshold* mais um, a dividir por dois. Se for esse o caso então o símbolo aparece com um tamanho grande, caso contrário aparece com um símbolo pequeno. A figura 3.26 ilustra um exemplo em que existem bombas de gasolina com maior grau de interesse que outras.

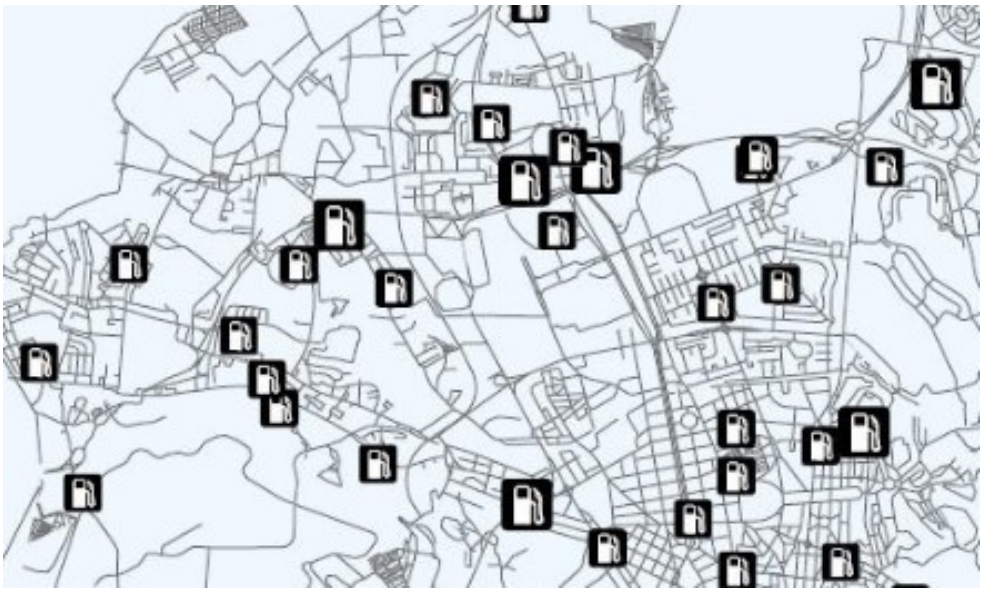


Figura 3.26 – Adaptar o tamanho dos símbolos de acordo com o seu interesse

Como conclusão, pensamos ainda, que não devemos usar mais do que uma característica de cada vez, na aplicação. Contudo, apenas os testes de avaliação da aplicação nos poderão levar a concluir sobre estes aspectos.

3.3.8 Função de Grau de Interesse

A função de grau de interesse é essencial no trabalho, de modo a fornecer ao utilizador um número de pontos de interesse que são mais indicados à pesquisa que realizou e com um número limitado dos mesmos. Desta maneira a informação no mapa torna-se mais inteligível e concisa.

Para a função de grau de interesse foi utilizada a seguinte fórmula:

$$DOI\ (\rho_j) = \frac{\sum_{i=i}^{k} UI(a_i, \rho_{ji})}{k} \in [0,1]$$

em que K corresponde aos diferentes atributos seleccionados pelo utilizador, no exemplo de Hotel (Ar Condicionado, Lazer, Piscina, Restaurante e Serviço de Quarto). O conjunto A

representa os K valores de atributos seleccionados pelo utilizador e o conjunto P os n pontos de interesse existentes.

A função UI($a_{i,} p_{ji}$) é dada pela seguinte fórmula:

$$UI(a_{i,} p_{ji}) = 1 - Dist(a_{i,} p_{ji}) \times w_i, w_i \in [0,1]$$

Neste caso w_i será a importância escolhida pelo utilizador para o atributo e a Dist($a_{i,} p_{ji}$) será 0 no caso de $a_i = p_{ji}$ e 1 no caso de $a_i \neq p_{ji,}$ ou seja se o utilizador tiver seleccionado um Hotel com Ar Condicionado e o ponto de interesse tiver Ar Condicionado na sua base de dados então a Dist($a_{i,} p_{ji}$) será 0, senão será 1.

- Distância entre atributos nominais/booleanos

$$Dist(a_{i,} p_{ji}) = \begin{cases} 0, \text{ se } a_i = p_{ji} \\ 1, \text{ se } a_i \neq p_{ji} \end{cases}$$

O valor da função de interesse foi realizado para ter um intervalo de [0,1] onde o 0 corresponde a um ponto de interesse que tem pouca ou nenhuma relevância para o utilizador e o 1 indica um, que tenha grande importância.

Para este trabalho, foi utilizado um limite mínimo (*threshold*) para o valor da função, sendo posteriormente mostrados os símbolos que realmente são importantes (que excedam o limite estipulado) para o utilizador, de modo a diminuir o número de elementos a apresentar, permitindo assim a omissão de informação não relevante para o utilizador.

Ao diminuir o valor deste limiar, serão apresentados mais pontos de interesse.

Tal como na função de grau de interesse, o *threshold* pertence ao intervalo [0,1].

Capítulo 4

Conclusão e Perspectivas Futuras

4.1 Conclusão

Este trabalho centrou-se na visualização de informação geo-referenciada por categorias em computadores de secretária e portáteis. Com o objectivo de criar imagens inteligíveis através da informação requisitada pelo utilizador, foram criados mecanismos de filtragem e estratégias para representar pontos de interesse.

Considero que foi um verdadeiro desafio desenvolver um projecto numa linguagem totalmente nova para mim e colocar todas as seis linguagens de programação a funcionarem, simultaneamente, sem problemas de interactividade. Este último factor deve-se, em grande parte, à facilidade do SVG interagir com outras linguagens. Contudo, por ser ainda recente deparámo-nos com pequenos obstáculos de compatibilidade que foram superados ou por troca de ferramentas de trabalho (foram utilizadas ferramentas como o inkscape para poder editar pequenos ficheiros), ou de linguagens de programação (AJAX foi colocado de parte), ou simplesmente adoptando diferentes técnicas programação, pois primeiramente foi pensado programar toda a interface em codígo SVG, mas com o evoluir do projecto foram surgindo obstáculos delicados e de díficil resolução, que levaram a alterar o rumo do trabalho. Toda a aplicação funciona perfeitamente para as diferentes categorias hotéis, estações de serviços e monumentos.

Como o mapa utilizado no projecto é da capital de Portugal apenas aparecem os símbolos correspondentes à area de Lisboa, mas se alterarmos a localização do mapa, a aplicação encontra-se suficientemente robusta para obtermos os pontos de interesse de outras regiões de Portugal.

Um outro benefício de que tirámos proveito, por usarmos mapas de formato vectorial, como já foi anteriormente referido, foi o aspecto de trabalharmos com um

ficheiro de tamanho mais pequeno o que permite poupar no tempo de carregamento, algo que se dá cada vez mais importância na internet nos dias correntes.

Em relação às diferentes estratégias formuladas no trabalho, como não foi possível realizar testes de avaliação não é possivel chegar a nenhuma conclusão, portanto é impossível determinar qual delas é a mais eficaz para representar a informação relevante, de uma forma intuitiva, e qual se adequa mais ao perfil do utilizador.

Existem como opções realizar uma combinação de duas ou mais estratégias ou simplesmente utilizar apenas uma para mostrar os resultados dados pela pesquisa do utilizador.

4.2 Perspectivas Futuras

Ao longo do desenvolvimento do trabalho foram surgindo algumas ideias para trabalho futuro, que passo a enumerar.

A interface da aplicação de acordo com uma função de razão de aspecto (largura e altura) do mapa. Como por exemplo:

- Razão Aspecto < 1

(a) (b)

Figura 4.1 – Mapa (a) e Interface (b) do VisWide com razão de aspecto < 1

- Razão Aspecto > 1

(a) (b)

Figura 4.2 – Mapa (a) e Interface (b) do VisWide com razão de aspecto > 1

Ainda em relação à interface seria curioso criar uma interface e/ou representação dos pontos de interesse de acordo com as características dos utilizadores, sua idade, sexo, áreas de interesses mas só com testes podemos tirar alguma conclusão.

Seria interessante como trabalho futuro aplicar este projecto a outras áreas de interesse como a História, a Biologia, etc.

Todos os símbolos foram implementados em 2D, mas numa bordagem futura existe a curiosidade de testar a utilização de símbolos 3D, contudo a linguagem SVG é uma linguagem 2D de maneira que uma solução passaria por utilizar efeitos 2.5. Esta dimensão dois e meio, também conhecida como *pseudo-3D* é um termo informal que define imagens que se consideram estar entre 2D e 3D.

Deste modo, é possível modificar gráficos 2D para simular uma terceira dimensão e, assim, criar símbolos mais atractivos e ao mesmo tempo símbolos que tornem mais fácil a percepção e leitura do mapa para o utilizador.

Figura 4.3 – Ideia para cria um efeito 2.5D através de um cubo

e associar animação aos símbolos de acordo com a im pela função de grau de interesse. Esta animação seria ef tática que corresponderia a um interesse baixo, uma fig édio e, finalmente, um terceiro nível com a animação rápida correspondendo a um nível elevado de interesse. Este tipo de observação contribuiria para uma melhor e mais rápida percepção do mapa, visto que se um símbolo de grande importância se encontra em movimento, o utilizador pode mais fácilmente e rapidamente encontrá-lo.

Para concluir falta referir o aspecto mais relevante: deve ser efectuado um plano de avaliação, composto por vários testes, de modo a interpretarmos as necessidades do utilizador e verificarmos se as técnicas e opções escolhidas na realização do projecto foram as mais adequadas.

Bibliografia

[Argerich02] Argerich, Luis., Egervari, Ken., Anton, Matt., Lea, Chris., Killian, Charlie., Hubbard, Chris., & Fuller, James. (2002). *Professional Php4Xml*. Wrox Press

[Bauer02] Bauer, Gerald. (2002). *Scalable vector graphics (SVG) : Creating high-end 2D Graphics using XML*. Java User Group (JUG) Austria Talk.

[Dix03] Dix, Alan., Finlay, Janet., Abowd, D. Gregory., & Beale, Russel. (2003). *Human Computer Interaction*. 3rd Edition, Prentice Hall.

[Edwardes05] A. Edwardes et al. Portrayal and Generalisation of Point Maps for Mobile Information Services. *Map-based Mobile Services Theories, Methods and Implementations, pp. 11-30, 2005*.

[Freitas05] Freitas, Sérgio., Carmo, Maria Beatriz,. Afonso, Ana, Paula., & Cláudio, Ana, Paula. (2005). *Visualização Personalizada de Dados Geo-referenciados*, poster, Encontro Nacional de Visualização Científica.

[Fujisawa08] Fujisawa, Jun., & Grasso, Anthony. (2008). Graphic Design with SVG: *Achieving 3D Effects with SVG*. Proceedings of the 6th International Conference on Scalable Vector Graphics

[Geroimenko05] Geroimenko, Vladimir., & Chen, Chaomei. (2005). *Visualizing Information Using SVG and X3D.* Springer. pp. 18-62.

[Keim94] D. A. Keim., & H. P. Kriegel. (1994). *VisDB: Database Exploration Using Multidimensional Visualization*. Computer Graphics and Applications, IEEE, 14(5), pp. 40-49.

[Martins02] A. M. Martins. (2002). *Funções de Grau de Interesse – Aplicação a-Informação Extraída de Bases de Dados Relacionais*. Dissertação de Mestrado. Departamento de Informática da Faculdade de Ciências da Universidade de Lisboa.

[Neumann01] Neumann, Andreas., & Winter, Andréas. (2001). *Time For SVG – Towards High Quality Interactive Web-Maps*, Paper work, ICC Congress China.

[Neumann03] Neumann, Andreas., & Winter, Andréas, M. (2003). *Vector-based Web Cartography: Enabler SVG*: http://www.carto.net/papers/svg/index_e.shtml

[Paiva09] Paiva, Bruno., Cláudio, Ana, Paula., Carmo, Maria Beatriz., & Matos, Paulo Pombinho. (2009). *Estratégias para a Representação de Pontos de Interesse sobre Mapas*. 17 EPCG. Aceite para publicação.

[Patalaviciute05] Patalaviciute, Vilma., Düpmeier, Clemens., Freckmann, Peter., & Ruchter, Markus. (2005). *Using SVG-based Maps for Mobile Guide Systems. A Case Study for the Design, Adaptation, and Usage of SVG-based Maps for Mobile Nature Guide Applications*. Proceedings of the 4th Annual Conference on Scalable Vector Graphics

[Pavlicko, Peterson04] Pavlicko, P., & Peterson, M. P. (2004). *Topographic Maps with SVG*. Proceedings of the 3rd Annual Conference on Scalable Vector Graphics .

[Pombinho2008] Pombinho, Paulo. (2008) *Visualização de Informação Geo-Referenciada em Dispositivos Móveis*. Dissertação de Mestrado. Departamento de Informática da Faculdade de Ciências da Universidade de Lisboa.

[Presco2003] Presco, A. Paul. (2003). *SVG: The Sure Thing*. Proceedings of the 2nd Annual Conference on Scalable Vector Graphics.

[Reichenbacher02] T. Reichenbacher. (2002) *SVG for adaptive visualisations in mobile situations*, SVG Open Conference.

[Reichenbacher04] T. Reichenbacher. (2004). Mobile Cartography - *Adaptative Visualization of Geographic Information on Mobile Devices*. PhD Thesis. *Verlag Dr. Hut, München*

[Shoemaker06] Shoemaker, Craig., Wallace, B. McClure., Cate, Scott., & Glavich, Paul. (2006). *Ajax with ASP.NET*. Wrox Press.

[Vaz2008] Vaz, Ana Margarida Farto. (2008) *Pesquisas Interactivas para Informação Geo-referenciada em Dispositivos Móveis*. Dissertação de Mestrado. Departamento de Informática da Faculdade de Ciências da Universidade de Lisboa.

[Wenz02] Wenz, Christian., & Hauser, Tobias. (2002). *SVG server-side. Generating SVG on the fly*. Proceedings SVG Open 2002.

[Winter01] Winter, Andréas., & Neumann, Andreas. (2001). *SVG − Scalable Vector Graphics: A future cornerstone of the WWW−infrastructure*: http://www.carto.net/ papers/svg/articles/paper_xml_usergroup_neumann_winter_2001.pdf

[url-Adobe] Adobe. (data de último acesso: 28/09/2009). Obtido de http://www.adobe.com/svg/viewer/install

[url-Carto] Carto. (data de último acesso: 28/09/2009). Obtido de http://www.carto.net

[url-Clip] SVG MAP Lab. (2008). Obtido de http://blog.svg-map.com/2008/07/ format-and-proc.html

[url-Devarticles] DevArticles. (data de último acesso: 28/09/2009). Obtido de http://www.devarticles.com/c/a/XML/SVG-Imagings-Pot-of-Gold/

[url-GeoServer] GeoServer. (data de último acesso: 28/09/2009). Obtido de http://geoserver.org

[url-Símbolos] O significado dos símbolos. (data de último acesso: 28/09/2009). Obtido de http://www.acessibilidade.net/at/kit2004/Programas%20CD/ATs/cnotinfor/ Documentos/O%20Significado%20dos%20S%EDmbolos/O%20Significado%20dos%20S%EDmbolos.htm

[url-VisGeoRef] Visualization of Geo.referenced Information. (data de último acesso: 28/09/2009). Obtido de http://labmag.di.fc.ul.pt/VisGeoRef

[url-W3] W3C Recommendation. (data de último acesso: 28/09/2009). Obtido de http://www.w3.org/

[url-W3Path] W3C Recommendation. (data de último acesso: 28/09/2009). Obtido de http://www.w3.org/ TR /2008/WD-SVGMobile12-20080912/paths.html

[url-Wikipedia] Wikipedia (data de último acesso: 28/09/2009). Obtido de http://wikipedia.org

[url-Xul] Xul. (data de último acesso: 28/09/2009). Obtido de http://www.xul.fr/en-xml-svg.html