Jornadas SASIG5
15 – 17 de Novembro de 2012, Faro
Universidade do Algarve

MAPIGNITER: GEOCMS

Marco Filipe Vidal Afonso

mafonso333@gmail.com

Resumo

Sendo a ciência da informação geográfica uma área de estudo multidisciplinar, é comum encontrar associada a esta, muita informação não geográfica pertencente a outras áreas. Muitas vezes a gestão desta informação não geográfica, nas tecnologias de informação, é realizada por métodos e ferramentas completamente diferentes das que são usadas nos SIG. Os gestores de conteúdos são plataformas reconhecidas pela sua capacidade de colmatar a necessidade de gerir conteúdo diversificado e de publicar informação utilizando um Web browser. MapIgniter é um gestor de conteúdos em código aberto que permite também fazer a gestão de informação geográfica criando um novo conceito de GeoCMS. As vantagens de interfaces gráficas são conhecidas pela sua facilidade de uso, por aumentarem a produtividade dos seus utilizadores e por requererem menos conhecimentos técnicos para a realização de determinadas tarefas. MapIgniter integra numa única interface gráfica funcionalidades imprescindíveis para a publicação de informação geográfica na Internet ou Intranet.

Palavras-chave: WebSIG, CMS, MapServer, GUI, OpenLayers

1. INTRODUÇÃO: IDENTIFICAÇÃO DAS NECESSIDADES

Em áreas não multidisciplinares, o *software* especifico pode ser o mais adequado para resolver um determinado problema, mas em áreas multidisciplinares, como os SIG, pode ser necessário recorrer a outras funcionalidades que podem não existir em software de SIG, como por exemplo, a publicação de mapas interativos na *Internet*.
O presente projeto surgiu como resposta a colmatar as necessidades que se descrevem a seguir, como um todo.

1.1 PUBLICAR INFORMAÇÃO NA INTERNET/INTRANET

Com o aumento do uso de tecnologias de informação dentro das organizações, também nos sistemas de informação se procuram criar estruturas que promovam a cooperação interna e externa à organização.
O modelo computacional cliente-servidor é um modelo que tem oferecido uma estrutura adequada à cooperação entre elementos e ao foco nos objetivos comuns e centrais dentro de uma organização. Este mesmo modelo tem sido o pilar que sustenta a *World Wide Web* e que tem permitido a sua evolução, criando novos conceitos como *Web 2.0*. Contudo a rapidez da adoção de formatos estandardizados definidos pela *W3C* e pela *OGC*, por parte de todos os intervenientes que disponibilizam conteúdo na Internet, tem sido lenta e ainda exigindo que a estrutura cliente reproduza conteúdos nos mais diversificados formatos; é neste domínio que atuam os *Web browsers*. Um *Web browser* é um software cliente que permite a comunicação com o servidor e a reprodução de formatos diversificados, tais como texto, imagem, som e vídeo.
Por se verificar os benefícios deste modelo computacional e pela necessidade de publicar a informação na *Internet*/*Intranet*, surgiu a necessidade por um *software* baseado na *Web*.

1.2 GESTÃO DE CONTEÚDOS

A informação geográfica muitas vezes encontra-se relacionada com outras informações com o objetivo de melhor contextualizar a informação que se pretende transmitir ao utilizador final. Devido a esta condição, existe a necessidade de associar a informação geográfica a outros conteúdos.

Atualmente, o melhor *software* com características para colmatar a necessidade de gestão e publicação de conteúdos num *Web browser*, enquadra-se na categoria de Gestor de Conteúdos, comummente denominado por *Content Management System*, ou *CMS*. Após uma breve pesquisa constatou-se que já existem soluções de *CMS*, nomeadamente, *Alfresco*, *Drupal* ou *Plone*. Contudo, nenhum destes *CMS* satisfaz as necessidades descritas neste documento como um *todo*, sendo que seria sempre necessário desenvolver novas extensões que permitissem satisfazer estas necessidades. Medir a ponderação sobre adicionar funcionalidades de SIG a um *CMS*, ou adicionar funcionalidades de *CMS* a um sistema SIG não é um exercício trivial mas sim complexo, considerando que existe um vasto leque de soluções em ambos os domínios.

1.3 SERVIDOR DE MAPAS

Um servidor de mapas, é um *software* que permite disponibilizar *webservices WxS* estandardizados pela *OGC,* bem como transformar dados geográficos num mapa. Por este motivo será executado no servidor e responderá a pedidos realizados pelos clientes,

enquadrando-se perfeitamente no modelo computacional cliente-servidor.
A necessidade de uma interface gráfica para o servidor de mapas prende-se com o tipo de utilizador que irá interagir com o sistema. Pretende-se que utilizadores administrativos e sem conhecimentos técnicos sobre o funcionamento de um servidor de mapas, possam realizar tarefas de administração sobre mapas, camadas e locais a publicar na *Internet* ou *Intranet*.

Atualmente, existem algumas soluções em *software* de código aberto para a criação de mapas nomeadamente, *GeoServer* e *MapServer*, colmatando a necessidade de desenvolver um novo *software* servidor de mapas.

1.4 CLIENTE DE MAPAS WEB

Uma vez que existe a necessidade de recorrer a um *software* baseado na *Web*, surgiu consequentemente a necessidade de optar por uma solução que permita reproduzir mapas no *Web browser*. Com base nos fundamentos *Web 2.0*, surge também a necessidade de que os mapas sejam interativos, proporcionando uma melhor experiência de utilização. Também neste domínio, já existem algumas soluções para a reprodução de mapas, nomeadamente, *Leaflet*, *p.mapper* e *OpenLayers*.

1.5 SISTEMA DE GESTÃO DE BASE DE DADOS

Devido à necessidade de criar, manter, consultar e relacionar grandes quantidades de dados, surgiu a necessidade de utilizar um sistema de gestão de bases de dados adequado para realizar estas tarefas incluindo a gestão de dados geográficos. Também neste domínio já existe, pelo menos uma solução: *PostgreSQL* com a extensão *Postgis*.

1.6 PLATAFORMA DE ESCOPO EMPRESARIAL

Alguns conceitos básicos nas plataformas de escopo empresarial (*enterprise*) incluem:

1. pessoa
2. informação
3. *workflow*
4. avaliação
5. tecnologia
6. modelo de negócio (ou operações)

Graças ao modelo computacional cliente-servidor, onde vários clientes acedem simultaneamente ao servidor, pretende-se uma plataforma multi-utilizador. A plataforma deverá também distinguir diferentes funções dos utilizadores e que possa refletir a sua relação existente na organização ou empresa.

Também no escopo empresarial, é necessário um mecanismo de permita o registo, transferência e acompanhamento de tarefas internas, produzindo fluxo de trabalho, ou *workflow,* para a resolução de problemas.

Para que possam existir indicadores sobre a qualidade da informação, com o objetivo de auxiliar na tomada de decisões, é ainda necessário um sistema de avaliação da informação.

1.7 ESTRUTURA MODULAR

Com vista a criar uma plataforma flexível que se possa ajustar a um conjunto de necessidades

específicas que pode variar de organização para organização, pretende-se criar uma estrutura modular que permita adicionar e remover funcionalidades que não existem no *Core* do sistema.

1.8 UM ÚNICO AMBIENTE GRÁFICO INTEGRADO

Como pudemos constatar pelo *software* indicado nas necessidades anteriores, cada um destes programas fornece uma interface própria. Nos casos em que não existe interface gráfica, não são de todo programas indicados para tarefas administrativas para um utilizador comum, mas sim para ser usado como uma ferramenta de programação.
Pretende-se criar um único ambiente gráfico que simplifique a aprendizagem e uso de várias funcionalidades que até agora estavam apenas ao alcance de um utilizador com conhecimentos sobre desenvolvimento de *software*; a criação de um ambiente gráfico não implica uma limitação de acesso às funcionalidades das ferramentas individualmente.

Considere-se a seguinte lista de tarefas de um utilizador administrativo em SIG que não possui conhecimentos sobre desenvolvimento de *software*:

1. registo do utilizador no sistema SIG
2. inserção ou importação de dados geográficos
3. catalogação de dados
4. gestão de conteúdos em formato texto, imagem ou vídeo relacionados com os dados geográficos
5. registo de temas, mapas e locais
6. criação de *webservices WxS*
7. gestão de outros utilizadores no sistema SIG
8. definir permissões de acesso de utilizadores com base na catalogação
9. efetuar auditoria aos dados introduzidos pelos utilizadores
10. criação de tarefas de gestão de dados e delegação ou transferência das mesmas
11. criar, obter, atualizar e remover (*CRUD*) conteúdos não geográficos
12. seleção de conteúdos a publicar (geográficos e não geográficos)
13. publicar conteúdos na *Internet*/*Intranet*

Como podemos constatar pelas tarefas descritas acima, não é possível realizar sem recorrer a mais do que uma interface. Considerando as soluções apresentadas nas necessidades anteriores, poderia-se usar a solução *GeoServer* para as tarefas 1, 2, 3, 5, 6, 7, 8, 9 e 13 e em separado uma solução *CMS* para as restantes tarefas: 4, 10, 11 e 12. De qualquer forma não haveria relação nenhuma entre estes dois sistemas isolados. O sistema final que se apresenta neste documento deverá agregar várias funcionalidades de várias soluções de *software*, permitindo realizar todas as tarefas indicadas acima numa única interface gráfica, aumentando significativamente a produtividade do utilizador.

2. MÉTODOS E RECURSOS UTILIZADOS

Como resposta ás necessidades identificadas no ponto 1 procedeu-se à elaboração do projeto de código aberto baseado em *MapServer*, *Postgis*, *OpenLayers* e *CodeIgniter*.

2.1.1 ARQUITECTURA PADRÃO MVC: *CODEIGNITER*

Sendo necessária uma aplicação Web baseada no modelo cliente-servidor, onde existem dois *runtime* distintos, cliente e servidor, é necessária uma estrutura que represente esta distinção. Contudo, também é necessário que a estrutura seja facilmente entendida por todos os intervenientes no desenvolvimento do software. Existe pelo menos uma arquitetura padrão com uma estrutura que representa esta separação de runtimes: *MVC*. *MVC* é um acrónimo de *Model View Controller* e é uma arquitetura padrão em desenvolvimento de software. Nesta arquitetura está representada a separação de cliente-servidor na medida em que o *Model* e o *Controller* irão atuar no servidor e a *View* irá atuar no cliente.

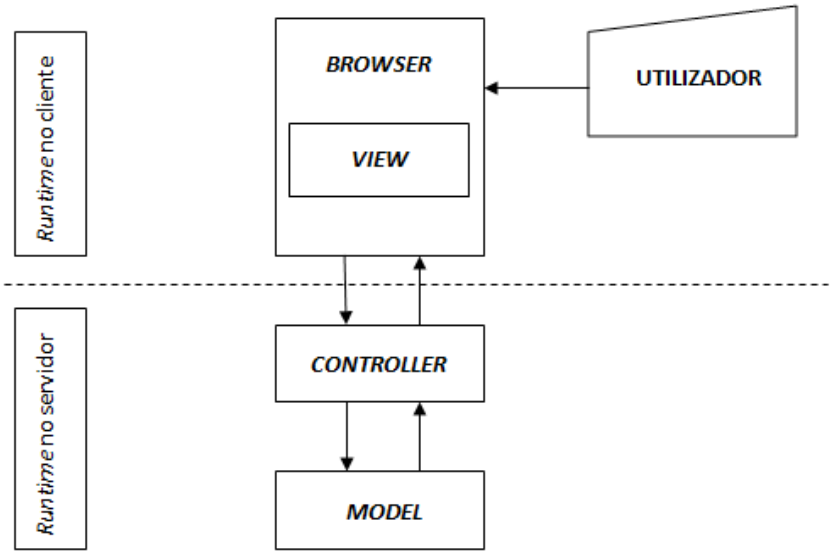


Figura 1 *- Model View Controller*

Pretende-se que este projeto se desenvolva no âmbito de Software Aberto, para tal, é imprescindível a utilização de uma plataforma reconhecida internacionalmente, com ótima documentação e excelente comunidade de suporte.

Também neste domínio já existem várias plataformas que satisfazem estes requisitos; a principal diferença reside na linguagem de programação utilizada. *CodeIgniter* é uma plataforma desenvolvida em *PHP* que se baseia na arquitetura padrão *MVC*, e foi escolhida por corresponder às necessidades indicadas anteriormente. O exercício de comparar esta plataforma com outras também baseadas na arquitetura MVC, não é trivial uma vez que envolve variáveis de medição complexa, como por exemplo a qualidade da documentação, sendo este um critério fulcral na escolha de qualquer ferramenta de desenvolvimento com significativa complexidade.

2.1.2 ABSTRAÇÃO AO MOTOR DE BASE DE DADOS DA APLICAÇÃO: *REDBEANPHP*

Dada a diversidade existente em sistemas de gestão de bases de dados ou SGBD – como será referido a partir deste momento – tais como MySQL, PostgreSQL, Oracle e SQLite entre outros, em que cada um destes sistemas apresenta vantagens e desvantagens, é importante que se possa aproveitar ao máximo os recursos disponíveis no servidor, considerando possíveis migrações entre SGBD. Para que esta flexibilidade seja possível é necessário criar uma camada de abstração no sistema capaz de interagir com diferentes SGBD.

RedBeanPHP é uma plataforma em *PHP* que permite o mapeamento objecto-relacional com suporte a *PostgreSQL*, *MySql*, *SQLite*, *CUBRID* e *Oracle* e foi escolhida por satisfazer estas necessidades e pela fácil integração com a plataforma *CodeIgniter*.

2.1.3 *MAPSERVER*

A opção por MapServer deve-se essencialmente à sua facilidade de integração com o sistema, considerando que a sua API CGI é abstrata à linguagem de programação utilizada pela plataforma de desenvolvimento utilizada neste projeto - CodeIgniter.

Embora a solução *GeoServer* já inclua uma interface gráfica *Web* para a gestão de conteúdos, considerando a adição de funcionalidades que o sistema final oferece e que não se encontram na solução *GeoServer out-of-the-box* – por exemplo associar conteúdo misto HTML a um mapa, camada ou *feature* – verificou-se uma maior facilidade em criar uma interface gráfica para *MapServer*, do que adaptar a atual interface gráfica de *GeoServer*.

Os métodos implementados na integração com MapServer são:

1. Criação de uma interface gráfica para gestão de objetos disponibilizados pela *API* de *MapServer*;
2. Geração automática de ficheiros *Mapfile* com recurso a *templates PHP*;
3. Receção e encaminhamento de pedidos para *MapServer CGI* via controlador *proxy* integrado;
4. Receção de respostas *MapServer CGI* e uso de cache *APC.*

2.1.4 *POSTGRESQL E POSTGIS*

A opção pela solução *PostgreSQL* com a extensão *Postgis* baseou-se principalmente na sinergia que oferece para com todo o sistema, uma vez que é suportado *out-of-the-box* pelo servidor de mapas – MapServer – e pela ORM RedBeanPHP.
Tal como verificado no caso do *GeoServer*, também existe uma interface gráfica *Web* para *PostgreSQL*, denominada *phpPgAdmin*. Como existe a necessidade de no sistema final servir uma única interface, também neste caso verificou-se ser mais fácil criar uma nova interface gráfica, considerando a adição de funcionalidades com foco na informação geográfica que não existem nesta solução – como por exemplo a edição visual de registos com campos do tipo Geometria – do que integrar e adaptar a interface de *phpPgAdmin*.

2.1.5 *OPENLAYERS*

A opção pela solução *OpenLayers* baseou-se nas seguintes condições:

1. Maturidade e número de funcionalidades
2. Qualidade da documentação
3. Grande comunidade de suporte para a resolução de problemas
4. Experiência de utilização de acordo com os fundamentos da *Web 2.0*

Constatou-se não haver uma interface gráfica na solução *OpenLayers* para a realização de tarefas administrativas sobre mapas, camadas e outros elementos disponibilizados pela sua *API*. Por este motivo, surgiu a necessidade de criar uma interface gráfica para este efeito.

2.1.6 SISTEMA MODULAR DE BLOCOS DE LAYOUT

A gestão de módulos, *layouts* e blocos é uma funcionalidade comum dos Gestores de Conteúdos e que satisfaz a necessidade por uma estrutura modular.

Esta estrutura modular assenta essencialmente sobre 4 entidades: Módulo, *Layout*, *Slot* e Bloco.

- Módulo é um conjunto de ficheiros que adiciona funcionalidades ao *Core* do sistema.
- *Layout* é uma *View* na arquitetura *MVC* que apresenta um ambiente gráfico ao utilizador.
- *Slot* é um espaço reservado no *Layout*.
- Bloco é uma instância do módulo configurável em *Slots* de *Layout*.

A representação de um módulo na estrutura MVC indica-se a seguir:

- Controller – classe de controlo que responde ao pedido do utilizador
- Model – Classe do bloco de *layout*; indica quais as ligações e *scripts* a serem carregados na *View* de *Layout*
- *View* – ficheiro PHP que irá conter a apresentação do módulo

2.1.7 SISTEMA DE CONTROLO RBAC

Para distinguir as diferentes funções dos utilizadores do sistema, tais como administrador, editor, autor e público, e para colmatar a necessidade de um sistema multi-utilizador que possa refletir as diferentes funções dos utilizadores dentro de uma organização, o modelo de sistema de controlo, adotado foi o *RBAC*. *RBAC* é acrónimo de *Role Based Access Control*.

2.1.8 FUNCIONALIDADE DE *WORKFLOW*: TICKETS

Foi implementado um sistema de gestão de pedidos de suporte (*ticket*) minimalista mas que permite satisfazer a necessidade por um sistema de registo, transferência e acompanhamento de tarefas. Com a implementação deste sistema é possível iniciar um fluxo de trabalho (*workflow*) com base na criação de pedidos internos ou externos.

Um *ticket* tem indicação do atual utilizador proprietário, o assunto, a mensagem, o estado, uma referência e a data e hora da última atualização. Um *ticket* pode ainda estar associado a um *ticket* "pai", mapa, tema ou local. Uma transferência do *ticket* tem indicação do *ticket* em questão, o último proprietário, um comentário, data e hora da transferência, e o último estado.

2.1.9 AVALIAÇÃO DE CONTEÚDOS

O sistema de avaliação de conteúdos implementado é também minimalista. O sistema adotado permite que os utilizadores possam classificar a qualidade, rigor ou atração da informação existente no sistema, aplicando um voto de valor entre 1 e 5. O valor é representado por uma imagem de uma estrela. Tal como acontece com outros recursos da Web 2.0, os utilizadores podem interagir com a plataforma e expressar a sua opinião quantitativamente.

Um conteúdo avaliado tem indicação do tipo de conteúdo, a identificação do conteúdo, o número total de votos, a média de classificação e a data da última atualização. Um voto tem a indicação do utilizador, a identificação do conteúdo, o endereço IP e a classificação atribuída.

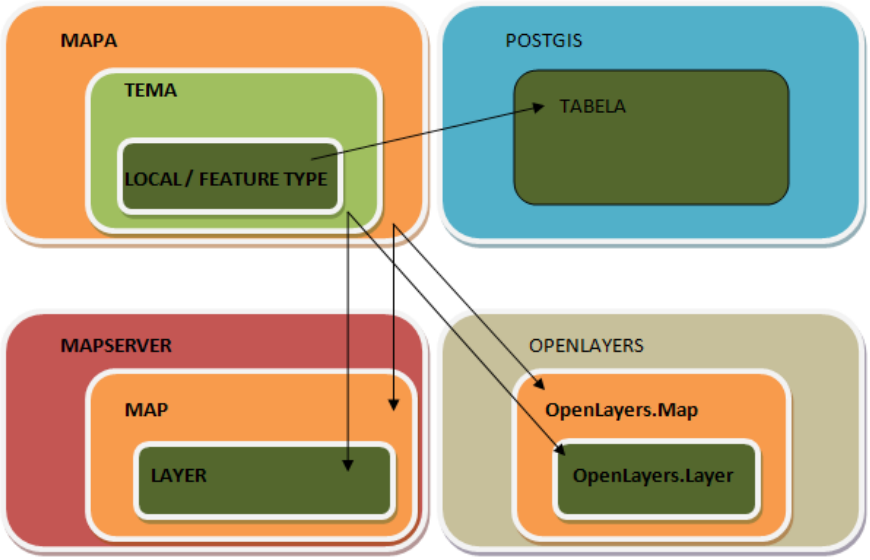
2.1.10 ORGANIZAÇÃO DA INFORMAÇÃO GEOGRÁFICA

Com foco na informação geográfica, pretende-se implementar um modelo de gestão que permita ao utilizador organizar a informação de forma simples e intuitiva. Basicamente são distinguidas 3 entidades: mapa, tema e local.

Mapa é uma entidade abstrata que serve para agregar temas no contexto de informação geográfica. Um mapa pode ter uma representação no servidor de mapas e uma representação no cliente de mapas *Web*.

Tema, ou categoria, é uma entidade abstrata que serve para contextualizar os elementos contidos. Um tema pode ter uma representação no servidor de mapas, como *Layer,* e uma representação no cliente de mapas *Web,* também como *Layer*. Um tema é partilhável entre mapas.

Local, ou tipo de registo, é uma entidade com propriedades de uma localização geográfica, também reconhecida como ***Feature*** em outros sistemas de informação geográfica. O local pode ter uma representação no sistema de gestão de bases de dados, que indica uma tabela que contém um campo do tipo Geometria.



3. RESULTADOS: PLATAFORMA MAPIGNITER

O nome MapIgniter surgiu por combinação de dois nomes dos seus componentes:
Map – como indicação de software para a criação de mapas com recurso a MapServer;
Igniter – como indicação da plataforma *PHP CodeIgniter*.

3.1 CARACTERÍSTICAS GERAIS

MapIgniter apresenta as seguintes características gerais:

- Plataforma *Web* Integrada
- Gestor de conteúdos para publicação na *Internet*/*Intranet*
- Suporte a informação geográfica
- *Software* de código aberto

3.2 CARACTERÍSTICAS ESPECÍFICAS DO CORE

3.2.1 INTERFACE GRÁFICA PARA *MAPSERVER*

A criação de uma interface gráfica para MapServer permitiu aumentar significativamente a acessibilidade no acesso a este software, melhorando a qualidade da experiência de utilização e reduzindo a necessidade pelo conhecimento técnico da sintaxe dos ficheiros *Mapfile*.

A interface permite a gestão de objetos *Maps*, *Layers*, *Styles*, *Labels*, *Legends* entre outros, e configuração de *webservices WxS*.

Redução significativa de erros de sintaxe na construção de ficheiros *Mapfile*.

Aumento de segurança na exposição de caminhos locais no servidor, uma vez que o acesso é realizado via controlador *proxy*.

Aumento de eficiência nas respostas aos clientes com recurso a *cache* PHP APC.

3.2.2 *INTERFACE GRÁFICA PARA POSTGRESQL E POSTGIS*

A criação de uma nova interface gráfica para *Postgis* permitiu melhorar a interação entre o utilizador e esta ferramenta, em especial, a edição visual de registos com campo do tipo Geometria.

Outras funcionalidades disponíveis com esta interface:

1. Importação de ficheiros no formato *Shapefile*
2. Gestão de campos
3. Gestão de registos

3.2.3 INTERFACE GRÁFICA PARA *OPENLAYERS*

A interface gráfica integrada permite no MapIgniter *out-of-the-box,* a criação de objetos da *API OpenLayers*, nomeadamente *OpenLayers.Map e OpenLayers.Layer* entre outros, sem recorrer a programação, o que permite que gestores de informação possam criar mapas em OpenLayers.

A integração com o *OpenLayers* permite as seguintes funcionalidades:

1. Interface gráfica para a gestão de mapas, camadas e integração com módulos
2. Gestão de módulos *OpenLayers*; permite colocar mais do que um mapa no mesmo *layout*

3.2.4 GESTÃO DE MÓDULOS, *LAYOUTS* E BLOCOS

O sistema modular de blocos de *layout* permite a distribuição de responsabilidades, com o objetivo de aumentar a eficiência dos recursos humanos no desenvolvimento de sites baseados em *CMS*. O desenvolvimento de módulos é da responsabilidade do programador. O design do *layout* e do modulo é da responsabilidade do *web designer*. A seleção dos *layouts*, a criação de blocos e configuração nos respetivos *slots,* é da responsabilidade do administrador ou editor do site.

3.2.5 SISTEMA DE CONTROLO: UTILIZADORES, GRUPOS E PERMISSÕES

O sistema de controlo *RBAC* permitiu um maior controlo sobre as ações dos utilizadores, bem como a distribuição de funções de modo a refletir as mesmas numa organização. Desta forma podem-se distribuir responsabilidades consoante a sua função e as ações que poderá efetuar.

As seguintes funcionalidades estão disponíveis:

1. Gerir grupos de utilizadores
2. Gerir recursos de *URL* (com expressões regulares)
3. Gerir contas do utilizador
4. Gerir permissões de grupos aos recursos

3.2.6 GESTÃO DE INFORMAÇÃO GEOGRÁFICA

A gestão de informação geográfica foi significativamente simplificada com recurso a uma interface integrada que permite ao gestor da informação realizar rapidamente operações de adição, alteração ou remoção de dados. A interface elaborada ao estilo de Gestor de Conteúdos representa a informação geográfica organizada em Mapas, Temas e Locais. Em vez se criar estes objetos individualmente nas diferentes ferramentas utilizadas, tais como *MapServer*, *Postgis* e *OpenLayers*, no *MapIgniter* estes objetos são geridos centralmente numa única interface, reduzindo significativamente a necessidade de mudar de ambiente gráfico e de ter conhecimentos de programação para a realização de tarefas simples na gestão de dados geográficos. Não obstante, recomenda-se que tarefas rigorosas sobre registos *Postgis* sejam executadas com recurso a software *Desktop*, como por exemplo, *Quantum Gis*, *gvSig* ou *Autocad*.

3.2.7 PEDIDOS DE SUPORTE (TICKETS)

A interface gráfica integrada permite aos utilizadores acompanharem o movimento de tarefas de uma forma simples e intuitiva. Mesmo que já exista na organização outro software para este efeito, é possível referenciar tarefas de outros sistemas de *workflow* através do campo “referência”.

3.2.8 AVALIAÇÃO DE CONTEÚDOS

Com base no sistema de avaliação de conteúdos, é possível produzir indicadores que possam auxiliar na tomada de decisão sobre a gestão dos conteúdos, ou iniciar tarefas com o objetivo de aumentar a qualidade da informação no sistema. Além disso, é possível produzir listas ordenadas sobre os conteúdos melhor classificados.

3.3 ARQUITECTURA MODULAR

MapIgniter é baseado numa arquitetura modular, permitindo que outras funcionalidades em falta no *Core* possam ser desenvolvidas em módulos e usadas em diferentes *sites* conforme as necessidades.

4. SOBRE DO PROJETO

MapIgniter é um projeto recente, tendo sido apresentado às comunidades OSGeo e OSGeoPT a 28 de Julho de 2012.
É um projeto de código aberto e está alojado no GitHub, permitindo que qualquer pessoa possa colaborar no seu desenvolvimento.

O site http://mapigniter.com inclui uma Wiki em Português e em Inglês com informação detalhada sobre a instalação e uso dos módulos.

Sendo um projeto orientado para o desenvolvimento colaborativo, existe um grupo de discussão no endereço https://groups.google.com/forum/#!forum/mapigniter onde é possível discutir sobre o projeto, tirar dúvidas ou sugerir adição de funcionalidades.

Está também disponível um canal MEO sobre o MapIgniter em http://kanal.pt/959526

MapIgniter usa uma licença dual: GPL e Apache v2

Agradecimentos

Agradecimento ao Dr. Stelmo Barbosa pelo contagiante entusiasmo e forte incentivo ao uso de Software Aberto, bem como pela iniciativa na criação de um gabinete de SIG na Câmara Municipal de Tavira.

Agradecimento aos elementos que compõem a Divisão de Sistemas de Informação e Cartografia da Câmara Municipal de Tavira pela partilha de experiências do uso de Software Aberto em Sistemas de Informação Geográfica.

Agradecimento aos formadores Fred Lehodey e Ricardo Sena, bem como a todos os intervenientes na organização do Curso SIG em Software Aberto promovido pela Associação de Produtores Florestais da Serra do Caldeirão (APFSC).

Agradecimento aos formadores Giovanni Manghi e Duarte Carreira pelos workshops realizados nas III Jornadas em SASIG, Campus de Campolide, ISEGI.

Agradecimento à minha esposa e um abraço aos meus familiares por todo o apoio.

Referências bibliográficas

AB MySQL - 2001 – volalaw.com, MySQL
Burbeck, Steve, 1997, http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html
Burger, Armin - Disponibile su, 2008, p.mapper: a MapServer PHP/MapScript Framework
CMS Plone - Disponível na internet: http://plone.org, 2007, Open Source Content Management
Dabbish, Stuart, Tsay, Herbsleb, Laura, Collen, Jason, Jim - Proceedings of the ACM 2012, 2012 – dl.acm.org, Social coding in github: transparency and collaboration in an open software repository
Deoliveira, Justin - Proceedings of the 10th International Conference, 2008 – gsdi.org, GeoServer: uniting the GeoWeb and spatial data infrastructures
ESRI - URL, July, 1998, Shapefile technical description
Ferraiolo, Cugini, David, Janet, DR Kuhn - Proceedings of 11th Annual, 1995 http://csrc.nist.gov/groups/SNS/rbac/documents/ferraiolo-cugini-kuhn-95.pdf
GNU Debian - 2005, Linux
Griffiths, Adam - 2010 – books.google.com, CodeIgniter 1.7 Professional Development
Hazzard, Erik - 2011 – books.google.com, OpenLayers 2.10 Beginner's Guide
Kropla, Bill - 2005 – books.google.com, Beginning MapServer: open source GIS development
Lerdorf, Tatroe, MacIntyre, Rasmus, Kevin, Peter - 2006 – books.google.com, Programming Php
Momjian, Bruce - 2001 – sgbdfr.info, PostgreSQL: introduction and concepts
O'Reilly, Tim - Message posted to http://radar.oreilly.com, 2005 – tcc-web20.googlecode.com, Web 2.0: compact definition
Open Geospatial Consortium - Url: http://www. opengeospatial. Org, 2010, Inc.(OGC)
Ramsey, Paul - Refractions Research Inc, 2005 – ihg.uni-duisburg.de, PostGIS manual
Rosen, Law - 2004 – rosenlaw.com, Open source licensing
Sanver, Michelle - 2011 – lnu.diva-portal.org, Object Relational Mapping in PHP5
Shariff, Masoud - 2006 - books.google.com Alfresco enterprise content management implementation
Thomas, Keir - 2006 – books.google.com, Beginning Ubuntu Linux: from novice to professional
VanDyk, Westgate, JK, M - 2007 – books.google.com, Pro Drupal Development
Weber, Steven - 2004 - Cambridge Univ Press, The success of open source
Wikipedia, The Free Encyclopedia http://en.wikipedia.org
World Wide Web Consortium, The http://www.w3.org