INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

OSMOSIS-RFID: Sistema de Ficheiros Semântico para Incorporar Objectos Reais no Mundo Virtual

André Filipe dos Santos Mendes

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Júri

Presidente:	Prof. João António Madeiras Pereira
Orientador:	Prof. Luís Manuel Antunes Veiga
Co-Orientador:	Prof. Paulo Jorge Pires Ferreira
Vogal:	Prof. Renato Jorge Caleira Nunes

Maio 2010

Agradecimentos

Quero expressar os meus profundos agradecimentos aos meus orientadores, Prof. Luís Veiga e Prof. Paulo Ferreira. Todo o seu apoio, confiança, acompanhamento e orientação fizeram com que conseguisse terminar este trabalho com sucesso e por isso deixo aqui o meu muito obrigado.

Quero agradecer ao Prof. João Garcia por todos os seus conselhos e ajuda no desenvolvimento do sistema de ficheiros semântico.

Quero também agradecer ao Edgar Marques por toda a ajuda que me deu no desenvolvimento da aplicação do PDA, leitores e restante material RFID.

Deixo aqui um agradecimento ao Filipe Cristóvão, Emanuel, Filipe Silva, André, Filipe Cabecinhas e Andreia por todos os comentários e sugestões ao meu trabalho e pelos muitos momentos de descontracção que me proporcionaram.

Quero ainda agradecer ao João, Helena e Patrícia Gomes por todo o suporte, ajuda e bons conselhos que me deram ao longo desta dissertação.

Aos meus pais e à minha irmã, por me terem sempre dado bons conselhos e apoiado mesmo nos momentos mais difíceis, o meu muito obrigado!

Quero ainda deixar um agradecimento muito especial à Xana. Todo o seu apoio, conselhos, compreensão e paciência foram cruciais nos momentos mais difíceis por que passei na elaboração deste trabalho.

A todos o meu Muito Obrigado!

Resumo

Com o aumento da capacidade de armazenamento dos computadores, tornou-se evidente que o uso de um sistema de ficheiros hierárquico, actualmente, já não se adequa aos interesses do utilizador, ou seja, não permite encontrar um ficheiro de forma fácil, rápida e intuitiva. Por esta razão foi desenvolvida a noção de Sistema de Ficheiros Semântico. Este é um sistema de armazenamento e organização de informação, que permite associar atributos aos ficheiros e posteriormente encontrá-los, de forma intuitiva, a partir desses atributos.

Todas as limitações existentes nos sistemas hierárquicos levaram-nos ao desenvolvimento do OSMOSIS. Este é um sistema de ficheiros semântico com a capacidade de integrar objectos físicos no mundo virtual. O OSMOSIS pode ser dividido em duas grandes partes. A primeira (OSMOSIS-SFS) é um sistema de ficheiros semântico que permite associar aos ficheiros, tanto atributos extraídos das propriedades e do conteúdo dos ficheiros, como relações explícitas entre dois ficheiros e até mesmo atributos (*Tags*) introduzidos pelo utilizador. A segunda parte (OSMOSIS-RFID) consiste na integração do sistema semântico com o mundo real, ou seja, permitir a associação de *Tags* RFID a localizações físicas (por exemplo sala de estar) ou objectos físicos, relacionados com ficheiros virtuais (por exemplo álbum de fotografias) e posteriormente usar essa informação para ajudar a encontrar objectos físicos, ou antecipar que ficheiros o utilizador poderá querer ver, com base na sua localização física.

Palavras-chave: sistema de ficheiros semântico, *Tags* semânticas, *Tags* RFID, pesquisa de ficheiros, directorias virtuais, localização de objectos físicos.

Abstract

With the increasing storage capacity of computers, it became evident that the use of a hierarchical file system no longer fits the interests of the user, i.e., with this kind of file system the user cannot find a file in an easy, quick and intuitive way. For this reason, the concept of Semantic File System became quite popular. This is a system for storing and organizing information, that allows associations between attributes and files and then searching those files using attributes (or keywords).

All the limitations of hierarchical systems led us to the development of OSMOSIS. This is a semantic file system with the ability to integrate physical objects in the virtual world. OSMOSIS can be divided into two major parts. The first (OSMOSIS-SFS) is a semantic file system that can associate, with files, attributes extracted from its properties and contents, can explicitly relate two files, and allows users to manually apply a Tag to a file. The second part (OSMOSIS-RFID) is the integration of the semantic system into the real world. In other words, with this system it will be possible to attach RFID Tags to physical locations (e.g. living room) or physical objects, related to virtual files (e.g. photo album) and then use that information to find physical objects, or to anticipate what files the user may want to see, based on his/hers physical location.

Keywords: semantic file system, semantic Tags, RFID Tags, file search, virtual directories, location of physical objects.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O aumento da capacidade de armazenamento dos computadores levou ao enorme crescimento do número de ficheiros armazenados nestes. Devido a este grande crescimento por vezes encontramos ficheiros, no nosso computador, que nós próprios já nem nos lembramos que possuíamos.

Actualmente a maioria dos Sistemas Operativos utiliza um sistema hierárquico de directorias para organizar os ficheiros. Assim, para um utilizador aceder a um ficheiro terá de fazer *browsing* através das directorias ou então especificar o caminho completo (*path*). Em qualquer dos casos o utilizador é obrigado a lembrar-se onde o ficheiro se encontra armazenado. Apesar deste sistema hierárquico ser útil para computadores com poucos ficheiros, quando se está na presença de uma grande quantidade de dados, torna-se por vezes difícil a um utilizador lembrar-se da localização exacta de um determinado ficheiro. Este facto acontece porque uma hierarquia de directorias é inflexível e não permite registar associações entre ficheiros de directorias diferentes, dificultando a tarefa de encontrar um ficheiro em específico, que se pode enquadrar em mais de uma directoria. Em vez disso é bastante mais intuitivo e fácil lembrarmo-nos de partes do nome do ficheiro, de palavras contidas neste, ou até mesmo do contexto em que foi criado.

Para ajudar a resolver este problema foi criada a noção de Sistema de Ficheiros Semântico [10]. Um Sistema de Ficheiros Semântico é um sistema de armazenamento e organização de informação, que permite extrair automaticamente atributos dos ficheiros e posteriormente encontrá-los a partir desses atributos. A semântica dos ficheiros é, neste sentido, um conjunto de características que fazem parte dos ficheiros, mas que não estão necessariamente relacionadas com a sua localização na hierarquia de pastas.

Além dos atributos extraídos automaticamente dos ficheiros, é também possível ao utilizador definir atributos, através da colocação de *Tags* (etiquetas). Estas *Tags* são palavras-chave que podem ser atribuídas a um ficheiro, ou mesmo a um conjunto de ficheiros, e que permitem identificar, de forma mais intuitiva, um dado ficheiro. Este é um conceito de utilização quotidiana na Web, mas que no entanto ainda se encontra pouco explorado no contexto de ficheiros locais, ao qual pode ser aplicado [25]. É também necessário realçar que os atributos extraídos automaticamente pelo sistema são no fundo *Tags*, mas sendo estas automaticamente atribuídas.

Outro tipo de *Tags*, que não as referidas anteriormente, são as *Tags* de *Radio-Frequency IDentification* (RFID), ou *Tags* físicas. A tecnologia RFID pode ser dividida em dois grandes componentes. O primeiro são *Tags* RFID, que não são mais do que circuitos integrados, onde é guardada informação acerca de uma determinada entidade. Como as *Tags* RFID transmitem a sua informação através de um sinal de rádio-frequência, a segunda parte desta tecnologia consiste num *reader*, que tem como função receber o sinal de rádio, transmitido pela *Tag*, e convertê-lo em informação útil.

Estas *Tags* físicas podem ser associadas ou incorporadas a objectos físicos no mundo real, de modo a existir um relação de um para um entre objecto e *Tag* RFID. Assim sendo, é possível fazer uma analogia entre um ficheiro, ou directoria, e as *Tags* RFID, porque por exemplo a directoria `Fotografias 2008`,

que contém o conjunto de fotografias de 2008, presentes no PC, pode corresponder à *Tag* RFID que está colada no álbum físico de fotografias de 2008.

Encontrando-se todos os ficheiros classificados com uma ou mais *Tags*, é então possível usar essa informação para mais facilmente se encontrar o ficheiro pretendido. Um utilizador pode simplesmente efectuar uma pesquisa, utilizando uma ou mais *Tags* como palavras-chave, sendo retornados todos os ficheiros que estão marcados com essa(s) *Tag(s)*, ou então navegar através de directorias virtuais. Neste último caso o utilizador começa por escolher uma directoria virtual, que corresponde a uma *Tag*, e à medida que for seleccionando pastas virtuais, na verdade navegando através de outras *Tags*, o conteúdo destas directorias irá ser computado nesse instante para que seja apresentado. Uma forma de optimizar o sistema será guardar em cache resultados de navegações anteriores, para que não seja necessário realizar processamento repetido. Na realidade estas directorias não existem em disco, são apenas representações das *Tags* e dos ficheiros que cada uma abrange.

1.1 Motivação

Tal como foi referido anteriormente, os sistemas de ficheiros actuais utilizam uma visão hierárquica de organização, o que se torna desadequado à quantidade de informação guardada nos PC's hoje em dia. Como uma hierarquia de directorias é inflexível e não permite registar associações entre ficheiros de directorias diferentes, a tarefa de encontrar um ficheiro em específico torna-se mais complicada. Por esta razão, tornou-se evidente a necessidade de tornar a navegação pelo sistema de ficheiros, uma tarefa mais flexível e intuitiva para o utilizador.

Os sistemas de ficheiros semânticos vêm, neste sentido, tornar a utilização do sistema de ficheiros, numa tarefa menos rígida e mais ligada com o modo como os utilizadores pensam no mundo real. Na verdade, é muito mais fácil para um utilizador lembrar-se de partes do nome do ficheiro, ou até mesmo do assunto desse ficheiro, do que o *path* completo para esse mesmo documento. Neste tipo de sistemas, os ficheiros podem ser resumidos a um conjunto de palavras-chave, que estão associadas a cada um, e que podem ser facilmente lembradas pelo utilizador. É importante referir ainda que, todos os sistemas semânticos devem possuir uma interface de pesquisa, que permita fazer uso da informação semântica recolhida de forma intuitiva. Os diversos tipos de informação semântica que pode ser associada aos ficheiros, serão descritos mais à frente no Capítulo 2.

No âmbito do nosso trabalho, mais do que um simples sistemas de ficheiros semântico, o OSMOSIS pretende funcionar como um localizador de objectos físicos (relacionados com ficheiros virtuais). Com o nosso sistema irá ser possível associar *Tags* RFID a localizações físicas (por exemplo sala de estar) ou objectos físicos (por exemplo álbum de fotografias) e posteriormente usar essa informação para ajudar a encontrar objectos físicos. Importante ainda referir que o utilizador deverá está munido de um PDA com leitor de *Tags* RFID, para que possa interagir com o mundo real.

Considere-se os dois exemplos seguintes como forma de apresentar as potencialidades do nosso sistema:

1. O utilizador coloca uma *Tag* RFID na porta da sala de estar e associa à localização física "Sala de Estar" as directorias `ColdPlay` e `Fotografias Ferias 2008`. Note-se que estas directorias, estão directamente relacionadas com os objectos físicos "cd de ColdPlay" e "álbum de fotografias das férias de 2008", respectivamente. Assim, quando um PDA ler a *Tag* RFID que está na sala de estar, o utilizador será imediatamente informado acerca da sua localização e de quais são os objectos que se encontram nessa mesma localização.

2. O utilizador coloca uma *Tag* RFID no álbum de fotografias das férias de 2008 e associa este objecto à directoria `Fotografias Ferias 2008`, existente no sistema de ficheiros. Deste modo, quando o

PDA ler essa *Tag* RFID, irá apresentar a directoria `Fotografias Ferias 2008` ao utilizador, para que este possa pré-visualizar as fotografias do álbum.

1.2 Limitações das Soluções Actuais

Actualmente existem diversos estudos e sistemas desenvolvidos na área dos sistemas de ficheiros semânticos, como se pode verificar no Capítulo 2. Estes variam entre si, principalmente pelo tipo de informação semântica recolhida dos documentos. Esta pode ser baseada nos metadados, no conteúdo de ficheiros de texto, no contexto de utilização dos ficheiros e até mesmo nas *Tags* atribuídas manualmente aos ficheiros pelos utilizadores.

No entanto, dos 23 sistemas estudados, apenas 3 complementam a informação semântica extraída automaticamente, com as *Tags* atribuídas manualmente pelo utilizador. Ou seja, praticamente todos os sistemas concentram-se apenas numa das categorias de sistemas de ficheiros semânticos, não integrando as vantagens das restantes categorias. Para além desta clara limitação, os sistemas estudados também não fornecem uma interface gráfica, intuitiva e apelativa ao utilizador, o que torna a compreensão do sistema de pesquisa, uma tarefa possivelmente complexa. Por fim, constatámos também, que praticamente todos os sistemas semânticos existentes, foram desenvolvidos para os sistemas operativos Linux e Mac OS X, existindo por isso, pouca compatibilidade com o Microsoft Windows, que é o sistema operativo mais utilizado em todo o mundo.

Para além de todas as limitações descritas acima, podemos também constatar que nenhum dos sistemas de ficheiros semântico estudados, faz uma integração do mundo real no mundo virtual. Ou seja, em nenhum deles existe uma forma de integrar a tecnologia com o espaço físico, não possibilitando assim que estes sistemas possam ser também úteis no mundo real (como por exemplo funcionar como um sistema de localização de objectos).

O sistema OSMOSIS vem assim colmatar as limitações descritas acima e por isso, possui características de todas as categorias de sistemas de ficheiros semânticos existentes. Ou seja, extrai automaticamente atributos a partir dos metadados dos ficheiros e do conteúdo de ficheiros de texto, utiliza o contexto (relações explícitas entre ficheiros) como fonte de informação semântica e permite que o utilizador atribua manualmente *Tags* aos ficheiros. Para além disso, o OSMOSIS foi desenvolvido para funcionar no sistema operativo Microsoft Windows, fornecendo uma interface gráfica, onde tivemos em atenção a facilidade de utilização. Por fim, no sistema OSMOSIS é também possível fazer a integração do mundo real no mundo virtual, ou seja, os ficheiros virtuais podem estar directamente relacionados com objectos físicos ou até mesmo com localizações do mundo real.

1.3 Objectivos do Trabalho

O objectivo global do trabalho é a criação de um sistema que integre objectos virtuais, em sistemas de ficheiros, com objectos reais marcados com *Tags* RFID. No entanto, este objectivo global pode ser dividido em dois grandes objectivos. O primeiro é a criação de um Sistema de Ficheiros Semântico (OSMOSIS-SFS), que irá permitir atribuir *Tags* aos ficheiros e posteriormente utilizar essa informação, semântica, para encontrar mais facilmente um dado ficheiro. O segundo objectivo é a integração do Sistema de Ficheiros Semântico com o Mundo Real, ou seja, pretende-se que objectos físicos sejam marcados com *Tags* RFID, existindo uma relação entre estas e os ficheiros virtuais existentes no sistema de ficheiros nativo do Sistema Operativo. Este sistema é denominado OSMOSIS-RFID, e com ele irá ser possível, por exemplo, ao entrar numa sala de estar, ler a sua *Tag* RFID e saber que objectos estão no seu interior.

Nas duas secções seguintes, serão abordados cada um destes objectivos específicos em particular.

1.3.1 Sistema de Ficheiros Semântico

Tal como dito anteriormente, um dos objectivos do trabalho é a criação de um Sistema de Ficheiros Semântico. Neste sentido, o que se pretende realizar é um novo sistema de ficheiros que coexiste com o sistema hierárquico do Sistema Operativo, mas que permite encontrar um determinado ficheiro sem ser necessário que o utilizador se lembre do seu caminho completo (*path*). Este sistema vai conseguir extrair automaticamente atributos (ou *Tags*) a partir dos metadados, como por exemplo o autor, a data de criação, a data de modificação. Além dos metadados é também possível extrair atributos a partir de ficheiros de texto, como por exemplo palavras que fazem parte do seu conteúdo.

Com a grande quantidade de atributos extraídos automaticamente, torna-se necessário fazer uma correcta gestão de modo a que estes se encontrem actualizados. Para isso, o OSMOSIS-SFS irá monitorizar o acesso aos ficheiros e sempre que exista uma leitura ou escrita, vai actualizar os atributos (ou *Tags*) do ficheiro acedido.

Para além do conteúdo, o nosso sistema pretende utilizar o contexto como fonte de informação semântica. Não se pretende que este tipo de informação seja extraída automaticamente, mas sim, que seja o utilizador a relacionar explicitamente dois ficheiros. Como exemplo, considere-se o caso em que um utilizador relaciona a directoria `Fotografias Ferias 2008`, com um documento pdf com o roteiro da viagem que efectuou nas férias de 2008. Nesta situação, pretende-se que quando o utilizador faça uma pesquisa pelas fotografias das férias de 2008, também seja apresentado o ficheiro pdf, como ficheiro relacionado.

Além de extrair automaticamente *Tags* a partir do conteúdo dos ficheiros, no OSMOSIS-SFS, pretendemos também que o utilizador tenha a possibilidade de atribuí-las manualmente aos ficheiros. Estas são simplesmente palavras-chave associadas aos ficheiros, como por exemplo "Relatorio" ou "Tese de Mestrado".

Após colocação de *Tags* nos ficheiros, o OSMOSIS deverá ter mecanismos que fazendo uso dessa informação, permitam ao utilizador encontrar, mais facilmente, o ficheiro pretendido. Um destes mecanismos pode ser o uso de directorias virtuais. Neste caso, as directorias não existem em disco, são apenas representações das *Tags* e dos ficheiros que cada uma abrange. Além disso, o conteúdo de cada directoria irá ser computado à medida que cada uma seja seleccionada durante a navegação. Considere-se o seguinte exemplo de utilização do sistema, onde o utilizador pretende encontrar as fotografias das férias de 2008. Um utilizador começa por realizar uma pesquisa pelos termos "imagens" (computação da directoria virtual `imagens`), e como resultado é devolvido, entre outros, a directoria virtual `Fotografias Ferias 2008` (que corresponde à *Tag* "Fotografias Ferias 2008"). De seguida, selecciona esta pasta e como resultado deverão ser apresentadas as fotografias pretendidas.

Os detalhes da arquitectura do OSMOSIS-SFS irão ser descritos à frente, no Capítulo 3.

1.3.2 Integração com o Mundo Real

A segunda fase do trabalho tem como objectivo fazer a integração do OSMOSIS-SFS, descrito anteriormente, com objectos e localizações do mundo físico. Para isso pretende-se desenvolver uma versão do OSMOSIS-SFS com menos funcionalidades (cliente remoto), para ser instalada num PDA. Esta versão já não irá disponibilizar a funcionalidade de extrair automaticamente *Tags* dos ficheiros, nem irá permitir que o utilizador introduza manualmente *Tags*. Para esta versão poder funcionar correctamente, o PDA (ou cliente) terá uma ligação ao Servidor (máquina *desktop* onde está a instalado o sistema de ficheiros semântico) e é através desta ligação que vai receber informação semântica acerca dos ficheiros.

Para além da ligação ao Servidor, o PDA estará equipado com um leitor de *Tags* RFID, para poder ler as *Tags* que estão coladas aos objectos. De referir ainda que, todos os objectos e localizações (como escritório ou sala de estar) que o utilizador pretende integrar com o OSMOSIS-SFS, devem estar marcados

com *Tags* RFID, mas apenas estes. Além disso, o código da *Tag* tem de ser previamente registado no sistema OSMOSIS-SFS como sendo um determinado objecto ou localização, para que o PDA possa posteriormente fornecer informação útil ao utilizador. De seguida irá ser apresentado um exemplo de como o PDA poderá interagir com o OSMOSIS-SFS.

Quando um utilizador entra numa sala de estar, com o seu PDA, lê a *Tag* RFID referente a essa sala e de seguida:

1. O PDA comunica com o Servidor, informando-o que leu a *Tag* RFID "Sala de Estar".
2. De seguida o Servidor acede à base de dados, onde está guardada a informação semântica e verifica que na Sala de Estar se encontra a caixa de fotografias das férias de 2008 e o álbum físico de ColdPlay. Confirma ainda que estes objectos físicos estão relacionados, respectivamente, com uma directoria denominada `Fotografias_Ferias_2008` (que contém as fotografias em formato JPEG) e uma outra denominada `Album_ColdPlay` (que contém as músicas em formato MP3).
3. O Servidor envia esta informação ao PDA e neste abre-se uma vista, denominada "Sala de Estar". No seu interior encontram-se duas pastas virtuais: `Fotografias_Ferias_2008` e `Album_ColdPlay`, que o utilizador pode explorar. Estas pastas virtuais são uma representação das directorias com o mesmo nome que existem no servidor.
4. Se o utilizador do PDA pretender aceder a um ficheiro, o PDA comunica com o Servidor para que este lhe envie o ficheiro e quando for recebido o utilizador poderá aceder-lhe normalmente.

Este exemplo mostra uma das principais funcionalidades existentes no OSMOSIS-RFID, sendo também um caso típico de utilização do nosso sistema.

1.4 Contribuições

As três principais contribuições do sistema OSMOSIS são as seguintes:

- Criação de um Sistema de Ficheiros Semântico (OSMOSIS-SFS), que irá permitir atribuir *Tags* aos ficheiros e posteriormente utilizar essa informação, semântica, para encontrar mais facilmente um dado ficheiro.
- Integração do OSMOSIS-SFS com o sistema de ficheiros nativo do Microsoft Windows, permitindo ao utilizador continuar a a navegar normalmente no sistema hierárquico, mas fornecendo as novas funcionalidades semânticas do OSMOSIS-SFS.
- Criação de um sistema para PDA, onde é possível integrar o mundo real no mundo virtual. Ou seja, este sistema pode funcionar como um localizador de objectos, ao mesmo tempo que permite consultar ficheiros virtuais, relacionados com objectos físicos. Para além disso, fornece a possibilidade de pesquisar, semanticamente, os ficheiros existentes no Servidor (máquina *desktop*).

1.5 Estrutura da Dissertação

O restante desta dissertação está organizado da seguinte forma:

- **Capítulo 2** - Análise dos sistemas desenvolvidos na área dos sistemas de ficheiros semânticos.
- **Capítulo 3** - Apresentação da arquitectura do sistema OSMOSIS.

- **Capítulo 4** - Descrição dos detalhes de implementação de todos os módulos que fazem parte do sistema OSMOSIS.
- **Capítulo 5** - Resultados da avaliação qualitativa e quantitativa realizada ao nosso sistema.
- **Capítulo 6** - Conclusões do trabalho desenvolvido e apresentação do trabalho futuro.

Capítulo 2

Trabalho Relacionado

Com base no tipo de informação semântica, os trabalhos realizados na área dos sistemas de ficheiros semânticos podem ser divididos em três categorias principais:

- Semântica baseada no conteúdo.
- Semântica baseada no contexto (também inclui conteúdo).
- Semântica baseada nas *Tags* introduzidas pelo utilizador.

No caso da primeira categoria, os sistemas obtêm automaticamente a informação semântica, a partir de atributos, como por exemplo autor, data de criação, data da última modificação, nome, tamanho e tipo de ficheiro. Além disso, estes sistemas também se caracterizam pelo facto de obterem informação semântica a partir do próprio conteúdo do ficheiro, como por exemplo palavras que se sejam encontradas neste. Neste caso, existe a desvantagem de só se poder fazer esta análise para ficheiros de texto. Os sistemas que se inserem nesta categoria são descritos em [1, 3, 4, 6, 7, 10, 12, 16, 17, 18, 19].

No caso do contexto, os sistemas, para além de poderem obter informação semântica do mesmo tipo que os sistemas anteriores, também obtém esta informação a partir do contexto da utilização do ficheiro (ou contexto do ficheiro de forma resumida). O contexto de um ficheiro pode ser, por exemplo, outros ficheiros que sejam acedidos concorrentemente com este, ou mesmo tarefas que o utilizador desempenhe em simultâneo com o acesso ao ficheiro. Para além disso, este contexto também pode estar relacionado com o mundo real, quando por exemplo, um utilizador classifica uma música MP3 como boa, normal ou má. Este tipo de informação semântica permite, deste modo, que se estabeleça relações entre um ficheiro e outros ficheiros, pessoas ou conceitos. Nesta categoria incluem-se sistemas como os descritos em [2, 15, 22, 23, 26, 27, 29].

Na terceira categoria, os sistemas utilizam como informação semântica, as *Tags* introduzidas pelo utilizador. Estas *Tags* são palavras-chave que são associadas aos ficheiros. Nestes sistemas, como a mesma *Tag* pode ser associada a um ou mais ficheiros, todos os que se encontrem marcados com esta estão relacionados entre si. Os sistemas que se incluem nesta categoria são descritos em [5, 28].

Para além dos sistemas mencionados acima, existem ainda outros que não se enquadram apenas numa destas categorias, mas sim em várias. Temos o caso do DBFS [11] e do Insight [14] que possuem características das categorias "Semântica Baseada no Conteúdo" e "Semântica Baseada nas *Tags* Atribuídas". Já o sistema SemDAV [24], possui características das categorias "Semântica Baseada no Contexto" e "Semântica Baseada nas *Tags* Atribuídas". Estes sistemas, que podemos denominar de híbridos, serão descritos na última secção deste capítulo.

2.1 Semântica Baseada no Conteúdo

Nesta secção irá ser feita uma descrição dos sistemas apresentados em [6, 10, 12, 17, 19], procurando-se analisar o modo como cada sistema trata as seguintes propriedades e aspectos chave identificados para esta categoria de sistemas:

- Tipo de ficheiros suportados
- Utilização de directorias virtuais
- Representação e organização da informação

2.1.1 SFS

O SFS (Semantic File System [10]) foi um dos primeiros sistemas a ser desenvolvido na área dos sistemas de ficheiros semânticos e, muitos dos posteriores sistemas desta área estão de alguma forma relacionados com as ideias lançadas pelo SFS.

Para que este sistema funcione correctamente é necessário existir indexação dos ficheiros e directorias, ou seja, a informação semântica associada a estes deverá ser extraída e guardada para posterior utilização. E de modo a manter o sistema semântico sempre actualizado, deverá proceder-se a uma indexação automática sempre que um ficheiro, ou directoria, é criado ou actualizado. Para esta indexação, existe no SFS um *transducer* específico para cada tipo de ficheiro, do qual se pretende extrair atributos (informação semântica). Estes *transducers* podem ser considerados filtros, que recebem como entrada um dado ficheiro, produzindo como resultado o conjunto de atributos que caracterizam o ficheiro. Estes atributos são pares atributo-valor *field-value*, onde *field* representa uma propriedade de um ficheiro (como autor, nome, tamanho) e *value* é uma palavra ou um número. De referir ainda que é possível existir, para o mesmo ficheiro, vários atributos com o mesmo *field*. Por exemplo, para um ficheiro .txt, existem vários atributos em que o *field* é text: e os *values* são as várias palavras contidas no documento.

Os ficheiros suportados pelo SFS são principalmente ficheiros cujo conteúdo é ASCII, não sendo necessariamente documentos de texto. Podem ser, por exemplo, ficheiros C, Pascal ou Scheme, de onde se retiram atributos como funções e variáveis globais, ou até mesmo artigos de notícias, retirando-se atributos como categoria e título. No entanto não são suportados ficheiros de vídeo, músicas ou imagens. O facto de não suportar alguns tipos de ficheiros tem a ver com a inexistência de *transducers* específicos para esses ficheiros, por isso, bastaria fazer um *transducer* para um tipo/formato de ficheiro e este passaria a ser suportado pelo SFS.

De modo a utilizar a informação semântica para mais facilmente encontrar um dado ficheiro, o sistema SFS aplica o conceito de directorias virtuais. Estas directorias não existem na realidade, sendo computadas quando é necessário aceder ao seu conteúdo. Por exemplo, uma directoria virtual pode ser `author:` e dentro desta encontrar-se as directorias virtuais, `joão` e `andré`, que correspondem aos ficheiros cujo autor é o João e o André, respectivamente. Para uma maior compatibilidade com o software existente, no SFS os nomes das directorias virtuais são interpretados como *queries*. De seguida será apresentado um exemplo, retirado de [10], onde é possível verificar esta propriedade.

```
$ ls -F /sfs/owner:/smith
bio.txt@ paper.tex@ prop.tex@
$
```

Neste exemplo é mostrado que ao fazer /owner:/smith está-se a pesquisar por ficheiros que tenham o atributo owner: smith, ou seja, todos os ficheiros que pertencem ao Smith. Além disso, é possível adicionar mais atributos para refinar a pesquisa, podendo ter-se *queries* do tipo /owner:/smith/text:/resume. Importante ainda referir que os ficheiros listados são apenas *links* simbólicos para os ficheiros reais.

2.1.2 SemFS

Tal como o SFS, descrito anteriormente, o SemFS [19] é um sistema que atribui a semântica aos ficheiros com base nos seus metadados e atributos. Neste sistema todos os ficheiros são suportados, uma vez que, todos os ficheiros têm pelo menos atributos como data de criação, data da última modificação, tipo de ficheiro. E mesmo ficheiros que não permitem extrair semântica directamente a partir do seu conteúdo, como imagens ou músicas, têm outros atributos como altura, largura, modelo da câmara fotográfica, no caso das imagens e atributos como tamanho, artista, álbum, no caso das músicas.

O SemFS possibilita a pesquisa de ficheiros com base nos seus atributos, utilizando directorias virtuais para representar a informação e resultados das pesquisas. De referir ainda que este sistema, ao contrário do SFS, permite a utilização de operadores lógicos como AND ^, OR |, NOT !, de modo a filtrar os resultados da pesquisa e consequentemente encontrar mais facilmente o ficheiro pretendido. De seguida irá ser apresentado um exemplo, retirado de [19], que mostra uma forma de pesquisa, assim como a utilização de operadores lógicos.

```
$ cd type:mp3 ^ len>3m ^ artist:Mike
```

Com este comando, o utilizador iria para uma directoria virtual contendo todos os ficheiros MP3 com mais de 3 minutos, do artista Mike.

O SemFS implementa também um sistema de "vistas", que são um género de directorias virtuais persistentes. Assim, o conteúdo de uma directoria virtual deste tipo estaria computado e guardado em *cache*, para evitar calcular o seu conteúdo sempre que se pretendesse aceder-lhe. Estas "vistas" são criadas apenas para as pesquisas mais comuns, evitando-se assim processamento desnecessário, sendo actualizadas quando um dos ficheiros contidos na "vista" é actualizado.

2.1.3 Nebula

O sistema de ficheiros Nebula [6], tal como os anteriores, pretende extrair um conjunto de atributos de cada ficheiro, para que a sua identificação se torne mais intuitiva. Estes atributos são pares atributo/valor (*tag/value*) que representam as propriedades de um ficheiro, estando organizados e guardados num Nebula *file object*. Ou seja, neste sistema, um ficheiro é representado por um objecto que contém todas as suas propriedades.

```
((uid !ab09)
(name hello.c)
(file-path HOME/source/C)
(type C-source)
(description The canonical first C program that prints "hello world"on standard output)
(functions main)
(includes stdio.h)
(text #include "stdio.h"
main(){
printf("Hello World");
}))
```

Este exemplo mostra um Nebula *file object* contendo os atributos de um ficheiro de código C.

Os atributos dos ficheiros são extraídos automaticamente pelo sistema, no entanto é necessário garantir que cada tipo de ficheiro tem uma estrutura bem definida, de modo a obter atributos que melhor identifiquem os ficheiros. Para auxiliar na obtenção de informação semântica, o Nebula implementa dois módulos: o *enforcer* e o *collector*. O primeiro facilita a criação de ficheiros, no Nebula, com uma estrutura bem definida, ao passo que o segundo importa ficheiros de outros sistemas de ficheiros, usando uma gramática específica para cada tipo de ficheiro.

De modo a encontrar um ficheiro pretendido, o Nebula implementa um sistema de vistas (género de directorias virtuais), assim como possibilita procuras por vários atributos e com operadores lógicos. Com esta combinação entre navegação pelas vistas e procuras, obtém-se uma organização dinâmica, ao invés do que acontece nos sistemas de ficheiros hierárquicos. Assim, ao realizar-se uma procura, pode ser apresentado apenas um ficheiro, caso a *query* seja suficientemente precisa, ou então ser apresentada uma vista temporária que contém todos os ficheiros e vistas que correspondem aos atributos introduzidos. Nesta vista temporária é possível navegar ou refinar a *query* de modo a obter uma nova vista, com resultados mais específicos.

2.1.4 GLIMPSE

Ao contrário dos sistemas que foram analisados até agora, o GLIMPSE [17] é um sistema de procura de ficheiros. As suas principais funções são indexar ficheiros e permitir pesquisas (complexas) com vários atributos e opções, de modo a encontrar mais rapidamente o ficheiro pretendido.

Os atributos que o GLIMPSE retira dos ficheiros, são apenas palavras que se encontrem no seu conteúdo e o seu nome. Assim, os únicos ficheiros suportados por este sistema são os ficheiros de texto. Aliás, antes de indexar um ficheiro, o sistema verifica se este é ou não de texto e caso não seja, não é indexado. Para além disso, é dada ao utilizador, a possibilidade de assinalar um ficheiro para não ser indexado.

O GLIMPSE não possui um sistema de navegação através de directorias virtuais, em vez disso, tem um sistema de pesquisa, que permite introduzir vários termos a procurar, utilizando operadores lógicos e com várias opções sobre a própria pesquisa. De seguida serão apresentados alguns exemplos, retirados de [17], que mostram como estas pesquisas podem ser complexas.

```
$ glimpse 'Winter Usenix'
```

Esta pesquisa retorna todos os ficheiros que contêm as palavras "Winter" e "Usenix" no seu conteúdo.

```
$ glimpse -1 HardToSpell
```

Encontra todos os ficheiros que contêm a palavra "HardToSpell" com um erro de escrita.

```
$ glimpse -w 't[a-z]j@#uk'
```

Devolve todos os endereços de e-mail cujo nome de login tem três letras, começa com a letra t e acaba com a letra j e o nome do servidor tem "uk" algures. A opção -w indica que apenas interessam as correspondências exactas com o padrão.

Uma grande qualidade do GLIMPSE é o facto de utilizar um índex bastante pequeno, não ocupando uma grande quantidade de memória com os atributos dos ficheiros. Isso deve-se ao método utilizado para indexação, *two-level searching* (procura em dois níveis). Neste método, começa-se por dividir a totalidade de ficheiros a indexar por vários conjuntos, denominados blocos. De seguida é feita uma análise de todos os ficheiros, de todos os blocos, palavra por palavra e é construído um índex. Este índex contém todas as palavras encontradas, mas não todas as ocorrências de uma palavra, ou seja, não existem palavras repetidas no índex. Cada entrada contém uma palavra e o número dos blocos em que essa palavra se encontra.

Devido à forma como é construído o índex, a pesquisa é feita em duas fases. A primeira consiste numa procura sequencial do índex, listando todos os blocos que têm correspondência com a *query* introduzida. Na segunda fase é feita uma procura sequencial por cada um desses blocos, devolvendo de seguida os resultados. De referir ainda que para a procura sequencial, tanto do índex, como de cada bloco, é utilizada a ferramenta de pesquisa *agrep*. Esta ferramenta é usada pois permite fazer pesquisas por padrões, introduzir expressões regulares nas *queries*, entre outras opções, tornando assim a pesquisa do GLIMPSE bastante elaborada.

2.1.5 Essence

Tal como o sistema SFS, o Essence [12] pretende extrair informação semântica a partir do conteúdo dos ficheiros, mas no entanto não funciona como um sistema de ficheiros semântico. Ou seja, este sistema tem como principal objectivo extrair as palavras-chave que caracterizam cada ficheiro e com isto criar um índice representativo do conjunto de ficheiros analisados. Pode ser usado principalmente para indexar grandes quantidades de informação não estruturada, como por exemplo um servidor FTP.

Este sistema compreende a estrutura dos documentos (por exemplo manuais de UNIX), ou seja a sua semântica, de forma a conseguir extrair palavras-chave que melhor identifiquem um dado documento. Assim, de forma a atingir este objectivo, a indexação semântica é realizada em duas etapas. A primeira, fase de classificação, analisa cada ficheiro de forma a obter o seu tipo. Na segunda, fase de resumo, é aplicado um *summarizer* específico para cada tipo de ficheiro, de forma a obter todas as palavras que melhor identifiquem o documento analisado.

Como cada *summarizer* entende a semântica do ficheiro que está a analisar, é possível extrair menos palavras-chave, mas com grande significado semântico, e assim obter um índex mais pequeno. Esta é sem dúvida uma das grandes vantagens do Essence.

Uma grande vantagem deste sistema é também o número de tipos de ficheiros suportados. Mas mais importante que o número de ficheiros suportados é o facto de conseguir extrair informação semântica a partir de ficheiros "escondidos", como é o caso de ficheiros comprimidos (por exemplo ficheiros tar). Como o Essence compreende a estrutura deste tipo de ficheiros, cada vez que determina que está na presença num ficheiro deste tipo, extrai os ficheiros "escondidos", determina o tipo de cada um e de seguida aplica os respectivos *summarizers*. Esta funcionalidade é muito importante visto que se pretende que o Essence funcione em ambientes como por exemplo servidores FTP, onde uma grande parte dos ficheiros é deste tipo.

Importante ainda referir que os *summarizers* são simples programas UNIX que são fáceis de escrever e integrar no sistema. Cada *summarizer* está associado a um tipo de ficheiro, compreendendo a sua estrutura e semântica, possibilitando assim a extracção de palavras-chave que melhor caracterizam/resumam o ficheiro. No entanto, apenas são indexados ficheiros para os quais existem *summarizers*. Os restantes ficheiros não são considerados, nem mesmo simples palavras-chave como autor, data de criação, data de modificação (comuns a todo o tipo de ficheiros), são extraídas de tipos de ficheiros "desconhecidos" pelo sistema.

2.1.6 Análise Comparativa dos Sistemas com Semântica Baseada no Conteúdo

Na Tabela 2.1 é apresentado um resumo dos sistemas que se enquadram na categoria "Semântica Baseada no Conteúdo". Aqui, pretendemos evidenciar as principais diferenças (nos aspectos chave) entre os vários sistemas desta categoria.

Para além dos sistemas descritos nas secções acima, existem também outros como Sedar [16], Apple Spotlight [3], Beagle [4], Lifestreams [7], Birch [18] e ConTag [1], que foram considerados na tabela resumo.

Através da análise da tabela podemos verificar que todos os sistemas suportam uma grande variedade de tipos de ficheiros. Para além disso, podemos concluir que grande parte dos sistemas utiliza directorias virtuais, no entanto também existem alguns que usam a pesquisa tradicional e até mesmo interfaces gráficas bastante elaboradas. Por fim podemos verificar, com base nos sistemas para os quais possuímos dados suficientes, que a forma de organização e representação da informação semântica varia bastante de sistema para sistema.

Propriedades Sistemas	**Tipo de ficheiros suportados**	**Utilização de directorias virtuais**	**Representação da informação**
SFS [10]	Quase todos os ficheiros são suportados, apenas é necessário que exista um *transducer* para cada tipo pretendido.	Utiliza directorias virtuais como modo de navegação e pesquisa no sistema semântico.	———
SemFS [19]	Praticamente todos os ficheiros são suportados.	Utiliza directorias virtuais como forma de navegação e procura.	———
Sedar [16]	Praticamente todos os ficheiros são suportados.	Cria "vistas" para apresentar os resultados das pesquisas.	A informação semântica de um ficheiro é guardada num vector semântico.
Nebula [6]	De ficheiros com uma estrutura bem definida conseguem-se obter atributos mais identificativos.	Utiliza "vistas" (género de directorias virtuais) como forma de navegação e pesquisa no sistema.	Para cada ficheiro existe um Nebula *file object*. Este contém todos os atributos do dado ficheiro.
GLIMPSE [17]	Apenas suporta ficheiros de texto.	Não utiliza directorias virtuais, mas tem um sistema de pesquisa bastante elaborado.	Utiliza um pequeno índex que guarda palavras e os respectivos conjuntos de ficheiros onde estas aparecem.
Apple Spotlight [3]	Uma grande quantidade e variedade de tipos de ficheiros são suportados. Para os restantes basta criar um *meta-data importer plug-in*.	Utiliza um sistema de pesquisa próprio.	A cada ficheiro está associado um objecto MDItem. Este contém o conjunto de atributos do dado ficheiro.
Beagle [4]	Praticamente todos os ficheiros são suportados. Além destes, também indexa e-mails e conversações de *instant messenger*.	Utiliza um sistema de pesquisa próprio.	———

Lifestreams [7]	Todos os ficheiros são suportados uma vez que existem sempre atributos como nome, data de criação do ficheiro, data da última actualização...	Fornece uma interface gráfica com a lista de todos os ficheiros, ordenados temporalmente (*lifestreams*). Fornece também um sistema de pesquisa, apresentando os resultados em *substreams* que funcionam como directorias virtuais.	———
Birch [18]	Suporta os mesmos ficheiros que o Apple Spotlight [3], visto que utiliza a sua base de dados (index).	Utiliza *directory-sets* (género de directorias vistuais), como forma de navegação e pesquisa.	Utiliza a informação semântica guardada na base de dados do Apple Spotlight [3].
ConTag [1]	Praticamente todos os ficheiros são suportados.	———	———
Essence [12]	São suportados os tipos de ficheiros para os quais existe um *summarizer*.	———	———

Tabela 2.1: Resumo dos sistemas com informação semântica baseada no conteúdo.

2.2 Semântica Baseada no Contexto

Nesta secção irá ser feita uma análise dos sistemas que extraem informação semântica a partir do contexto de utilização dos ficheiros [2, 15, 27], tentando-se explicar de que forma cada sistema trata as seguintes propriedades e aspectos chave desta categoria:

- Tipo de contexto suportado
- Forma de extrair atributos a partir do contexto
- Forma de representar a informação de contexto

Os sistemas que serão descritos nesta categoria são o Connections [27], o LiFS [2] e o Copernicus [15]

2.2.1 Connections

O Connections [27] é um sistema de ficheiros que combina os atributos extraídos a partir do conteúdo dos ficheiros, com a semântica obtida do contexto onde estes são manipulados. São usadas estas duas formas de obter informação semântica, já que por vezes é difícil extrair, somente a partir do conteúdo, atributos bastante identificativos, como é o caso de músicas ou vídeos. Além disso, analisar apenas o conteúdo nada diz sobre uma forma como os utilizadores pensam para organizar os seus ficheiros, contextualmente.

Podemos considerar vários tipos de contextos de ficheiros. Por exemplo, todos os ficheiros que estão a ser acedidos, ao mesmo tempo que se edita um em particular, as várias tarefas que estão a ser desempenhadas pelo utilizador num determinado instante temporal, ou até mesmo a localização física do utilizador. No caso do Connections, é usada uma forma de contexto, a localidade temporal. Observando a forma como os utilizadores acedem aos ficheiros, é possível obter relações contextuais entre estes. Por exemplo, os ficheiros que são acedidos para leitura, durante um intervalo de tempo, estão relacionados entre si, e um ficheiro que seja escrito, está relacionado com outros que estejam a ser lidos ao mesmo tempo.

Como forma de obter as relações contextuais entre ficheiros, o Connections implementa dois componentes, *tracer* e *relation-graph*. O *tracer* situa-se entre as aplicações e o sistema de ficheiros, registando todas as interacções entre eles. A partir desta monitorização podem ser identificadas as relações entre ficheiros. O segundo componente é responsável por guardar um grafo, com as relações contextuais entre ficheiros. Cada nó do grafo representa um ficheiro e os arcos entre nós simbolizam a relação entre ficheiros. Além disso, os arcos possuem um peso associado, representando a força da relação. De forma a construir o *relation-graph*, este sistema possui um algoritmo, cuja função é analisar a informação proveniente do *tracer* e, a partir desta, identificar os ficheiros que foram acedidos durante uma determinada janela temporal, construindo por fim o *relation-graph*. Para uma melhor percepção, na Figura 2.1 é apresentado um *relation-graph* simples, onde as letras representam ficheiros, os arcos as relações entre estes e os pesos no arcos a força da relação.

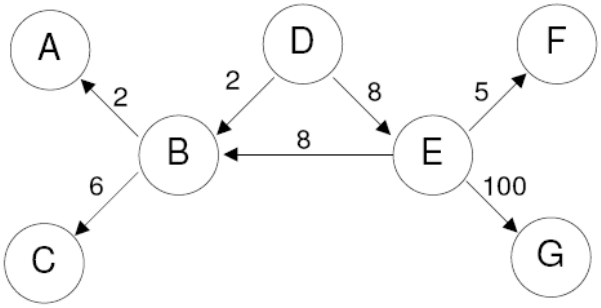

Figura 2.1: Exemplo de um *relation-graph*, retirado de [27].

No Connections, quando um utilizador efectua uma pesquisa, o sistema, internamente, faz uma procura somente com base no conteúdo, obtendo um conjunto de resultados. De seguida, o *relation-graph* é analisado com base nesse conjunto, sendo obtidos mais alguns ficheiros provenientes de relações contextuais. Por fim é devolvido ao utilizador o conjunto total de resultados, que foi obtido com base em informação semântica de conteúdo e de contexto.

2.2.2 LiFS

O LiFS [2] caracteriza-se pelo facto de suportar, para além de atributos dos ficheiros, *links* (relações) entre ficheiros, sendo que estes *links* também têm atributos associados. Tanto os atributos dos ficheiros, como os dos *links*, podem ser extraídos automaticamente ou então introduzidos pelo utilizador.

Neste sistema cada *link* possui como propriedades, um ficheiro fonte, um ficheiro alvo e um conjunto de atributos. Por esta razão, estes *links* não são meras referências entre ficheiros, representam sim relações entre eles, assim como o tipo de relação (devido aos atributos).

A criação de *links* pode ser feita tanto automaticamente como manualmente. Por exemplo, quando se abre um ficheiro, é criado automaticamente um *link* entre este e a directoria onde o utilizador se encontra. Para além disso, o sistema dá a possibilidade de introduzir novos *links*, apenas é necessário especificar o

ficheiro fonte, o ficheiro alvo e um ou mais atributos dessa relação.

Com a introdução deste conceito de *links* entre ficheiros, torna-se possível efectuar uma pesquisa neste sistema e obter como resultados, não só ficheiros, cujos atributos correspondem aos critérios de procura, como ficheiros que se relacionem com os do primeiro conjunto de resultados.

2.2.3 Copernicus

Tal como os dois sistemas descritos acima, o Copernicus [15] pretende capturar não só as propriedades dos ficheiros mas também as relações existentes entre eles. Deste modo, a informação recolhida consegue aumentar a usabilidade do sistema, na medida em que melhora a facilidade na pesquisa de ficheiros.

No Copernicus são consideradas duas formas de relações entre ficheiros, *provenance* e localidade temporal. Na primeira podemos por exemplo inferir relações do tipo: "Os ficheiros acedidos por um determinado *script* estão relacionados com esse script". Já na segunda são analisados os padrões de acesso aos ficheiros, podendo ser inferidas relações do tipo: "Dois ficheiros que sejam acedidos durante a mesma janela temporal estão relacionados entre si". Para além de inferir relações entre ficheiros, o Copernicus também permite que estas sejam atribuídas explicitamente pelos utilizadores ou até mesmo por aplicações. De notar que qualquer relação é sempre representada da seguinte forma: <*relationship type*, *source file*, *target file*>.

De forma a aumentar o desempenho das pesquisas, este sistema guarda toda a informação semântica recolhida num grafo. Este grafo tem a particularidade de colocar em *clusters*, os ficheiros que são semanticamente semelhantes e que costumam ser acedidos em conjunto, como se pode verificar na Figura 2.2.

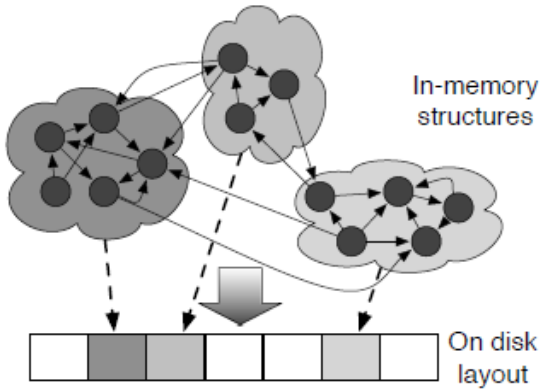

Figura 2.2: Esquema de um grafo guardado pelo sistema, retirado de [15].

Analisando a figura acima podemos verificar que os vértices do grafo representam os ficheiros, os arcos entre vértices assinalam as relações entre ficheiros e os grupos de vértices são os *clusters* de ficheiros. Podemos ainda verificar que os ficheiros pertencentes ao mesmo *cluster* estão guardados em disco de forma contígua, o que aumenta o desempenho no acesso aos ficheiros. Importante ainda referir que as políticas para a construção de *clusters* podem ser por exemplo: ficheiros pertencentes ao mesmo projecto, ficheiros com atributos partilhados e ficheiros com os mesmos padrões de acesso. No Copernicus cada ficheiro apenas pode pertencer a um *cluster*, já que manter o mesmo ficheiro em vários *clusters* tornaria a sincronização bastante difícil.

Esta forma, em grafo, de organização da informação semântica é uma grande vantagem do sistema, visto que facilita a pesquisa de ficheiros, possibilitando uma navegação semântica mais eficaz.

2.2.4 Análise Comparativa dos Sistemas com Semântica Baseada no Contexto

A Tabela 2.2 apresenta um resumo dos sistemas com semântica baseada no contexto. Com esta tabela, pretendemos evidenciar o modo como cada sistema trata os principais aspectos chave desta categoria de sistemas. Importante ainda referir que, para além dos sistemas descritos acima, também se incluem nesta tabela os sistemas Oxygen [23] e pStore [29].

Propriedades Sistemas	**Tipo de contexto suportado**	**Forma de extrair atributos a partir do contexto**	**Forma de representar a informação de contexto**
Connections [27]	É utilizada a localidade temporal. Os ficheiros acedidos durante um certo intervalo de tempo estão relacionados entre si.	Existe um módulo, *tracer*, responsável por monitorizar as interacções entre as aplicações e o sistema de ficheiros. A partir desta análise é possível identificar as relações entre ficheiros.	É guardado um grafo com as relações contextuais entre ficheiros. Os nós representam os ficheiros e os arcos as relações entre estes.
LiFS [2]	Existem *links* entre ficheiros. Estes, descrevem relações entre os ficheiros.	———	———
Oxygen [23]	Forma como o utilizador interage com os objectos do sistema.	Analisando o modo como o utilizador acede aos dados, é possível identificar relações entre objectos.	As relações entre objectos são representadas por arcos, que ligam os objectos.
pStore [29]	Padrões de acesso aos ficheiros e relações explícitas entre ficheiros (como por exemplo relações de dependência).	Analisando os padrões de acesso aos ficheiros é possível obter relações entre estes. É ainda possível colocar relações explícitas entre ficheiros.	———
Copernicus [15]	*Provenance* e localidade temporal. Também é possível que o utilizador e aplicações coloquem relações explícitas entre ficheiros.	Observando o fluxo de acesso aos ficheiros é possível inferir relações entre estes. Também podem ser colocadas relações explícitas entre ficheiros.	É guardado um grafo que representa as relações entre ficheiros. Os vértices são os ficheiros e os arcos representam as relações entre estes.

Tabela 2.2: Resumo dos sistemas com semântica baseada no contexto.

Através da análise da tabela podemos concluir que existem essencialmente duas formas de relacionar ficheiros, ou inferindo relações a partir dos padrões de acesso aos ficheiros, ou então colocando explici-

tamente relações entre eles. Podemos também verificar, que para extrair informação semântica a partir do contexto, os sistemas têm de monitorizar os acessos aos ficheiros e a partir daí inferir relações entre eles. Por fim, podemos também verificar que nos sistemas Connections, Oxygen e Copernicus é sempre utilizado um grafo como forma de representar a informação de contexto.

2.3 Semântica Baseada nas *Tags* Atribuídas

Nesta secção irá ser feita uma descrição dos sistemas que utilizam as *Tags* atribuídas pelo utilizador como fonte de informação semântica, realçando o modo como cada sistema trata os seguintes aspectos chave desta categoria:

- Organização das *Tags*
- Sistema de pesquisa utilizado

Nesta categoria incluem-se sistemas como o TagFS [5] e o Tagsistant [28], os quais serão descritos nesta secção.

2.3.1 TagFS

Ao contrário dos sistemas descritos até este ponto, o TagFS [5] não pretende extrair atributos a partir das propriedades, conteúdo ou contexto dos ficheiros. Em vez disso, dá aos utilizadores, a possibilidade de atribuir *Tags* (simples palavras ou anotações) aos ficheiros. Ou seja, os utilizadores são responsáveis por atribuir e gerir os atributos dos seus dados.

No TagFS um ficheiro pode conter quantas *Tags* o utilizador desejar. Deste modo, ao contrário do que acontece nos sistemas hierárquicos, passam a existir várias formas de encontrar um dado ficheiro. Neste sistema, os caminhos para directorias são interpretados como *queries*, sendo retornados todos os ficheiros marcados com as *Tags* presentes numa dada *query*. Por exemplo, considere-se o caso em que um ficheiro MP3 está marcado com as *Tags* "U2", "2007" e "favoritos". Esta música, tanto pode ser encontrada ao pesquisar por /U2/favoritos, como por /favoritos/U2. No TagFS estas duas formas são equivalentes. A primeira significa fazer uma procura pelos ficheiros marcados com "U2" e desses resultados procurar aqueles que estão marcados com "favoritos". A segunda forma significa pesquisar por todos os ficheiros marcados com "favoritos", e desses encontrar os que estão marcados com "U2". Verifica-se assim que o conjunto de resultados será o mesmo. Na Figura 2.3 está apresentado um exemplo simples de quais são os conjuntos de ficheiros que podem ser retornados, consoante os termos introduzidos para pesquisa. Podemos assim afirmar que a lista de resultados de uma pesquisa, depende dos resultados relativos a cada *Tag*.

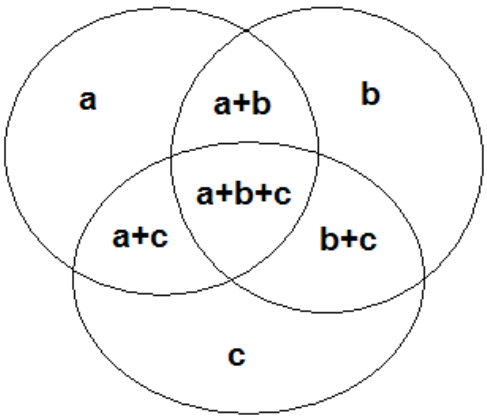

Figura 2.3: Exemplo simples, apenas com três *Tags*, que mostra os diferentes conjuntos de ficheiros que podem ser retornados, consoante a pesquisa.

Neste sistema é possível que ao fazer uma pesquisa sejam apresentados, além de um conjunto de ficheiros, subpastas, com os nomes de outras *Tags*, permitindo assim refinar a pesquisa. Para o seguinte exemplo considere-se o ficheiro MP3 descrito acima. Quando se pesquisa por /U2/favoritos pode-se obter uma lista exaustiva de ficheiros, sendo algo difícil encontrar o que se pretende no meio de tantos outros. Por essa razão, o TagFS inclui na lista de resultados, *Tags* que estão presentes nos ficheiros de resultado, mas que não foram incluídas na pesquisa. No caso do exemplo, seria apresentada uma subpasta com o nome `2007`, de forma a permitir refinar a procura. É importante referir ainda que o facto de uma *Tag* ser apresentada como resultado de uma *query*, não significa que exista uma relação hierárquica entre *Tags*. Esta é apenas uma forma de possibilitar o refinamento da pesquisa.

2.3.2 Tagsistant

Tal como o sistema descrito anteriormente, o Tagsistant [28] tem como principais objectivos, permitir que o utilizador coloque *Tags* nos ficheiros, e que possa efectuar pesquisas usando esta informação semântica. Do mesmo modo que o anterior, este sistema não possui mecanismos para extrair automaticamente atributos dos ficheiros. Os únicos atributos presentes são as *Tags*.

O Tagsistant possibilita, como forma de aumentar a usabilidade, que as pesquisas possam usar os operadores lógicos AND e OR. Permite por exemplo fazer uma pesquisa do tipo:

```
$ ls tags/U2/AND/2007/OR/ColdPlay/AND/2005/
```

e obter como resultado, todos os ficheiros com as *Tags* “U2” e “2007” juntamente com todos os ficheiros marcados com “ColdPlay” e “2005”.

De forma a guardar toda a informação relativa a *Tags*, é usada uma base de dados. Nessa base de dados estão guardadas todas as *Tags* existentes, todas as relações entre estas e os ficheiros, as regras semânticas e para além disso são também guardados resultados de pesquisas anteriores. Guardar estes resultados é uma mais-valia para o sistema, aumentando o seu desempenho. Quando são guardados, têm um determinado tempo de vida associado e são actualizados dinamicamente se se verificarem alterações nas relações entre *Tags* e ficheiros.

Para além de permitir associar *Tags* a ficheiros, o Tagsistant também permite especificar relações entre estas, sendo mantidas na base de dados. Estas relações podem ser de dois tipos, inclusão ou equivalência. Uma relação de inclusão entre duas *Tags* significa que existe um género de organização hierárquica, na medida em que uma *Tag* pode incluir várias. Por exemplo, “musica” inclui as *Tags* “rock” e “pop”. Assim, uma pesquisa por ficheiros marcados com “musica” vai retornar também todos os que estejam marcados com “rock” e “pop”. Já uma relação de equivalência indica que uma determinada *Tag* tem o

mesmo significado que outra. Por exemplo, "Férias" é equivalente a "ferias". Neste caso, uma pesquisa por "Férias" retorna todos os ficheiros marcados com "Férias" e todos os marcados com "ferias". O mesmo aconteceria se se pesquisasse por "ferias".

2.3.3 Análise Comparativa dos Sistemas com Semântica Baseada nas *Tags* Atribuídas

Na Tabela 2.3 é apresentado um resumo dos sistemas com semântica baseada nas *Tags* atribuídas pelo utilizador. Aqui, pretendemos evidenciar as principais diferenças (nos aspectos chave) entre os vários sistemas desta categoria.

Analisando a tabela, podemos verificar que enquanto no TagFS as *Tags* são apenas simples palavras atribuídas aos ficheiros, não existindo qualquer relação entre elas, no Tagsistant as *Tags* podem estar relacionadas entre si. Estas relações podem ser de "inclusão" ou de "equivalência", sendo também colocadas pelo utilizador. Podemos ainda verificar que no sistema de pesquisa fornecido pelo TagFS, são devolvidos resultados com o nome de outras *Tags*, permitindo assim refinar a pesquisa. Já no Tagsistant, o sistema de pesquisa dá a possibilidade de serem utilizados os operadores lógicos AND e OR.

Propriedade Sistemas	**Organização das *Tags***	**Sistema de pesquisa utilizado**
TagFS [5]	Não existem relações entre *Tags*. Estas são etiquetas independentes que se atribuem aos ficheiros.	Os "caminhos" para directorias são interpretados como *querys*. Por exemplo, /U2/favoritos retorna todos os ficheiros marcados com as *Tags* "U2" e "favoritos". Ao efectuar uma pesquisa, para além dos ficheiros de resultado, também podem ser apresentadas outras *Tags* que permitem refinar a pesquisa.
Tagsistant [28]	As *Tags* podem estar relacionadas entre si. Estas relações são introduzidas pelo utilizador e podem ser de inclusão, em que uma *Tag* inclui outras, ou então de equivalência, onde uma *Tag* tem o mesmo significado de outra.	Permite que sejam utilizados, na pesquisa, operadores lógicos AND e OR, de forma a aumentar a usabilidade do sistema.

Tabela 2.3: Resumo dos sistemas com semântica baseada nas *Tags* atribuídas.

2.4 Sistemas Híbridos

Nesta secção irá ser feita uma descrição dos sistemas apresentados em [11, 14, 24], procurando-se analisar o modo como cada sistema trata as seguintes propriedades e aspectos chave identificados para esta categoria de sistemas:

- Representação e organização da informação
- Interface de pesquisa

2.4.1 DBFS

Tal como os sistemas descritos neste capítulo, o DBFS [11] pretende criar uma nova forma de interacção com os ficheiros guardados no nosso computador. No entanto, este sistema deu uma grande importância à nova interface gráfica fornecida ao utilizador. Esta tem o objectivo de ser o mais intuitiva possível, de forma a aumentar a usabilidade do novo sistema de ficheiros.

O DBFS não restringe que os ficheiros sejam guardados de forma hierárquica, em vez disso, guarda todos os documentos num grande repositório de dados, ou base de dados. De notar que podem ser guardados diversos ficheiros com os mesmos metadados, não existindo por isso conflitos de nomes. Na realidade, não são guardados ficheiros directamente na base de dados, mas sim mantidas referências para os documentos guardados no sistema hierárquico. Desta forma mantém-se a compatibilidade com o sistema de ficheiros já existente.

Este sistema, para além extrair informação semântica a partir dos metadados dos ficheiros, como por exemplo nome, tipo, tamanho e data de modificação, também permite que o utilizador atribua propriedades (*Tags*) aos ficheiros. Desta forma, é possível fazer pesquisas não só pelos atributos dos ficheiros, mas também pelas *Tags* atribuídas manualmente, aumentando a usabilidade do sistema.

Como já referido anteriormente, foi dada grande importância à interface gráfica do DBFS. Esta é o mais intuitiva possível, possibilitando que o utilizador possa facilmente navegar pelos seus ficheiros, somente através das suas propriedades semânticas.

2.4.2 Insight

O sistema Insight [14] é considerado híbrido uma vez que extrai automaticamente informação semântica a partir dos metadados ("Semântica Baseada no Conteúdo") e para além disso permite que o utilizador coloque *Tags* nos ficheiros ("Semântica Baseada nas *Tags* Atribuídas"). Assim, através da utilização destas duas fontes de informação semântica, o sistema consegue aumentar o desempenho e flexibilidade das pesquisas.

Neste sistema não se pretende que a informação semântica seja apenas formada por simples palavras-chave associadas aos ficheiros. Esta também deve poder ser estruturada de forma a existirem pares "chave-valor", como por exemplo:

- *autor* = "André Mendes"
- *universidade.ano* = 5, *fotografia.pessoa* = "André Mendes"
- *universidade.ano.5*, *fotografia.pessoa.*"*André Mendes*"

Com a utilização desta informação estruturada, é possível realizar pesquisas mais rápidas e tornar a navegação, por atributos, mais eficiente. Assim, esta característica pode ser considerada um dos pontos fortes do sistema Insight.

À semelhança do SFS, descrito na Secção 2.1.1, este sistema utiliza directorias virtuais como forma de navegação semântica através dos ficheiros. Deste modo, podemos por exemplo aceder a `/insight/ferias` sendo apresentados todos os ficheiros marcados com a palavra "ferias". Podemos também por exemplo pesquisar os documentos cujo autor é "andre", bastando aceder a `/insight/autor.andre`.

2.4.3 SemDAV

Ao contrário dos dois sistemas descritos acima, o SemDAV [24] extrai informação semântica a partir do contexto dos ficheiros. Ou seja, para além de obter atributos a partir dos metadados dos ficheiros e das *Tags* atribuídas manualmente, também consegue inferir relações entre ficheiros. Assim, ao reunir

um vasto número de informação semântica, está a permitir que um utilizador encontre mais facilmente o ficheiro pretendido.

Este sistema substitui os nomes dos ficheiros, que são formados seguindo uma estrutura hierárquica, atribuindo-lhes identificadores únicos. Para além disso, associa um conjunto de atributos aos documentos, para que estes possam ser facilmente encontrados. Deste modo, os ficheiros passam a poder ser guardados sem preocupações com a estrutura hierárquica, existindo em vez disso, um conjunto de atributos que identificam cada um.

Nos sistemas de ficheiros hierárquicos cada ficheiro é tratado como um objecto isolado, não existindo relações entre os mesmos. Assim, é impossível criar relações entre ficheiros semanticamente semelhantes, mas que estão guardados em sítios distintos numa hierarquia. Como forma de resolver este problema, o sistema SemDAV analisa o contexto dos ficheiros, conseguindo inferir relações entre os mesmos, o que permite aumentar o desempenho da navegação semântica.

2.4.4 Análise Comparativa dos Sistemas Híbridos

Na Tabela 2.4 está apresentado um resumo dos sistemas híbridos. Aqui, pretende-se evidenciar o modo como cada sistema trata os aspectos chave desta categoria de sistemas.

Analisando a tabela, podemos verificar que cada um dos três sistemas desta categoria, tem a sua própria forma de guardar a informação semântica recolhida. Ou seja, cada sistema guarda a informação da forma que melhor se adapte à sua lógica e arquitectura. Podemos ainda verificar que, a interface fornecida para pesquisa difere de sistema para sistema. Enquanto o DBFS fornece uma interface gráfica que permite navegar pelos ficheiros de forma intuitiva, o Insight utiliza directorias virtuais como forma de navegação e pesquisa.

Propriedades Sistemas	**Representação e organização da informação**	**Interface de pesquisa**
DBFS [11]	As referências para os ficheiros que se encontram no sistema hierárquico são guardadas numa base de dados. Para além disso, são também guardadas as palavras-chave associadas a cada ficheiro.	Possui uma interface gráfica que permite navegar pelos ficheiros de forma intuitiva.
Insight [14]	A informação semântica deve estar estruturada de forma a existirem pares "chave-valor". Assim, é possível aumentar o desempenho das pesquisas e da navegação semântica.	Utiliza directorias virtuais como forma de navegação pelo sistema de ficheiros.
SemDAV [24]	Como informação semântica, guarda os atributos extraídos automaticamente a partir dos metadados, as relações inferidas do contexto dos ficheiros e as *Tags* colocadas manualmente.	———

Tabela 2.4: Resumo dos sistemas híbridos.

2.5 Síntese

Neste capítulo apresentámos o estado da arte na área dos sistemas de ficheiros semânticos. Os sistemas desenvolvidos nesta área podem ser divididos em três categorias principais, tendo por base a fonte da informação semântica.

A primeira categoria engloba os sistemas que extraem informação semântica a partir do conteúdo dos ficheiros. Ou seja, este tipo de sistemas obtêm informação semântica de atributos como autor, data de criação, nome, tamanho, tipo e até mesmo de outras informações presentes nos metadados dos ficheiros. É nesta categoria que se inserem a grande maioria dos sistemas desenvolvidos na área dos sistemas de ficheiros semânticos, no entanto, têm a desvantagem de não captarem o contexto de utilização dos ficheiros, nem permitirem a atribuição de *Tags* por parte do próprio utilizador.

De forma a colmatar uma das falhas dos sistemas com semântica baseada no conteúdo, foram desenvolvidos trabalhos que se inserem na categoria "Semântica Baseada no Contexto". Estes sistemas, para além de obterem atributos do conteúdo, também recorrem ao contexto de utilização dos ficheiros como fonte de informação semântica. O contexto de um ficheiro pode ser, por exemplo, outros ficheiros que sejam acedidos concorrentemente com este, ou mesmo tarefas que o utilizador desempenhe em simultâneo com o acesso ao ficheiro. No fundo, é uma forma de inferir relações entre ficheiros, o que permite aumentar a usabilidade geral do sistema. Apesar das vantagens introduzidas, este tipo de sistemas possui uma maior complexidade no levantamento e gestão das *Tags* obtidas a partir do contexto.

A categoria "Semântica Baseada nas Tags Atribuídas" engloba todos os sistemas que utilizam como informação semântica, as *Tags* introduzidas pelo utilizador. As *Tags* são, neste sentido, simples palavras-chave associadas a um ou mais ficheiros. Contudo, apesar de possibilitarem atribuir manualmente *Tags* aos ficheiros, este tipo de sistemas não consegue extrair automaticamente atributos a partir das propriedades, conteúdo ou contexto dos ficheiros.

De forma a completar as faltas de cada categoria, existem os sistemas híbridos, ou seja, trabalhos que possuem características das três categorias de sistemas de ficheiros semânticos. O OSMOSIS, pode assim ser considerado um sistema híbrido, visto que extrai automaticamente *Tags* a partir dos atributos e conteúdo dos ficheiros ("Semântica Baseada no Conteúdo"), permite colocar (manualmente) relações entre ficheiros ("Semântica Baseada no Contexto") e permite atribuir manualmente *Tags* a ficheiros ou directorias ("Semântica Baseada nas Tags Atribuídas").

Capítulo 3

Arquitectura

A arquitectura do sistema OSMOSIS pode ser dividida em duas grandes componentes, que correspondem aos dois principais objectivos do trabalho. Neste capítulo começa-se por apresentar um exemplo com a vista global do sistema OSMOSIS, depois um esquema onde se mostra o que acontece quando o PDA lê uma *Tag* RFID e uma figura que mostra a divisão entre os dois componentes (OSMOSIS-SFS e OSMOSIS-RFID). Por fim, será apresentada a arquitectura do sistema de ficheiros semântico desenvolvido para funcionar no *Desktop* (OSMOSIS-SFS) e a arquitectura do sistema do PDA (OSMOSIS-RFID). No próximo capítulo serão analisados em pormenor todos os módulos que constituem o sistema, assim como os detalhes de implementação de cada um.

3.1 Vista Global do Sistema OSMOSIS

Na Figura 3.1 é apresentado um exemplo de como o sistema OSMOSIS pode ser integrado com o mundo real.

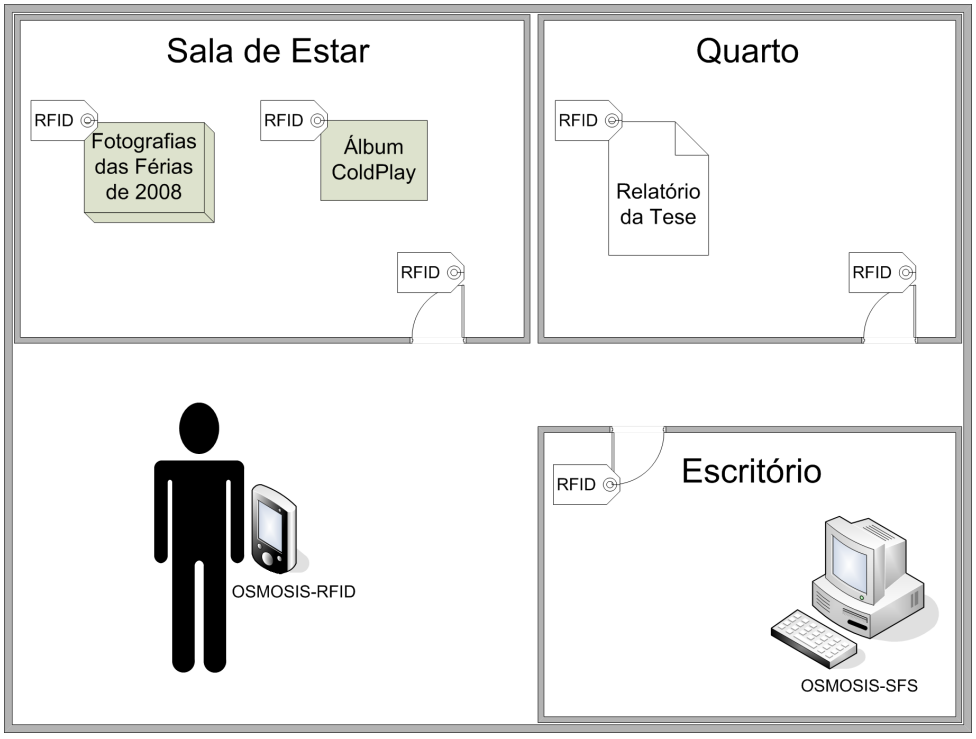

Figura 3.1: Exemplo de aplicação do sistema OSMOSIS.

Na figura acima está representada uma casa com três divisões: sala de estar, quarto e escritório. Podemos verificar que na sala de estar se encontram dois objectos marcados com *Tags* RFID, uma caixa de fotografias das férias de 2008 e o cd de ColdPlay. Estes são objectos físicos que têm relação directa com ficheiros no servidor (OSMOSIS-SFS), e que no fundo que existem no mundo virtual e no mundo físico. Da mesma forma, no quarto encontra-se este relatório da tese em formato físico (marcado com uma *Tag* RFID) e que está relacionado com o ficheiro "tese.pdf" existente no servidor.

Neste exemplo podemos ainda verificar que no escritório se encontra o PC (servidor) onde está a correr o sistema OSMOSIS-SFS. É neste PC que está guardada toda a informação semântica dos ficheiros, assim como as relações entre estes e o mundo físico. Note-se ainda que todas as divisões da casa estão marcadas com uma *Tag* RFID e que o utilizador possui um PDA, equipado com um leitor RFID, onde está a correr o sistema OSMOSIS-RFID.

3.2 Interacção típica entre o OSMOSIS-RFID e o OSMOSIS-SFS

Na Figura 3.2 está representada uma situação simples em que o OSMOSIS-RFID lê uma *Tag* RFID interagindo de seguida com o OSMOSIS-SFS. Para uma melhor percepção, a seguir à figura estão descritas as quatro fases deste caso de uso.

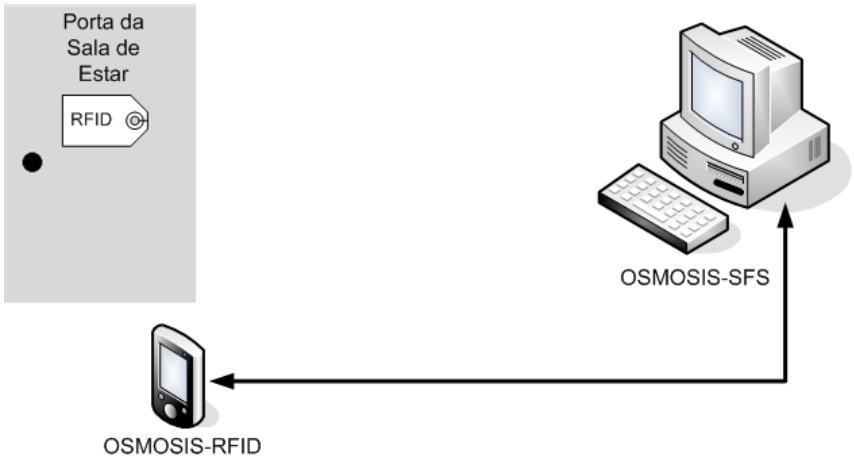

Figura 3.2: Exemplo de uma interacção típica entre o sistema OSMOSIS e o mundo real.

Quando um utilizador entra numa sala de estar, com o seu PDA, lê a *Tag* RFID referente a essa sala e de seguida:

1. O PDA comunica com o Servidor, informando-o que leu a *Tag* RFID "Sala de Estar".

2. De seguida o Servidor acede à base de dados, onde está guardada a informação semântica e verifica que na Sala de Estar se encontra a caixa de fotografias das férias de 2008 e o álbum físico de ColdPlay. Confirma ainda que estes objectos físicos estão relacionados, respectivamente, com uma directoria denominada `Fotografias_Ferias_2008` (que contém as fotografias em formato JPEG) e uma outra denominada `Album_ColdPlay` (que contém as músicas em formato MP3).

3. O Servidor envia esta informação ao PDA e neste abre-se uma vista denominada "Sala de Estar". No seu interior encontram-se duas pastas virtuais: `Fotografias_Ferias_2008` e `Album_ColdPlay`, que o utilizador pode explorar. Estas pastas virtuais são uma representação das directorias com o mesmo nome que existem no servidor.

4. Se o utilizador do PDA pretender aceder a um ficheiro, o PDA comunica com o Servidor para que este lhe envie o ficheiro e quando for recebido o utilizador poderá aceder-lhe normalmente.

3.3 Arquitectura Global do Sistema OSMOSIS

Na Figura 3.3 está representada a arquitectura global do sistema OSMOSIS. Como podemos verificar, o sistema está dividido em dois grandes componentes: o OSMOSIS-SFS (Servidor) e o OSMOSIS-RFID (PDA). Estes implementam os módulos `Interface com o PDA` e `Interface com o Servidor`, respectivamente, de forma a comunicarem entre si através de uma interface bem definida. No entanto, todos os restantes módulos do OSMOSIS-SFS não são conhecidos pelo OSMOSIS-RFID e vice-versa.

Importante ainda referir que o sistema OSMOSIS-SFS pode funcionar, como um sistema de ficheiros semântico, sem que exista ligação ao OSMOSIS-RFID. Este último, no entanto, necessita do OSMOSIS-SFS para poder funcionar, visto que toda a informação semântica se encontra do lado do servidor. Assim, só consegue fornecer todas as funcionalidades se existir comunicação com o servidor.

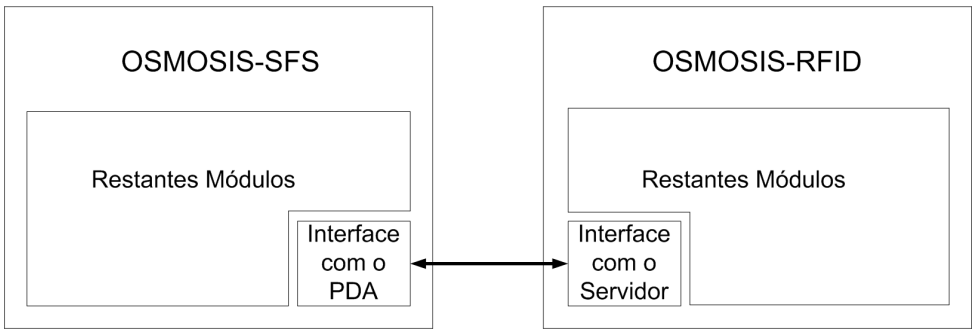

Figura 3.3: Arquitectura global do sistema OSMOSIS.

3.4 Arquitectura do OSMOSIS-SFS

Na Figura 3.4 está apresentada a vista em módulos, da arquitectura do Sistema de Ficheiros Semântico desenvolvido. De seguida, serão explicadas as responsabilidades de cada módulo.

Como podemos verificar, a arquitectura do OSMOSIS-SFS é composta pelos seguintes módulos:

- `Aplicações`
- `Tracer`
- `Sistema de Ficheiros`
- `Análise do Conteúdo`
- `Tags Introduzidas pelo Utilizador`
- `Base de Dados`
- `Motor de Pesquisa Semântico`
- `Interface com o Utilizador`
- `Interface com o PDA`

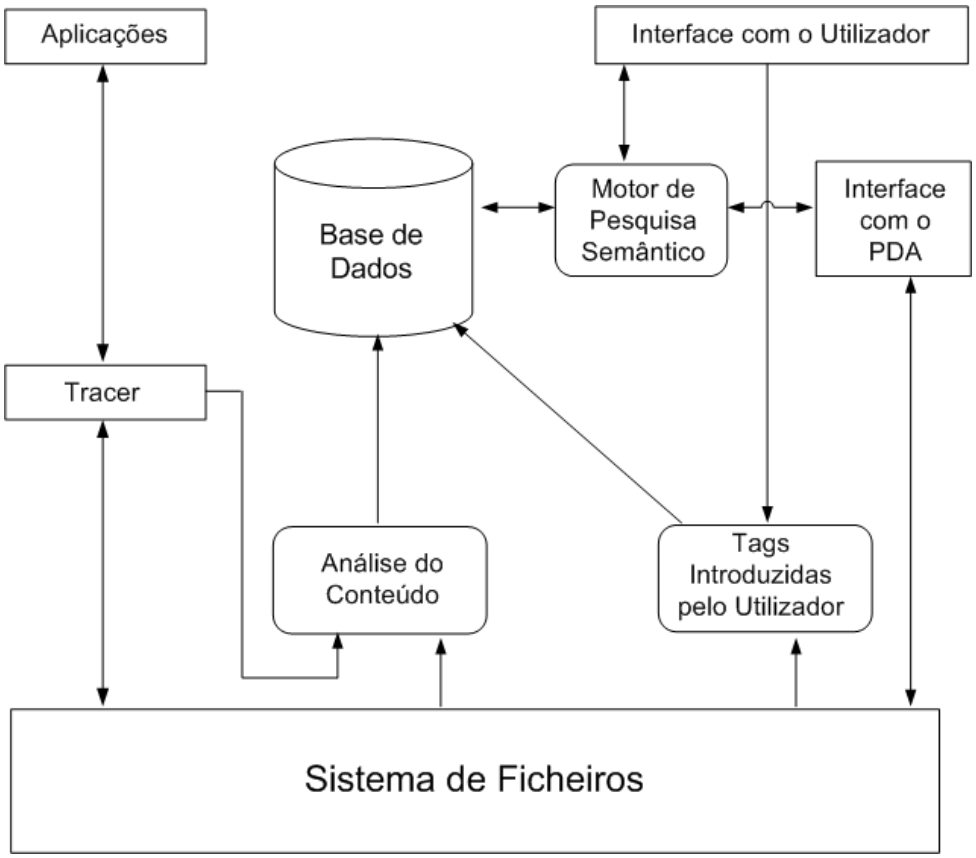

Figura 3.4: Vista em módulos da arquitectura do OSMOSIS-SFS.

As `Aplicações` são todos os programas existentes no PC que podem aceder e modificar os ficheiros. Estas, não sendo propriamente um módulo do nosso sistema, são aqui representadas para que se perceba o fluxo de execução gerado pela modificação dos ficheiros. Como se pode verificar, o módulo `Tracer` encontra-se entre as aplicações e o sistema de ficheiros nativo do Windows, conseguindo assim monitorizar todos os acessos e modificações no sistema de ficheiros.

Como já referido, o `Tracer` monitoriza todos os acessos ao sistema de ficheiros e deste modo, sempre que uma aplicação criar, modificar, renomear ou apagar ficheiros e directorias, o Tracer regista essa acção. Para além de registar os acessos, o `Tracer` é também responsável por alertar o módulo `Análise do Conteúdo` acerca do ficheiro ou directoria alterada e correspondente tipo de alteração (criar, modificar, renomear ou apagar). Desta forma, conseguimos garantir que os dados existentes na BD estão sempre consistentes com o sistema de ficheiros.

O `Sistema de Ficheiros` referenciado na figura não é um novo sistema de ficheiros, mas sim o que existe nativamente no Microsoft Windows. Mais uma vez, este não é um módulo do OSMOSIS-SFS, apenas se encontra representado na figura para que se perceba que o sistema de ficheiros continua a poder ser utilizado normalmente, sem interferência da nossa aplicação. Ou seja, no fundo um utilizador pode continuar a interagir com o sistema de ficheiros da mesma forma que fazia antes de instalar o OSMOSIS-SFS, este apenas fornece funcionalidades (de sistemas de ficheiros semânticos) adicionais.

Toda a funcionalidade do módulo `Análise do Conteúdo` tem por base as informações que lhe são passadas através do `Tracer`. Internamente, a sua estrutura pode ser dividida em três partes com funcionalidades distintas. A primeira é responsável por tratar os dados provenientes do `Tracer` e tomar acções com base nessas informações. A segunda é responsável por extrair *Tags* a partir dos metadados dos ficheiros e do conteúdo de ficheiros de texto. Por último, existe uma parte responsável por fazer a actualização da base de dados.

O módulo `Tags Introduzidas pelo Utilizador` é responsável por receber e executar pedidos do utilizador para adicionar *Tags* a um ficheiro ou directoria. Para esta funcionalidade optámos por acrescentar a opção "Adicionar Tag" ao *Context Menu* dos ficheiros e directorias, tornando a atribuição manual de *Tags* o mais intuitiva possível.

A `Base de Dados` funciona como repositório de toda a informação semântica necessária ao correcto funcionamento do sistema OSMOSIS. Esta base de dados guarda para cada ficheiro as suas *Tags*, guarda informação sobre os ficheiros do OSMOSIS-SFS que estão relacionados com objectos reais (marcados com *Tags* RFID), relações entre as localizações físicas e os ficheiros do OSMOSIS-SFS e guarda ainda todas as localizações físicas existentes no sistema assim como o correspondente número codificado na *Tag* RFID.

Para que a informação semântica recolhida possa ser útil, é necessário fornecer um sistema de pesquisa ao utilizador. Esta funcionalidade é assim fornecida através do módulo `Motor de Pesquisa Semântico`, sendo aqui que está implementado todo o algoritmo de pesquisa do sistema OSMOSIS. Os pedidos de pesquisa podem ser realizados através da `Interface com o Utilizador`, no caso de pesquisas introduzidas pelo utilizador, ou então através da `Interface com o PDA`, no caso de pesquisas realizadas no PDA.

O módulo `Interface com o Utilizador` tem como principal função receber os pedidos submetidos pelo utilizador e apresentar resultados (por exemplo de pesquisas). Actualmente esta interface fornece funcionalidades como pesquisa semântica, visualizar as *Tags* associadas a um determinado ficheiro ou directoria, associar explicitamente dois ficheiros, adicionar localizações físicas ao sistema e relacionar ficheiros com localizações físicas.

Por último, o módulo `Interface com o PDA` tem como função receber e responder aos vários pedidos vindos do OSMOSIS-RFID. No fundo, este módulo funciona como o servidor porque é aqui que o PDA estabelece ligação e onde são realizados todos os pedidos. De referir ainda que o OSMOSIS-SFS pode funcionar correctamente, somente como um sistema de ficheiros semântico, sem o uso deste módulo.

3.5 Arquitectura do OSMOSIS-RFID

A arquitectura do OSMOSIS-RFID é bastante mais simples que a do OSMOSIS-SFS, devido ao facto de não extrair atributos automaticamente dos ficheiros, de não possibilitar a introdução manual de *Tags*, de não guardar informação semântica e de não possuir um mecanismo de pesquisa próprio. De referir ainda que todo o processamento computacional é realizado do lado do servidor (OSMOSIS-SFS), sendo que o OSMOSIS-RFID apenas trata da interacção com o utilizador e apresenta os resultados devolvidos pelo servidor.

Na Figura 3.5 está apresentada a arquitectura do OSMOSIS-RFID e, de seguida, encontra-se uma descrição de cada módulo.

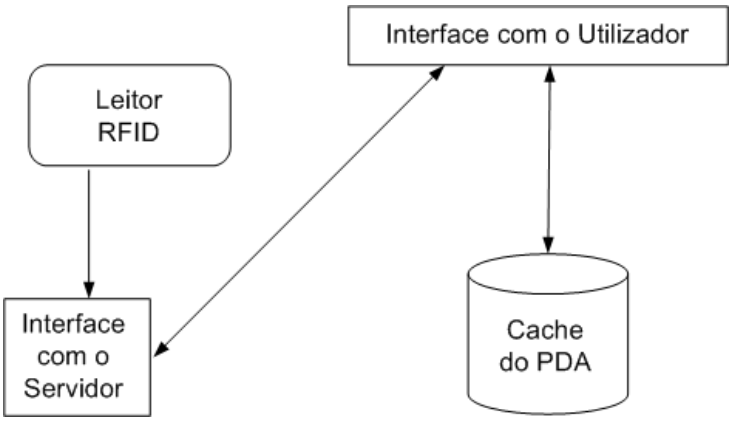

Figura 3.5: Vista em módulos da arquitectura do OSMOSIS-RFID.

Como podemos verificar na figura 3.5, a arquitectura do OSMOSIS-RFID é composta pelos seguintes módulos:

- `Leitor RFID`
- `Cache`
- `Interface com o Utilizador`
- `Interface com o Servidor`

O `Leitor RFID` é o módulo responsável por fazer a leitura de *Tags* RFID, e sempre que isso aconteça, deverá informar o Servidor (onde está instalado o OSMOSIS-SFS). Por cada *Tag* RFID lida, este módulo começa por extrair o número da *Tag* (TagUID) e seguidamente envia um pedido ao servidor, de forma a obter a localização e os ficheiros associados a essa *Tag*. Note-se que este pedido não é efectuado directamente, mas sim através do módulo `Interface com o Servidor`.

De forma a aumentar o desempenho do sistema OSMOSIS-RFID, tornou-se evidente a necessidade de guardar resultados de pesquisas anteriores, de forma a evitar estar constantemente a fazer pedidos ao servidor (OSMOSIS-SFS). Por esta razão, foi criada uma `Cache` no PDA, que no fundo funciona como um repositório onde são guardados resultados de pesquisas anteriores, e que poderão vir a ser utilizados em pesquisas futuras.

A `Interface com o Utilizador` existente no PDA, tem como principal função receber os pedidos submetidos pelo utilizador e apresentar resultados (por exemplo de pesquisas). Actualmente, esta interface fornece funcionalidades como pesquisa semântica, associar uma *Tag* RFID a uma localização física e associar uma *Tag* RFID a um ficheiro.

Por fim, o módulo `Interface com o Servidor` tem como função tratar de todos os detalhes da comunicação com o servidor, enviando os pedidos para o OSMOSIS-SFS e processando a resposta recebida. No entanto, ao contrário do OSMOSIS-SFS, que pode funcionar sem o módulo `Interface com o PDA`, no OSMOSIS-RFID o módulo `Interface com o Servidor` é essencial, visto que todas as funcionalidades fornecidas requerem pedidos ao servidor.

Capítulo 4

Implementação

Uma vez apresentada a arquitectura do sistema OSMOSIS, serão agora explicados os detalhes de implementação de todos os módulos que fazem parte do sistema. Numa primeira parte serão apresentados os módulos do OSMOSIS-SFS, seguindo-se as interfaces para comunicação entre o OSMOSIS-SFS e o OSMOSIS-RFID e por último explicados os módulos do OSMOSIS-RFID.

Importante referir que todo o sistema de ficheiros semântico foi desenvolvido para funcionar no Sistema Operativo, Microsoft Windows. Uma outra opção seria implementar o sistema, com auxilio da ferramenta FUSE [9] de modo a funcionar em Unix. No entanto, visto que os PDA's utilizados têm instalado o Windows CE como Sistema Operativo, por uma questão de compatibilidade, o sistema OSMOSIS (OSMOSIS-SFS + OSMOSIS-RFID) foi desenvolvido na linguagem de programação C# com auxílio da *framework* .NET 2.0.

4.1 Detalhes do OSMOSIS-SFS

O OSMOSIS-SFS desenvolvido é um sistema de ficheiros semântico, que para além de permitir realizar as funcionalidades típicas (indexação e pesquisa de ficheiros) de um sistema deste tipo, também possibilita a atribuição manual de *Tags* aos ficheiros assim como a integração de objectos reais no mundo virtual. Ou seja, com este sistema é possível guardar informação sobre localizações físicas e objectos que se encontram numa dada localização e que possuem uma relação com um ficheiro virtual. Posteriormente esta informação poderá ser utilizada para saber que ficheiros se encontram numa localização em específico.

Seguidamente serão explicados todos os módulos que fazem parte da arquitectura do OSMOSIS-SFS e que foram apresentados na Figura 3.4.

4.1.1 Tracer

O `Tracer` é o módulo responsável por monitorizar todos os acessos que são feitos ao sistema de ficheiros nativo do Microsoft Windows. Desta forma, sempre que uma aplicação criar, modificar, renomear ou apagar ficheiros e directorias, o Tracer regista essa acção.

Para conseguir realizar esta monitorização inicialmente foi considerada a hipótese de utilizar a ferramenta Installable File System [13] (IFS). Esta ferramenta fornece uma interface para a criação de novos sistemas de ficheiros, sem necessidade de modificar o *kernel* do Sistema Operativo, e para além disso permite interceptar todos os acessos ao sistema de ficheiros nativo do Microsoft Windows. Contudo, apesar do IFS conseguir fornecer um variado número de funcionalidades, requer uma programação de baixo nível, já bastante complexa. Por esta razão, optámos pela utilização do componente `FileSystemWatcher` [8] da *framework* .NET. O `FileSystemWatcher`, apesar de mais limitado em termos de funcionalidade,

permite identificar sempre que uma aplicação criar, modificar, renomear ou apagar ficheiros e directorias, e é bastante mais simples e intuitivo que o IFS. Assim, como o `FileSystemWatcher` fornece a funcionalidade necessária e é um componente da *framework* .NET, utilizada em todo o sistema, optou-se por este componente em vez do IFS, que possui muitas funcionalidades desnecessárias ao nosso sistema, e além disso é bastante complexo.

Como este componente é responsável por monitorizar acessos aos ficheiros é inicializado assim que o sistema OSMOSIS-SFS começa a sua execução. Desta forma conseguimos garantir que durante a execução do programa todas as interacções com ficheiros ou directorias irão ser registadas.

Como já referido anteriormente, para este módulo foi utilizado o componente `FileSystemWatcher` da *framework* .NET. Quando se inicializa este componente é necessário definir propriedades como a *drive* a monitorizar (no nosso caso *drive* "C:") e que métodos devem ser invocados consoante o evento lançado. De referir que os eventos a que foram associados métodos são "Created", "Changed", "Renamed" e "Deleted" para o caso em que um ficheiro é criado, modificado, renomeado e apagado, respectivamente. Antes da inicialização deste componente precisamos ainda definir que tipos de ficheiros pretendemos monitorizar. A forma de o fazer é atribuir à propriedade *filter* expressões do tipo "*.txt" e "". Com a primeira expressão pretende-se monitorizar apenas ficheiros do tipo "txt" e com a segunda todos os tipos de ficheiros e directorias. No nosso caso, e de forma a testar o correcto funcionamento do sistema, só pretendemos observar modificações em ficheiros com as seguintes extensões: .doc, .xls, .ppt, .pdf, .txt, .jpg, .mpg, .mp3. No entanto não é possível inserir uma expressão para monitorizar apenas um conjunto de tipos de ficheiros, como aquele que pretendemos. De forma a resolver esta limitação atribuímos à propriedade *filter* a expressão "", de forma a monitorizar todos os ficheiros e directorias, e quando é lançado um evento verifica-se inicialmente se está associado a um ficheiro do tipo que queremos monitorizar ou a uma directoria, caso contrário não é processado.

Uma vez que o OSMOSIS-SFS está em execução durante toda a sessão do Windows (para registar todos os acessos a ficheiros) foi necessário encontrar uma forma de evitar que o nosso sistema consuma demasiados recursos (principalmente CPU) enquanto o utilizador interage normalmente com a sua sessão de Windows. Assim, tornou-se evidente que a computação para extracção automática de *Tags* não poderia ter a mesma prioridade que outras aplicações, uma vez que do ponto de vista do utilizador todo o sistema ficaria mais "lento". De forma a resolver este problema decidimos que o processo do OSMOSIS-SFS teria uma prioridade "Abaixo do normal", o que é feito como última tarefa da inicialização do `Tracer`.

Após explicadas todas as inicializações necessárias para que o `Tracer` funcione do modo pretendido, iremos agora mostrar o processamento realizado sempre que é lançado um evento. Começaremos por explicar o método `OnChanged`, que está associado aos eventos "Created", "Changed" e "Deleted", e depois o método `OnRenamed`, que está associado ao evento "Renamed".

Método OnChanged

Este método é executado cada vez que se detecte a criação, modificação ou eliminação de um ficheiro ou directoria. Durante a sua execução começa por verificar se o ficheiro alterado é do tipo que se pretende monitorizar e se sim avisa o módulo `Análise do Conteúdo` sobre esta situação. Para isso envia-lhe o *path* completo do ficheiro, o tipo de ficheiro e o tipo de alteração (criação, modificação ou eliminação). Caso não seja a alteração de um ficheiro, mas sim de uma directoria a despoletar a execução deste método, primeiramente verifica-se se foi uma alteração do tipo eliminação e só nesta situação o módulo `Análise do Conteúdo` será avisado.

A razão pela qual apenas nos interessa saber quando uma directoria é eliminada e não quando é criada ou modificada deve-se ao facto de não extrairmos automaticamente *Tags* a partir de directorias. Ou seja, quando um ficheiro é criado ou modificado precisamos ser alertados para esta situação já que vai

ser necessário extrair *Tags* do seu conteúdo, o mesmo não acontece no caso de directorias. No entanto, precisamos saber quando uma directoria é eliminada porque esta pode ter associações com *Tags* que lhe foram atribuídas manualmente, sendo assim necessário remover essas associações da Base de Dados.

Método OnRenamed

O método `OnRenamed` está associado ao evento "Renamed" sendo assim executado cada vez que um ficheiro ou directoria é renomeado. Começa por verificar se o ficheiro renomeado é do tipo que se pretende monitorizar e em caso afirmativo avisa o módulo `Análise do Conteúdo`, enviando-lhe o *path* antigo do ficheiro, o novo *path* e o tipo de ficheiro. No caso de uma directoria ser renomeada procede-se da mesma forma, à excepção de apenas se enviar o antigo e o novo *path* da directoria para o módulo `Análise do Conteúdo`.

4.1.2 Análise do Conteúdo

Toda a funcionalidade deste módulo é baseada nas informações que lhe são passadas através do módulo `Tracer`. Ou seja, todas as acções tomadas têm por base os dados recebidos do `Tracer`. Para além disso, a sua estrutura interna está dividida em três partes com funcionalidades distintas. A primeira tem como função receber e processar os dados do `Tracer`, registando as acções que devem ser tomadas (extracção automática de *Tags* e actualização da base de dados) com base nessas informações. A segunda parte é responsável pela extracção automática de *Tags* a partir do conteúdo de ficheiros de texto e dos metadados dos ficheiros. A terceira tem como função actualizar a base de dados.

Para uma melhor compreensão, de seguida serão explicadas cada uma destas partes em separado.

Gestor Semântico

O gestor semântico é responsável por receber os dados provenientes do `Tracer` e com base nessas informações deve invocar os métodos para extracção automática de *Tags* (caso seja necessário) e para actualização da base de dados. A informação que recebe do `Tracer` diz-lhe se a alteração foi numa directoria ou num ficheiro, e neste caso que tipo de ficheiro, e ainda o tipo de alteração (criação, modificação, renomeação e eliminação).

Neste gestor semântico existe ainda um método para cada caso possível de alteração de ficheiros ou directorias. Os vários casos podem ser criar, modificar, renomear ou apagar um ficheiro, renomear e apagar um directoria. De seguida é explicado o tipo de processamento que é feito em cada um destes casos.

Quando um ficheiro é criado, modificado ou renomeado a primeira tarefa a realizar é a extracção automática de *Tags*. Esta extracção é sempre concretizada porque em qualquer um destes três casos existe uma grande probabilidade do conjunto de *Tags*, que identifica o ficheiro, ter ficado diferente. Para isso o gestor semântico invoca métodos da extracção automática de *Tags* e de seguida actualiza a base de dados invocando métodos do actualizador BD. Note-se que apenas o conjunto de *Tags* extraídas automaticamente poderá ter sido afectado pelas alterações dos ficheiros e por isso apenas este conjunto deverá ser actualizado, mantendo-se inalterado o conjunto de *Tags* atribuídas manualmente.

No caso em que um ficheiro é apagado ou uma directoria é renomeada ou apagada, não é necessário fazer extracção automática de *Tags*, visto que estas alterações não envolvem modificações no conjunto de *Tags*. Nesta situação apenas é necessário apagar registos da base de dados (no caso da eliminação de ficheiros ou directorias) ou actualização de *paths* nos registos da base de dados (no caso da renomeação de directorias).

Após a explicação do gestor semântico serão agora apresentadas as duas restantes partes deste módulo `Análise do Conteúdo`. Nomeadamente a extracção automática de *Tags* onde se mostra o algoritmo de

extracção a partir de ficheiros de texto e a partir dos metadados dos ficheiros. Posteriormente será explicado o funcionamento do actualizador BD.

Extracção Automática de *Tags*

Através do estudo dos sistemas descritos no Capítulo 2 podemos verificar que muitas vezes um utilizador lembra-se do assunto e de palavras que se encontram no interior de ficheiros, mas simplesmente não se lembra onde os guardou. Por esta razão tornou-se evidente que um conjunto de *Tags* bastante identificativas de um ficheiro facilita o seu processo de pesquisa. Assim, no nosso sistema foram desenvolvidos algoritmos para extrair automaticamente *Tags* do interior de ficheiros de texto (palavras mais usuais) assim como dos metadados (autor, data de criação, data da última modificação, nome, tamanho, tipo) de qualquer ficheiro.

Como já referido, a extracção de *Tags* a partir do conteúdo dos ficheiros é bastante importante no nosso sistema uma vez que se obtém um conjunto de palavras bastante identificativas de cada ficheiro. No entanto existe uma limitação óbvia de apenas se poder fazer esta extracção a partir de ficheiros de texto. Actualmente o sistema construído apenas analisa ficheiros com extensões .doc, .xls, .ppt, .pdf, .txt, contudo é bastante fácil expandir o sistema para que possa extrair *Tags* a partir de outro tipo de ficheiros de texto.

A classe responsável por esta extracção automática não exige inicializações complexas, sendo apenas necessário ler um ficheiro .txt e construir uma lista de "stopwords". Estas "stopwords" não são mais do que simples palavras portuguesas e inglesas que não têm qualquer significado semântico como por exemplo "Algum", "Aquela", "About"... Mais à frente, quando for explicado o algoritmo de extracção, a funcionalidade desta lista de "stopwords" ficará mais clara.

Para extrair *Tags* do interior de ficheiros de texto é necessário, em primeiro lugar, obter todo o texto do ficheiro para posteriormente poder ser processado. Como a forma de obter o texto varia de ficheiro para ficheiro, foi necessário criar um método diferente para cada tipo de ficheiro, apenas tendo como função extrair todo o texto, passando-o de seguida para o algoritmo de cálculo de *Tags* a partir do texto. Podemos assim verificar que expandir o sistema, para poder suportar outro tipo de ficheiros, pode ser feito de uma forma simples, bastando para isso construir um método que será responsável por obter o texto desse determinado tipo de ficheiro.

Vamos agora explicar o algoritmo de extracção automática de *Tags* a partir de texto. Para uma melhor compreensão o algoritmo será explicado em vários passos correspondentes às várias fases do processamento.

1. Fazer o *split* de todo o texto por um conjunto de caracteres, que funcionam como delimitadores de palavras (" ", ".", ",", ...), e guardar o resultado num *array*.

2. Para cada palavra no *array* verificar se é uma "stopword" e se sim passar para a próxima palavra, caso contrário é considerada uma palavra com possível significado semântico e é adicionada a uma lista que guarda este tipo de palavras, juntamente com o número de vezes que cada uma aparece no texto (par <palavra, nrOcorrencias>).

3. Para cada palavra da lista resultante de 2, calcular a taxa de ocorrência de cada palavra, ou seja, a percentagem de vezes que uma palavra é encontrada no texto. Para isso utiliza-se a seguinte fórmula para cada entrada da lista:

$$\text{percentagem} = \frac{\text{nrOcorrencias}}{\text{nrOcorrenciasTotal}}$$

nrOcorrencias = número de ocorrências de cada palavra

nrOcorrenciasTotal = Σ nrOcorrencias de cada palavra (excluindo naturalmente "stopwords").

Após efectuado este cálculo para cada palavra é construída uma nova lista que contém pares <palavra, percentagem>, mas onde apenas se encontram as palavras cuja taxa de ocorrência é igual ou superior a 0,5%. Todas as restantes palavras são descartadas uma vez que não se encontram muitas vezes no texto e considera-se por isso que provavelmente não contêm informação semântica sobre o dado ficheiro. Este valor de 0,5% foi obtido após realizar vários testes, sobre diversos ficheiros de texto, e da posterior análise dos resultados obtidos.

4. Ordenar a lista final, obtida em 3, por ordem decrescente de taxa de ocorrências e devolver as primeiras vinte palavras. Estas são as que têm uma grande probabilidade de ser facilmente lembradas por um utilizador.

Para além de extrair automaticamente *Tags* a partir do conteúdo de ficheiros de texto, o nosso sistema também possui um mecanismo para obter *Tags* a partir dos metadados. Neste caso não existe limitação quanto ao tipo de ficheiros visto que qualquer que seja o tipo tem pelo menos atributos como autor, data de criação, data de modificação, data do último acesso e o próprio nome do ficheiro. Para além destes atributos podem ser extraídos outras propriedades dependendo do tipo de ficheiro. Por exemplo nos ficheiros .mp3 podem ser obtidos atributos como o Intérprete, Álbum, Duração, entre outros.

A classe responsável por extrair *Tags* dos metadados possui apenas um método que é responsável por toda esta tarefa, qualquer que seja o ficheiro. Importante referir que na inicialização também necessita preencher uma lista com "stopwords". Esta é utilizada de forma a garantir que todas as *Tags* adicionadas à lista final não são "stopwords", ou seja, palavras sem significado semântico.

Através da utilização do componente `FileInfo` da *framework* .NET obtém-se *Tags* a partir da data de criação, data do último acesso, data da última modificação, nome, tipo de ficheiro, nome da directoria onde o ficheiro se encontra e tamanho do ficheiro. Estas *Tags* são extraídas de qualquer tipo de ficheiro porque qualquer um tem pelo menos este tipo de atributos. No entanto, podem também ser obtidas *Tags* de atributos adicionais como acontece por exemplo nos ficheiros .mp3. Para isso é sempre feita uma pesquisa por outros atributos que os ficheiros possam ter e em caso afirmativo esses atributos são incluídos como *Tags* dos ficheiros.

Importante ainda mencionar que antes de adicionar qualquer palavra à lista final de *Tags*, é feita uma verificação de modo a garantir que essas palavras não são "stopwords". Através desta verificação conseguimos assegurar que a lista final contém apenas *Tags* com significado semântico e que possam ser facilmente recordadas pelo utilizador.

Actualizador BD

Depois de realizado todo o processamento necessário, devido a alterações em ficheiros ou directorias, é necessário actualizar a base de dados para que esta fique consistente com estas novas alterações. Para isso existe uma classe responsável por todas as actualizações na BD que garante a consistência nos dados. Esta possui um método específico para cada evento que possa ser lançado (ficheiro criado, modificado, renomeado, apagado, directoria renomeada ou apagada), sendo cada um responsável por actualizar a BD consoante a alteração no sistema de ficheiros.

Na Secção 4.1.3, contexto do módulo `Base de Dados`, serão explicados em pormenor todos os métodos que fazem parte desta classe de actualização. Aí serão enunciadas todas as actualizações necessárias consoante o tipo de evento lançado.

4.1.3 Base de Dados

De forma a poder utilizar a informação semântica obtida, tornou-se evidente a necessidade de a guardar de forma persistente. Assim, decidimos que a melhor opção seria guardar toda a informação, necessária ao correcto funcionamento do nosso sistema, numa base de dados MySQL [20]. De seguida serão apresentadas as tabelas existentes na base de dados, assim como a função e tipo de informação que cada uma guarda.

Tabelas

A principal tabela da nossa base de dados é a que permite guardar para cada ficheiro as suas *Tags* ("filestags"). Esta é composta por três colunas: *path* do ficheiro, *Tag* e uma coluna para indicar se a *Tag* foi extraída automaticamente ou atribuída pelo utilizador. Esta última permite diferenciar os dois tipos de *Tags* de forma a facilitar a sua actualização e de aumentar a precisão do algoritmo de pesquisa.

Para o correcto funcionamento do OSMOSIS-SFS apenas é necessária a tabela "filestags", visto que contém toda a informação semântica recolhida. No entanto, a integração com o mundo real exige que seja guardada informação acerca da localização de objectos (relacionados com ficheiros do OSMOSIS-SFS). Para isso existem mais três tabelas ("filesrfids", "localizacoesfiles" e "localizacoesfisicas") de forma a obter as funcionalidades pretendidas para o OSMOSIS-RFID. A tabela "filesrfids" possui uma coluna com o *path* do ficheiro e outra com o número da *Tag* RFID. Esta tabela permite guardar informação sobre os ficheiros do OSMOSIS-SFS que estão relacionados com objectos reais (marcados com *Tags* RFID). Na tabela "localizacoesfiles" são relacionadas as localizações físicas com ficheiros do OSMOSIS-SFS, ou seja, que objectos (relacionados com ficheiros do OSMOSIS-SFS) estão numa determinada localização física. Por último, a tabela "localizacoesfisicas" guarda todas as localizações físicas existentes no sistema assim como o correspondente número da *Tag* RFID.

Actualizador Base de Dados

Para evitar dispersar as funções de acesso à base de dados por várias partes do sistema, criámos uma classe responsável por todas as actualizações na BD. É nesta classe que estão definidos os diversos métodos responsáveis por manter os dados coerentes.

Os métodos existentes nesta classe são os seguintes:

- `public void ficheiroCriado(string ficheiro, List<string> tags)` - Este método é invocado de cada vez que um ficheiro é criado, recebendo o nome do ficheiro e a lista de *Tags* extraídas automaticamente. Actualiza a tabela "filestags".

- `public void ficheiroModificado(string ficheiro, List<string> tags)` - Este método é invocado de cada vez que um ficheiro é modificado, recebendo o nome do ficheiro e a lista de *Tags* extraídas automaticamente. Actualiza a tabela "filestags".

- `public void ficheiroRenomeado(string oldFile, string newFile, List<string> tags)` - Este método é invocado de cada vez que um ficheiro é renomeado, recebendo o nome antigo e novo do ficheiro e a lista de *Tags* extraídas automaticamente. Actualiza as tabelas "filestags", "filesrfids" e "localizacoesfiles".

- `public void ficheiroApagado(string ficheiro)` - Este método é invocado de cada vez que um ficheiro é eliminado, recebendo o nome do ficheiro. Actualiza as tabelas "filestags", "filesrfids" e "localizacoesfiles".

- `public void tagIntroduzida(string ficheiro, string tag)` - Este método é invocado quando o utilizador pretende atribuir manualmente uma *Tag*, recebendo o nome do ficheiro e a *Tag* a atribuir. Actualiza a tabela "filestags".

- `public void pastaRenomeada(string nomeAntigo, string nomeNovo)` - Este método é invocado sempre que uma pasta é renomeada (poderá ser necessário actualizar o *path* dos ficheiros), recebendo o nome antigo e novo da directoria. Actualiza as tabelas "filestags", "filesrfids" e "localizacoesfiles".

- `public void relacionaFicheiros(string fileName, string otherFileName)` - Este método é invocado quando um utilizador pretende colocar uma referência explícita entre dois ficheiros. Recebe o nome dos dois ficheiros e actualiza a tabela "filestags".

- `public void localizacaoIntroduzida(string localizacao)` - Este método é invocado quando um utilizador pretende inserir uma nova localização física no sistema. Recebe a localização e actualiza a tabela "localizacoesfisicas".

- `public void localizacaoRemovida(string localizacao)` - Este método é invocado quando um utilizador pretende remover uma localização física do sistema. Recebe a localização e actualiza as tabelas "localizacoesfisicas" e "localizacoesfiles".

- `public void adicionaAssociacaoFisica(string localizacao, string file)` - Este método é invocado quando um utilizador pretende associar um ficheiro (relacionado com um objecto físico) a uma localização física. Recebe a localização e o nome do ficheiro e actualiza a tabela "localizacoesfiles".

- `public string associaLocalizacaoTagRFID(string localizacao, string tagUID)` - Este método é invocado quando o PDA envia um pedido de associação entre uma *Tag* RFID e uma localização física. Recebe a localização e o número da *Tag* e actualiza as tabelas "filesrfids" (caso a *Tag* RFID já se encontre associada a um ficheiro) e "localizacoesfisicas".

- `public string associaFicheiroTagRFID(string ficheiro, string tagUID)` - Este método é invocado quando o PDA envia um pedido de associação entre uma *Tag* RFID e um ficheiro. Recebe o nome do ficheiro e o número da *Tag* e actualiza as tabelas "localizacoesfisicas" (caso a *Tag* RFID já se encontre associada a uma localização) e "filesrfids".

Visualizador Base de Dados

Da mesma forma que o Actualizador BD, foi também criada uma classe responsável por fazer consultas na base de dados. É nesta classe que se encontram todos os métodos que podem fazer consultas na base de dados, à excepção das pesquisas de ficheiros, que se encontram directamente no módulo `Motor de Pesquisa Semântico`. Estes métodos não se encontram no visualizador BD, uma vez que a pesquisa é uma das principais funcionalidades do nosso sistema, envolvendo grande tratamento de dados. Por isso optou-se por fazer as consultas à BD directamente no módulo de pesquisa, de forma a manter todos os detalhes do algoritmo num único módulo.

Os diversos métodos existentes para consultas na BD são os seguintes:

- `public List<string> searchFileTags(string ficheiro)` - Método que devolve uma lista com todas as *Tags* de um dado ficheiro.

- `public List<string> searchPhysicalLocation()` - Método que devolve todas as localizações físicas existentes no sistema.

- `public List<string> searchFilesByRFID(string tagUID)` - Método que devolve todos os ficheiros associados com um número de *Tag* RFID. Se a *Tag* RFID estiver associada com uma localização física serão devolvidos todos os ficheiros que se encontrem nessa localização. Já se estiver associada directamente com um ficheiro, será devolvido apenas esse ficheiro.

4.1.4 Motor de Pesquisa Semântico

Uma vez explicada a forma como é extraída e guardada toda a informação semântica necessária ao correcto funcionamento do nosso sistema, iremos agora mostrar o motor de pesquisa semântico. Este módulo é essencial visto que toda a informação recolhida só é útil se puder ser utilizada para informar as pesquisas. Para isso, foi construída uma classe que tem como responsabilidade efectuar pesquisas de ficheiros a partir de palavras-chave (*Tags*). No programa 4.1 é apresentado o método ("search") responsável pela pesquisa.

Programa 4.1: Algoritmo de pesquisa de ficheiros

```
1  public List<string>[] search(List<string> termosPesquisa) {
2    List<string>[] resultList = new List<string>[2];
3    resultList[0] = new List<string>();
4    resultList[1] = new List<string>();
5
6    Dictionary<string,int> resultadosParciais = new Dictionary<string,
       int>();
7
8    foreach(termo in termosPesquisa){
9      tmp = executeQuery("SELECT fileName FROM filestags WHERE tag LIKE
           '%" + termo + "%' AND tag NOT LIKE '%c:\\%'");
10     foreach(file in tmp){
11       if(resultadosParciais.ContainsKey(file)){
12         int importancia = resultadosParciais[file];
13         resultadosParciais[file] = importancia + 1;
14       }else{
15         resultadosParciais.Add(file, 1);
16       }
17     }
18   }
19
20   List<string> resultadosParciaisOrdenados =
       ordenaResultadosPorImportancia(resultadosParciais);
21   resultList[0] = resultadosParciaisOrdenados;
22
23
24   tmp = executeQuery("SELECT * FROM filestags WHERE conteudo='no'");
25   foreach(line in tmp) {
26     string fileName = line[1];
27     string tag = line[2];
28     if (!resultList[1].Contains(tag) && !resultList[0].Contains(tag)
         && resultList[0].Contains(fileName))
```

```
29          resultList[1].Add(tag);
30      }
31
32      return resultList;
33  }
```

Este método recebe uma lista com termos para pesquisa, introduzidos pelo utilizador e devolve um *array* com duas listas de *strings*. A primeira contém os ficheiros[1] de resultado, ordenados por grau de importância e a segunda contém os ficheiros relacionados explicitamente com os da primeira lista, assim como *Tags* que permitem refinar a pesquisa. Estas *Tags* são apenas as que foram atribuídas manualmente aos ficheiros de resultado e não foram introduzidas como termos de pesquisa.

Nas linhas 8-18 é feito um "SELECT" à base de dados por cada termo na lista de termos. Para cada ficheiro de resultado (linha 10) verificar se este já existe no *Dictionary* e em caso afirmativo aumentar a sua importância em mais um valor. Caso contrário, adicionar o ficheiro ao *Dictionary* com importância 1. Após realizada esta computação, segue-se a ordenação da lista de resultados por ordem decrescente de importância e o preenchimento do `resultList[0]` com esta lista de ficheiros ordenada (linhas 20 e 21).

A segunda parte do algoritmo consiste na construção de uma lista com os ficheiros relacionados explicitamente com os de resultado, assim como *Tags* que permitem o posterior refinamento da pesquisa (linhas 24-30). A forma como é colocada uma relação explícita entre dois ficheiros, será explicada mais à frente na Secção 4.1.5. Aqui, de forma a evitar realizar vários acessos à BD, começa-se por efectuar um "SELECT" à BD (linha 24), obtendo todos os ficheiros que possuem *tags* atribuídas manualmente e ficheiros relacionados. Depois, para cada linha de resultado do "SELECT" (linhas 25-29), obtém-se o nome do ficheiro e a *tag* manual (ou ficheiro relacionado) e verifica-se se esta não existe na `resultList[0]` e na `resultList[1]` e se o nome do ficheiro existe no resultList[0]. Caso estas condições se verifiquem, a *tag* é adicionada à lista `resultList[1]`. Com estas verificações garantimos que não são retornados resultados repetidos, mas que todos estão relacionados com os ficheiros do `resultList[0]`.

Após realizar todo o processamento explicado em cima, retorna-se o `resultList` que contém na primeira lista os ficheiros de resultado e na segunda os ficheiros relacionados, assim como *Tags* que permitem o refinamento da pesquisa por parte do utilizador.

4.1.5 Interface com o Utilizador

Após explicado todo o processamento que é realizado sem que o utilizador se aperceba, vamos agora apresentar a interface fornecida para interagir com o nosso sistema. Nesta fase não foi dada importância ao aspecto gráfico, mas sim às funcionalidades da interface. De seguida serão apresentadas as várias "janelas" que fazem parte do nosso sistema.

[1]Por uma questão de simplicidade iremos referir-nos (nesta secção) a "caminhos para ficheiros" como apenas "ficheiros"

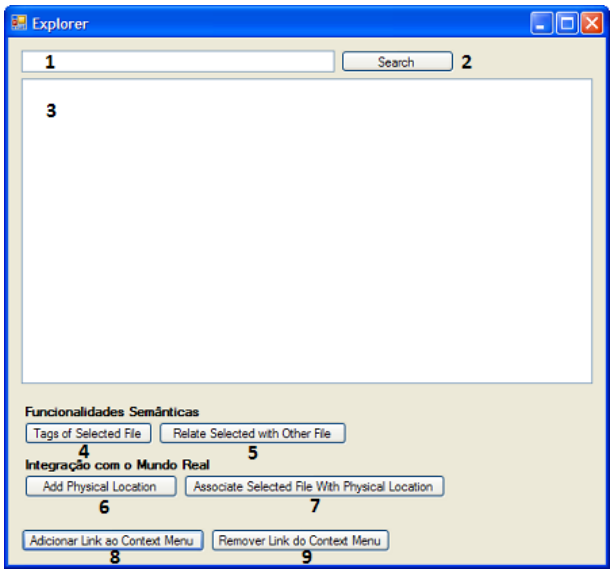

Figura 4.1: Interface principal.

Na Figura 4.1 está apresentada a interface principal do OSMOSIS-SFS. Vamos agora explicar o funcionamento de cada componente, marcado com um número a negrito.

1. Caixa de texto onde o utilizador insere os termos pelos quais pretende efectuar uma pesquisa.

2. Botão para efectuar uma pesquisa de ficheiros.

3. Secção onde são apresentados os resultados da pesquisa.

4. Botão que permite visualizar as *Tags* associadas ao ficheiro seleccionado. Selecciona-se o ficheiro pretendido, carrega-se neste botão e de seguida é apresentada a janela da Figura 4.4.

5. Botão que permite relacionar explicitamente o ficheiro seleccionado com outro qualquer. Esta funcionalidade permite colocar relações explícitas entre dois ficheiros, de modo a que quando se efectua uma pesquisa possam ser devolvidos também ficheiros que estão relacionados com os ficheiros de resultado. Para usar esta funcionalidade, o utilizador deve seleccionar o ficheiro pretendido, carregar neste botão e em seguida é apresentado um *OpenFileDialog* para seleccionar o ficheiro relacionado. De referir ainda que estas referências explícitas são bidireccionais, ou seja, quando se coloca uma referência de A para B, automaticamente coloca-se também de B para A.

6. Botão que abre a janela representada na Figura 4.5, para que o utilizador possa adicionar ou remover localizações físicas do sistema.

7. Botão que permite associar o ficheiro seleccionado com uma localização física (já existente no sistema). Selecciona-se o ficheiro pretendido, carrega-se neste botão e de seguida é apresentada a janela da Figura 4.6 para escolher a localização física pretendida.

8. Botão que permite activar a atribuição manual de *Tags* aos ficheiros.

9. Botão que permite desactivar a atribuição manual de *Tags* aos ficheiros.

Depois de explicado o funcionamento global da interface principal, iremos agora detalhar cada funcionalidade do nosso sistema.

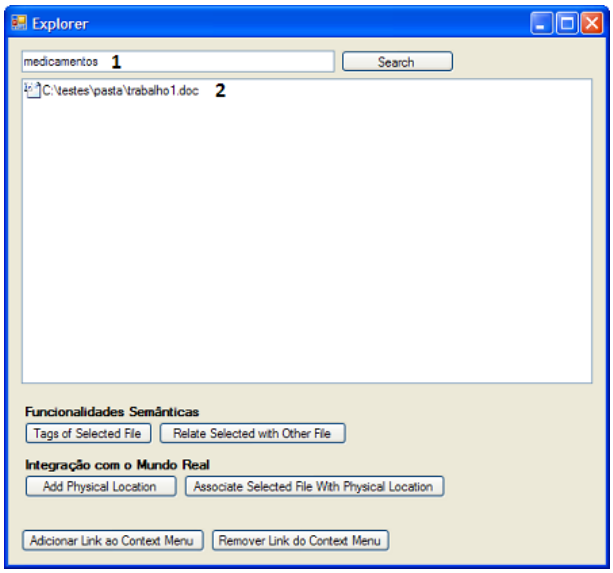

Figura 4.2: Pesquisa pelo termo "medicamentos".

Na Figura 4.2 está representado o resultado de uma pesquisa pelo termo "medicamentos". Como se pode observar, em 1 foi introduzida a palavra "medicamentos" e em 2 foram apresentados os resultados da pesquisa. Neste caso em específico só foi encontrado um ficheiro que corresponde aos termos introduzidos. De referir ainda que este "trabalho1.doc" é um relatório na área de Farmácia e que "medicamentos" é uma palavra que se encontra várias vezes no texto.

Para um exemplo de pesquisas que retornem um maior número de resultados, observe-se a Figura 4.3. Neste caso, foi efectuada uma pesquisa pelo termo "texto" (1), sendo retornados os resultados 2 ("texto4.pdf" a "texto11.txt"), 3 ("imagem.jpg" e "filme.mpg") e 4 ("tese"). O conjunto 2 contém todos os ficheiros que correspondem à pesquisa pela palavra "texto". Em 3 encontram-se os ficheiros que estão relacionados explicitamente com alguns do conjunto 2. E em 4 encontra-se uma *Tag*, que foi atribuída manualmente a algum dos ficheiros de 2 e que possibilita o refinamento da pesquisa.

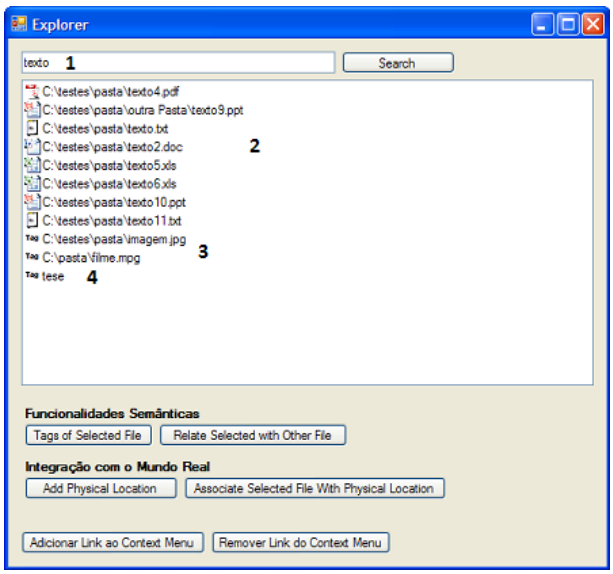

Figura 4.3: Pesquisa pelo termo "texto".

Importante referir ainda que esta interface principal permite ao utilizador fazer *Double Click* em cada

um dos resultados da pesquisa, sendo obtidos os seguintes comportamentos, consoante o tipo de resultado:

- **Ficheiro:** O ficheiro é aberto com o programa pré-definido para abrir esse tipo de ficheiro.
- **Directoria:** É apresentado o conteúdo da directoria na lista de resultados.
- ***Tag:*** É realizada uma nova pesquisa, acrescentando aos termos introduzidos a *Tag* seleccionada. Ou seja, no caso da figura 4.3, quando o utilizador faz *Double Click* sobre a *Tag* "tese", é automaticamente feita uma pesquisa por "texto" e "tese".

No nosso sistema, um utilizador após efectuar uma pesquisa pode ver as *Tags* associadas a um determinado ficheiro. Para isso, basta seleccionar (um *Click*) o ficheiro pretendido, carregar no botão "Tags of Selected File" e de seguida é apresentada a interface da figura 4.4. Neste caso em específico são mostradas as *Tags* do ficheiro "texto11.txt".

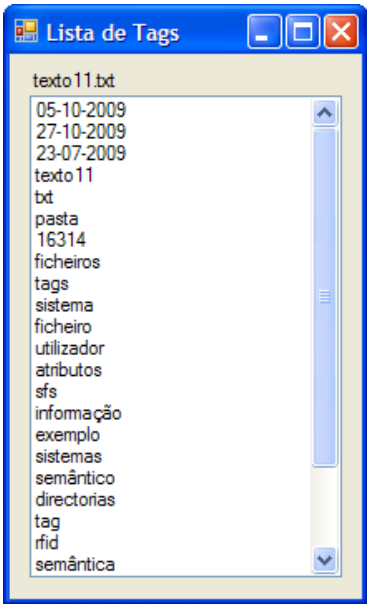

Figura 4.4: Lista de *Tags* do ficheiro "texto11.txt".

Para colocar relações explícitas entre ficheiros, um utilizador deve seleccionar o ficheiro pretendido (da lista de resultados da pesquisa) e seguidamente carregar no botão "Relate Selected with Other File". Após estas acções abre-se um *OpenFileDialog*, para que o utilizador possa escolher o segundo ficheiro a relacionar. Depois de seleccionado o segundo ficheiro, será criada uma relação explícita entre os dois.

Na Figura 4.5 está representada a interface para adicionar ou remover localizações físicas do sistema. Na lista 1 encontram-se as localizações físicas já existentes no sistema e o botão 2 permite remover uma localização seleccionada. Em 3 podemos introduzir o nome de uma nova localização e o botão 4 adiciona essa nova localização ao sistema.

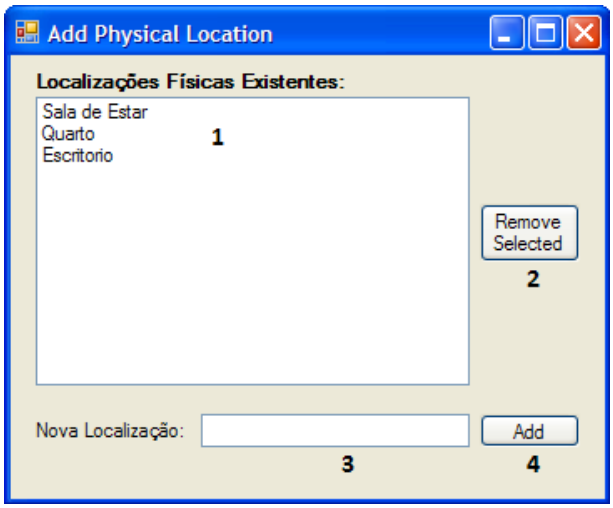

Figura 4.5: Interface para adicionar ou remover localizações físicas do sistema.

Como já referido anteriormente, o nosso sistema permite relacionar ficheiros com localizações físicas. Ou seja, se por exemplo existir um ficheiro "tese.pdf" que foi impresso e colocado fisicamente na sala de estar, vamos querer relacionar o ficheiro "tese.pdf" com a localização física "Sala de Estar". Para esta funcionalidade podemos recorrer à interface representada na Figura 4.6. Após seleccionar o ficheiro pretendido, é lançada esta interface onde o utilizador pode escolher a localização física (através de uma *ComboBox*) e de seguida confirmar a associação com o botão "Make Association".

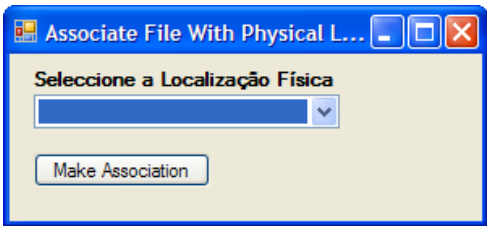

Figura 4.6: Interface para associar um ficheiro a uma localização física.

4.1.6 *Tags* Introduzidas pelo Utilizador

Para a funcionalidade de atribuição manual de *Tags* aos ficheiros, decidimos criar uma interface o mais intuitiva possível. Optámos então por acrescentar a opção de "Adicionar Tag" ao *Context Menu* dos ficheiros e directorias. Deste modo, o utilizador pode simplesmente carregar com o botão direito do rato em cima do ficheiro pretendido (figura 4.7), seleccionar a opção "Adicionar Tag" e em seguida é apresentada uma interface (Figura 4.8) para introduzir a *Tag*.

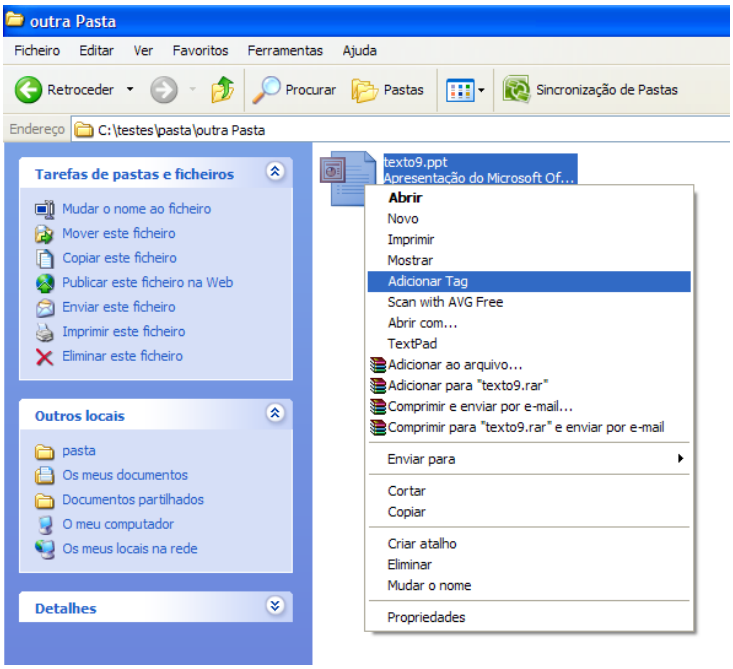

Figura 4.7: *Context Menu* com a opção "Adicionar Tag".

Para colocar a opção de "Adicionar Tag" no *Context Menu* dos ficheiros e directorias, modificámos o *registry* do Microsoft Windows de modo a que esta opção fosse visível para qualquer tipo de ficheiro (*) e para directorias (Folder). De notar que, quando esta nova opção é escrita no *registry* é-lhe associado o executável do nosso programa. Assim, quando o utilizador selecciona a opção "Adicionar Tag", imediatamente é lançada a interface representada na figura 4.8. Podemos também remover esta opção do *Context Menu*, bastando para isso apagar a entrada correspondente do *registry*. Actualmente, o utilizador pode activar ou desactivar a opção "Adicionar Tag" através da interface principal do nosso sistema (botões "Adicionar Link ao Context Menu" e "Remover Link do Context Menu" respectivamente).

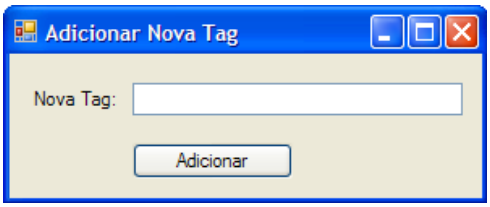

Figura 4.8: Interface para a atribuição manual de *Tags*.

Importante ainda referir que, quando o utilizador selecciona a opção "Adicionar Tag", o nosso programa começa por verificar quais os itens (ficheiros ou pastas) da janela actual é que estão seleccionados, e quando lança a interface da Figura 4.8, já sabe o *path* dos ficheiros e directorias seleccionadas. Assim, é possível atribuir de uma só vez a mesma *Tag* a um conjunto de ficheiros, ou até mesmo a um conjunto de directorias, bastando para isso, seleccionar o conjunto de ficheiros pretendido e no *Context Menu* escolher a opção "Adicionar Tag".

4.1.7 Interface com o PDA

Até este ponto, foram explicados os principais módulos do OSMOSIS-SFS, que permitem o seu correcto funcionamento como um sistema de ficheiros semântico. No entanto, para a interacção com o OSMOSIS-RFID (integração com o mundo real) terá de existir um servidor que possa satisfazer os seus pedidos.

Por esta razão, implementámos o módulo `Interface com o PDA`, que tem como principal função receber e responder a pedidos vindos do OSMOSIS-RFID. De notar ainda que o OSMOSIS-SFS pode funcionar correctamente, somente como um sistema de ficheiros semântico, sem o uso deste módulo.

Para a comunicação com o PDA implementámos um servidor que está a funcionar num modo independente do resto do OSMOSIS-SFS. Ou seja, quando o sistema é iniciado o servidor fica activo numa nova *thread* de forma a responder aos pedidos do PDA, sem perturbar o funcionamento do restante sistema. Este servidor está a correr num endereço IP e porto conhecidos entre o OSMOSIS-SFS e o OSMOSIS-RFID e cada pedido que recebe é transferido para uma classe responsável pelo processamento das mensagens. Todas estas mensagens recebidas vêm em texto e a razão pela qual foi utilizada esta forma de comunicação, de baixo nível, irá ser explicada mais à frente na Secção 4.2.

Na classe responsável pelo processamento das mensagens existe um método para cada tipo de pedido, por isso a primeira tarefa a realizar quando se recebe uma mensagem é verificar qual o tipo de pedido e invocar o respectivo método. Actualmente o sistema aceita os seguintes pedidos: pedido de ficheiros (*path*) e localizações associadas a uma *Tag* RFID, pesquisa de ficheiros, localizações físicas existentes, associar uma *Tag* RFID a uma localização física, associar uma *Tag* RFID a um ficheiro, transferência de um ficheiro e conteúdo (ficheiros) de uma pasta. Estes pedidos estão todos relacionados com a funcionalidade do OSMOSIS-RFID que será detalhada mais à frente na Secção 4.3.

Os métodos responsáveis pelo processamento de cada tipo de pedido são os seguintes:

- `processaPedidoRFIDP` - Este método é responsável por devolver as localizações físicas e ficheiros associados a uma *Tag* RFID. Ou seja, se a *Tag* estiver atribuída a um localização física deve enviar a localização e os ficheiros que lhe estão associados, já se estiver atribuída a um ficheiro, devolve esse ficheiro.
- `processaPedidoPESQP` - Responsável pelos pedidos de pesquisa vindos do PDA. Funciona da mesma forma que uma pesquisa no OSMOSIS-SFS, ou seja, efectua uma procura pelos termos recebidos e no final devolve os *paths* dos ficheiros de resultado como resposta.
- `processaPedidoGETLP` - Método que tem como função devolver todas as localizações físicas existentes no sistema.
- `processaPedidoALTRP` - Este método é responsável por associar a *Tag* RFID à localização física recebida.
- `processaPedidoAFTRP` - É responsável por atribuir a *Tag* RFID ao *path* do ficheiro recebido.
- `processaPedidoGETFP` - Este método é responsável por transferir o ficheiro solicitado para o PDA.
- `processaPedidoGETPP` - Método que devolve o conteúdo (ficheiros) de uma directoria.

4.2 Interfaces para Comunicação entre o OSMOSIS-SFS e o OSMOSIS-RFID

Para o correcto funcionamento do nosso sistema é essencial existir uma interface de comunicação bem definida e conhecida pelos dois intervenientes (OSMOSIS-SFS e OSMOSIS-RFID). Inicialmente pensámos que a melhor solução seria a utilização do componente Remoting [21] da *framework* .NET 2.0. Este componente permite invocar métodos remotos, sem nos preocuparmos com detalhes (de baixo nível) da comunicação. Esta solução seria a ideal visto que o OSMOSIS-RFID poderia invocar directamente os métodos (Secção 4.1.7) remotos do OSMOSIS-SFS. No entanto, verificámos que a *.NET Compact Framework 2.0* utilizada no sistema do PDA (OSMOSIS-RFID) não suporta o componente Remoting.

Foi ainda considerada a utilização de *Web Services* e serialização XML, no entanto, esta solução iria aumentar o consumo de memória e largura de banda. Assim, após alguma análise, decidimos optar pela utilização directa de *Sockets*, suportados tanto pela *.NET Framework 2.0* como pela *.NET Compact Framework 2.0*. Como os *sockets* exigem uma programação de baixo nível, só permitindo enviar e receber bytes, foi necessário criar um conjunto bem definido de mensagens de texto para a comunicação entre o OSMOSIS-SFS e OSMOSIS-RFID. Note-se que para cada funcionalidade foram criadas duas mensagens, uma de pedido ao servidor (OSMOSIS-SFS) e outra de resposta ao pedido. Estas novas mensagens criadas como protocolo de comunicação, são apresentadas de seguida.

- `RFIDP:‘‘código da Tag RFID’’` - Mensagem enviada para o servidor (OSMOSIS-SFS) quando o PDA lê uma *Tag* RFID. É um pedido da localização e ficheiros associados à *Tag* lida.
- `RFIDR:‘‘número de caracteres da localização|localização|path do ficheiro1|path do ficheiro2|path do ficheiro3’’` - Enviado para o cliente (OSMOSIS-RFID) como resposta a um pedido do tipo RFIDP. Se a *Tag* RFID lida não estiver atribuída a uma localização mas sim a um ficheiro, os campos “número de caracteres da localização” e “localização” vêm vazios.
- `PESQP:‘‘termos para pesquisa’’` - Enviado para o servidor quando o utilizador faz uma pesquisa no PDA.
- `PESQR:‘‘path do ficheiro1|path do ficheiro2|path do ficheiro3’’` - Mensagem enviada para o cliente como resposta a um pedido do tipo PESQP.
- `GETLP:‘‘’’` - Enviado para o servidor de forma a obter todas as localizações físicas existentes no sistema.
- `GETLR:‘‘número de caracteres da localizacao1|localizacao1|número de caracteres da localizacao2|localizacao2’’` - Enviado para o cliente como resposta a um pedido do tipo GETLP.
- `ALTRP:‘‘número de caracteres da localização|localização|código da Tag RFID’’` - Mensagem enviada para o servidor de forma a associar uma localização física a uma *Tag* RFID.
- `ALTRR:‘‘resultado (0-erro, 1-sucesso)’’` - Enviado para o cliente de forma a confirmar, ou não, o sucesso da atribuição.
- `AFTRP:‘‘path do ficheiro|código da Tag RFID’’` - Mensagem enviada para o servidor de forma a associar um ficheiro a uma *Tag* RFID.
- `AFTRR:‘‘resultado (0-erro, 1-sucesso)’’` - Enviado para o cliente de forma a confirmar, ou não, o sucesso da atribuição.
- `GETFP:‘‘path do ficheiro’’` - Mensagem enviada para o servidor quando um utilizador pretende aceder a um ficheiro no PDA (*download* do ficheiro).
- `‘‘bytes do ficheiro’’` - Como resposta a um pedido do tipo GETFP são enviadas duas mensagens para o cliente. A primeira contém o número de bytes do ficheiro a ser transmitido e a segunda os bytes do ficheiro.
- `GETPP:‘‘path da directoria’’` - Enviado para o servidor quando um utilizador pretende aceder (no PDA) ao conteúdo de uma directoria.
- `GETPR:‘‘path do ficheiro1|path do ficheiro2|path do ficheiro3’’` - Mensagem enviada para o cliente como resposta a um pedido do tipo GETPP.

4.3 Detalhes do OSMOSIS-RFID

O OSMOSIS-RFID desenvolvido, tem como principal funcionalidade dar ao utilizador uma forma de poder navegar no mundo virtual, ao mesmo tempo que se desloca no mundo físico. Ou seja, para além de permitir pesquisas de ficheiros (como o OSMOSIS-SFS), com este sistema é possível por exemplo ler a *Tag* RFID que está na porta da sala de estar e saber imediatamente que objectos (relacionados com ficheiros) estão no seu interior. Importante referir que estes objectos têm de ser previamente relacionados com a sua localização física, no OSMOSIS-SFS. Como exemplo destes objectos, podemos considerar um álbum físico de fotografias que foi obtido a partir da impressão de fotografias digitais existentes no PC.

De seguida serão explicados todos os módulos que fazem parte da arquitectura do OSMOSIS-RFID e que foram introduzidos na Figura 3.5.

4.3.1 Leitor RFID

Uma vez que o nosso PDA está equipado com um leitor de *Tags* RFID, tornou-se evidente a necessidade de criar um módulo responsável pelo tratamento dos dados lidos. Este módulo, após todas as configurações iniciais, activa o leitor, associando um evento à chegada de dados. Ou seja, por cada *Tag* RFID lida é lançado um evento para que os dados recebidos sejam processados pelo sistema.

O método responsável pelo tratamento dos dados, recebidos do leitor, é invocado cada vez que seja lida uma *Tag* RFID. Este, começa por extrair o número da *Tag* (TagUID) lida e seguidamente envia um pedido ao servidor, de forma a obter a localização e os ficheiros associados à *Tag* lida. Note-se que este pedido não é efectuado directamente, mas sim através do módulo `Interface com o Servidor`, explicado na Secção 4.3.4. Após receber a resposta do servidor, analisa-a de forma a perceber se a *Tag* lida está associada a uma localização ou a um ficheiro. Caso se verifique que está associada uma localização, esta é apresentada assim como os ficheiros relacionados, caso contrário apenas é apresentado um ficheiro (associado com a *Tag* lida).

Importante referir ainda que para esta funcionalidade fornecer informação útil e interessante para o utilizador, este deve ter previamente efectuado as associações entre "*Tags* RFID - localizações", "*Tags* RFID - ficheiros" e "ficheiros - localizações", no OSMOSIS-SFS.

4.3.2 Interface com o Utilizador

Como já referido, o OSMOSIS-RFID é um cliente do sistema OSMOSIS-SFS, que tem como principal objectivo apresentar informação ao utilizador. Por esta razão, foi dada especial atenção à interface gráfica do OSMOSIS-RFID. Apesar de tentarmos elaborar uma interface o mais parecida possível com a interface do OSMOSIS-SFS (de forma a aumentar a usabilidade), mais uma vez não foi dada grande relevância ao aspecto gráfico, mas sim às funcionalidades suportadas. De seguida são apresentadas as três "janelas" que fazem parte do OSMOSIS-RFID.

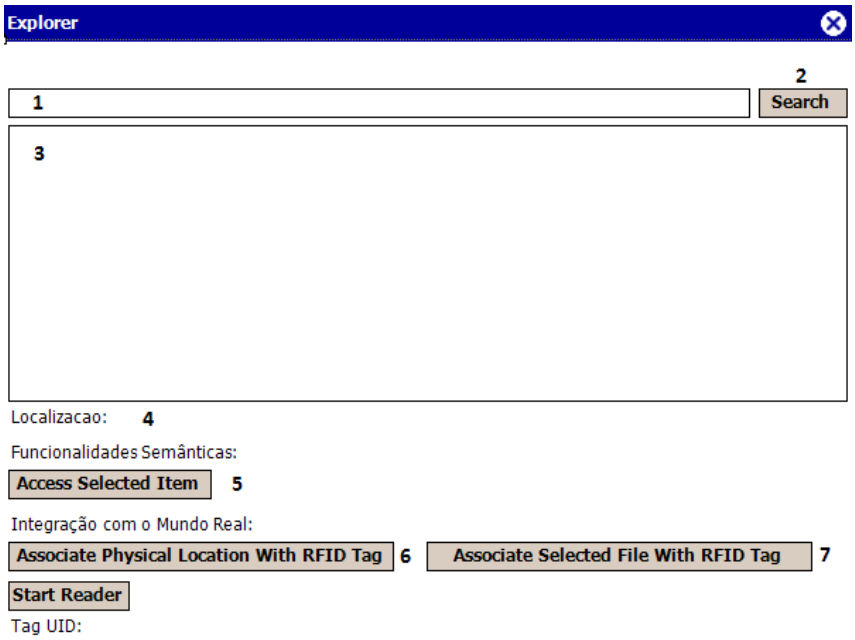

Figura 4.9: Interface principal do OSMOSIS-RFID.

Na Figura 4.9 está representada a interface principal do sistema OSMOSIS-RFID. Vamos de seguida explicar o funcionamento de cada componente, marcado com um número a negrito.

1. Caixa de texto onde o utilizador insere os termos pelos quais pretende efectuar uma pesquisa.
2. Botão para efectuar uma pesquisa de ficheiros.
3. Secção onde são apresentados os resultados das pesquisas e das leituras de *Tags* RFID.
4. Secção onde é apresentada a localização física do utilizador (através da leitura de uma *Tag* RFID).
5. Botão que permite aceder ao ficheiro seleccionado. Na realidade é o equivalente ao *double click* utilizado na interface do OSMOSIS-SFS. Contudo, foi necessário utilizar esta forma de aceder aos ficheiros, uma vez que a *.NET Compact Framework 2.0* não suporta o evento *double click* na secção de resultados (3).
6. Botão que permite associar uma *Tag* RFID a uma localização física já existente no sistema. Ao carregar no botão é apresentada uma interface (Figura 4.10) que permite seleccionar a localização pretendida.
7. Botão que permite associar o ficheiro seleccionado com uma *Tag* RFID. Primeiro selecciona-se o ficheiro, depois carrega-se neste botão e seguidamente é apresentada a interface da Figura 4.11.

No OSMOSIS-RFID a funcionalidade de pesquisa é processada da mesma forma que no OSMOSIS-SFS. A única diferença, reside no facto de neste caso a pesquisa ser realizada com um pedido ao servidor (OSMOSIS-SFS). Contudo, como o motor de pesquisa utilizado é o mesmo (motor de pesquisa semântico do OSMOSIS-SFS), uma pesquisa pelo termo "medicamentos" irá retornar os mesmos resultados em ambos os sistemas. Da mesma forma, quando o PDA lê uma *Tag* RFID, é feito um pedido ao servidor e os resultados (ficheiros) são apresentados na secção 3 da figura 4.9. Assim, conseguimos aumentar a usabilidade do sistema, visto que neste caso o utilizador pode interagir com os resultados, da mesma forma que o faz no caso de pesquisas.

Importante ainda explicar que o utilizador, tal como no OSMOSIS-SFS, pode aceder a qualquer elemento da secção de resultados da interface principal. Para isso basta seleccionar o item pretendido e carregar no botão "Access Selected Item". O comportamento obtido, consoante o tipo de item, é o seguinte:

- **Ficheiro:** O ficheiro é transferido do OSMOSIS-SFS para o PDA e aberto com o programa pré-definido para abrir esse tipo de ficheiro.
- **Directoria:** É realizado um pedido ao servidor, de forma a obter o conteúdo da directoria que posteriormente é apresentado na lista de resultados.
- ***Tag:*** É realizada uma nova pesquisa, acrescentando aos termos introduzidos a *Tag* seleccionada.

No OSMOSIS-RFID é possível associar uma *Tag* RFID a uma localização física (existente no sistema). Para isso, basta carregar no botão "Associate Physical Location With RFID Tag" e é apresentada a interface da Figura 4.10. Aqui, o utilizador só tem de escolher a localização (1), passar o leitor pela *Tag* RFID pretendida e carregar no botão "Make Association". Note-se que antes de apresentar esta interface, o OSMOSIS-RFID tem de fazer um pedido ao servidor, de forma a obter a localizações físicas existentes.

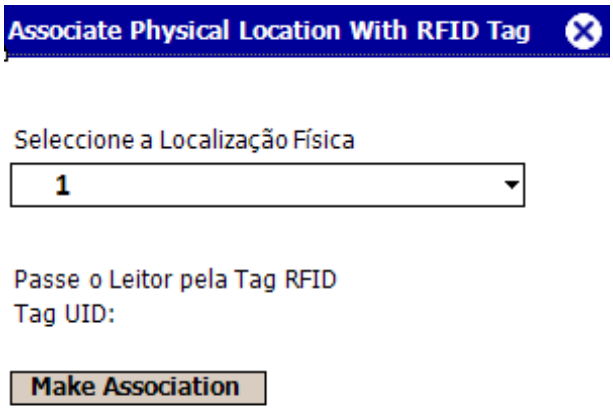

Figura 4.10: Interface para associar uma *Tag* RFID a uma localização física.

Para associar um ficheiro a uma *Tag* RFID, o utilizador deve seleccionar o ficheiro pretendido (da lista de resultados) e de seguida carregar no botão "Associate Selected File With RFID Tag". Depois destas acções, é apresentada a interface da Figura 4.11, para que o utilizador possa passar o leitor pela *Tag* RFID pretendida e fazer a associação.

Figura 4.11: Interface para associar um ficheiro a uma *Tag* RFID.

4.3.3 Cache do PDA

De forma a aumentar o desempenho do OSMOSIS-RFID, foi criada uma cache com resultados de pesquisas recentes. Esta, guarda para cada pesquisa um par <termos de pesquisa,resultados retornados> e a cada par está associado um tempo de vida. Este tempo de vida limitado (na ordem de poucos minutos) evita que a pesquisa no OSMOSIS-RFID fique obsoleta em relação à procura no OSMOSIS-SFS.

Cada vez que o utilizador pretende efectuar uma pesquisa, o sistema consulta a cache para verificar se existe uma entrada válida, dos termos a pesquisar, e em caso afirmativo apresenta imediatamente os resultados, sem fazer o pedido ao servidor. Se a entrada existir mas já estiver inválida, o sistema apaga essa entrada, faz um pedido ao servidor e apresenta os resultados retornados. Da mesma forma, se a entrada não existir em cache, é feito um pedido ao servidor e os resultados são posteriormente apresentados. Note-se que em qualquer dos casos que seja feito um pedido de pesquisa ao servidor, é sempre guardado em cache um par <termos de pesquisa,resultados retornados>.

Importante ainda referir que esta cache é limpa de cada vez que o OSMOSIS-RFID é reiniciado, evitando que fique repleta de entradas obsoletas.

4.3.4 Interface com o Servidor

Em todos os módulos descritos até este ponto foram feitas diversas referências a "pedidos ao servidor", não sendo explicada a forma como são realizados esses pedidos. É este módulo, `Interface com o Servidor`, que define todos os métodos necessários para satisfazer os vários pedidos ao servidor e por isso os outros módulos não precisam de ter conhecimento dos detalhes da comunicação. No entanto, ao contrário do OSMOSIS-SFS, que pode funcionar sem o módulo `Interface com o PDA` (Secção 4.1.7), no OSMOSIS-RFID o módulo `Interface com o Servidor` é essencial, visto que todas as funcionalidades fornecidas requerem pedidos ao servidor.

Na classe que implementa este módulo, estão definidos os métodos que satisfazem as seguintes funcionalidades: pedido de localização e ficheiros (*path*) associados a uma *Tag* RFID, pesquisa de ficheiros, localizações físicas existentes, associar uma *Tag* RFID a uma localização física, associar uma *Tag* RFID a um ficheiro, *download* de um ficheiro e obter o conteúdo de uma pasta. De um modo geral, todos os métodos começam por criar um *TcpClient*, depois enviam o pedido para o servidor através desse *TcpClient*, esperam pela resposta, quando é recebida efectuam o processamento da mensagem e por último retornam os resultados ao módulo que fez o pedido.

Os métodos responsáveis por satisfazer cada funcionalidade, do OSMOSIS-RFID, são os seguintes:

- `processaTagUID` - Este método é responsável por pedir ao servidor as localizações físicas e ficheiros associados a uma *Tag* RFID. Dependendo da resposta, se a *Tag* estiver atribuída a um localização física deve retornar a localização e os ficheiros que lhe estão associados, já se estiver atribuída a um ficheiro, retorna esse ficheiro.

- `processaPesquisa` - Método que envia um pedido de pesquisa (pelos termos introduzidos) ao servidor, retornando posteriormente uma lista com os resultados.

- `processaLocalizacoes` - É responsável por enviar um pedido ao servidor, de forma a obter todas as localizações físicas existentes no sistema. Retorna uma lista com as localizações.

- `processaAssociarLocalizacaoTagRFID` - Este método é responsável por pedir ao servidor que efectue uma associação entre a localização e o número de *Tag* transmitidos. Retorna um *boolean* com o resultado do sucesso ou falha na associação

- `processaAssociarFicheiroTagRFID` - Método que envia um pedido ao servidor para que este efectue uma associação entre a *path* do ficheiro e o número da *Tag* transmitidos. Retorna um *boolean* com o resultado do sucesso ou falha na associação.

- `processaDownloadFicheiro` - É responsável por pedir ao servidor que este lhe envie o ficheiro correspondente a um determinado *path*. Após receber o ficheiro, guarda-o numa pasta temporária do PDA.

- `processaConteudoPasta` - Este método pede ao servidor que lhe envie o conteúdo (ficheiros) de uma dada directoria e após receber a resposta retorna uma lista de ficheiros.

4.4 Síntese

Neste capítulo apresentámos os principais detalhes de implementação do sistema OSMOSIS. Como podemos verificar, o nosso sistema é constituído por três componentes principais: OSMOSIS-RFID, Interfaces para comunicação e OSMOSIS-RFID.

O OSMOSIS-SFS desenvolvido é um sistema de ficheiros semântico, que para além de permitir realizar as funcionalidades típicas (indexação e pesquisa de ficheiros) de um sistema deste tipo, também possibilita a atribuição manual de tags aos ficheiros, relacionar explicitamente dois ficheiros entre si, assim como a integração de objectos reais no mundo virtual. Ou seja, mais que um simples sistema de ficheiros semântico, o OSMOSIS-SFS guarda informação acerca de objectos físicos (relacionados com ficheiros virtuais) e posteriormente usa essa informação para ajudar a encontrar os objectos, ou até mesmo antecipar que ficheiros o utilizador poderá querer ver, analisando a sua localização física.

A interface para comunicação é um componente essencial do nosso sistema, uma vez que sem esta interface bem definida, a comunicação entre o OSMOSIS-SFS e o OSMOSIS-RFID seria impossível. Após análise das várias possibilidades para implementar este componente, decidimos optar pela solução que possui uma menor ocupação de memória e largura de banda. Assim, criámos um conjunto bem definido de mensagens de texto que são trocadas entre o OSMOSIS-SFS e OSMOSIS-RFID, através de *Sockets*, possibilitando o correcto funcionamento do sistema.

O terceiro componente apresentado neste capítulo foi o OSMOSIS-RFID. Este tem como principal função dar ao utilizador uma forma de poder navegar no mundo virtual, ao mesmo tempo que se desloca no mundo físico. Ou seja, para além de permitir realizar pesquisas de ficheiros, com este sistema é possível por exemplo ler a Tag RFID que está na porta da sala de estar e saber imediatamente que objectos (relacionados com ficheiros) estão no seu interior. No fundo, o OSMOSIS-RFID é um cliente do sistema OSMOSIS-SFS, que tem como principal objectivo apresentar informação ao utilizador e indicar ao sistema a localização física deste.

Capítulo 5

Avaliação do Sistema OSMOSIS

De forma a medir o desempenho e usabilidade do OSMOSIS, foram realizados um conjunto de testes à aplicação desenvolvida. Estes testes pretendem primeiramente verificar a usabilidade do sistema e posteriormente o desempenho do OSMOSIS-SFS. Por fim, foi também medido o desempenho da interacção entre o OSMOSIS-SFS e o OSMOSIS-RFID. Neste capítulo começa-se por explicar a análise qualitativa e posteriormente é apresentada a avaliação quantitativa de todo o sistema.

5.1 Avaliação Qualitativa

Um dos aspectos relevantes de testar no nosso sistema é a usabilidade, ou seja, quão úteis são as funcionalidades fornecidas pelo OSMOSIS. No fundo, pretendemos responder à pergunta: "As funcionalidades fornecidas pelo OSMOSIS são úteis, do ponto de vista do utilizador?".

Para testar a usabilidade do OSMOSIS pedimos a alguns utilizadores (num total de 8) para que testassem e comentassem cada funcionalidade do nosso sistema. Como era difícil testar a funcionalidade do OSMOSIS-RFID com utilizadores, na avaliação qualitativa apenas foi considerado o OSMOSIS-SFS. De qualquer forma, quando foram realizados os testes, todas as funcionalidades do OSMOSIS (OSMOSIS-SFS + OSMOSIS-RFID) foram explicadas ao utilizador, para que este pudesse ter uma visão do funcionamento global do sistema.

Antes de iniciar o teste, foi pedido a cada utilizador para que recolhesse um conjunto de ficheiros seus, do tipo .doc, .xls, .ppt, .pdf, .txt, .jpg, .mpg, .mp3. Após recolha destes ficheiros, foram colocados numa directoria completamente ao acaso (sem conhecimento por parte do utilizador) no PC de teste, e posteriormente foi realizada uma indexação dos mesmos. Com estes testes qualitativos, pretendemos mostrar que o OSMOSIS-SFS cumpre o principal objectivo para o qual foi desenvolvido, ou seja, recolha de informação semântica e pesquisa de ficheiros a partir dessa mesma informação. Importante ainda referir que no PC de teste foi utilizado o Microsoft Windows XP, onde não é realizada indexação de ficheiros por parte do sistema operativo.

De seguida apresenta-se o teste/questionário que foi realizado a cada um dos utilizadores.

1. Pesquisar o relatório da tese no OSMOSIS-SFS e no Microsoft Windows. Nesta pesquisa deverá ser utilizado o nome do ficheiro como palavra-chave.

2. Pesquisar o relatório da tese no OSMOSIS-SFS e no Microsoft Windows. Nesta pesquisa deverá ser utilizada informação semântica (por exemplo tema da tese) como palavra-chave.

3. Pesquisar uma música no OSMOSIS-SFS e no Microsoft Windows. Nesta pesquisa deverá ser utilizada informação semântica (por exemplo nome do álbum) como palavra-chave.

4. Pesquisar uma imagem no OSMOSIS-SFS e no Microsoft Windows. Nesta pesquisa deverá ser utilizada informação semântica (por exemplo modelo da câmara fotográfica) como palavra-chave.

5. Atribuir *Tags* manualmente a um conjunto de ficheiros e de seguida fazer uma pesquisa por essa *Tag*.

6. Realizar uma pesquisa que implique utilizar a funcionalidade de refinamento de pesquisa. Ou seja, utilizar as sugestões oferecidas pelo próprio sistema.

7. Primeiro relacionar explicitamente dois ficheiros entre si. De seguida, pesquisar por um deles e verificar se o outro também é apresentado como resultado.

8. Comentar de forma geral as funcionalidades oferecidas pela interface com o utilizador.

9. Comentar o sistema na sua globalidade.

Em relação à tarefa 1, verificámos que todos os utilizadores conseguiram encontrar o ficheiro da tese à primeira tentativa, e de forma instantânea (menos de um segundo), utilizando o OSMOSIS-SFS. Como não sabiam em que directoria se encontrava o ficheiro, tiveram de utilizar a pesquisa do Windows demorando em média mais de um minuto para encontrar o mesmo ficheiro.

Na tarefa 2, todos os utilizadores conseguiram encontrar o relatório da tese de forma instantânea (menos de um segundo), utilizando o OSMOSIS-SFS com informação semântica (por exemplo tema da tese). No entanto, utilizando a pesquisa do Microsoft Windows, esta procura revelou-se demasiado demorada.

As tarefas 3 e 4 foram também concluídas com êxito por todos os utilizadores, ou seja, as músicas e imagens foram encontradas com sucesso a partir do nome do álbum e modelo da câmara fotográfica, respectivamente, utilizando o OSMOSIS-SFS. Já utilizando a pesquisa do Windows, foi necessário mais de um minuto para encontrar estes mesmos ficheiros.

Em relação às tarefas 5, 6 e 7 podemos afirmar que todas estas foram concluídas com sucesso e que os resultados da pesquisa corresponderam aos interesses do utilizador. Note-se que antes de realizar estas tarefas, foi explicado a cada utilizador o modo como se coloca *Tags* manualmente nos ficheiros, como se utiliza o refinamento da pesquisa e como se podem relacionar explicitamente ficheiros entre si.

As tarefas 8 e 9 consistiam em fazer comentários à interface com o utilizador e ao sistema OSMOSIS. De uma forma geral os comentários foram bastante positivos, sendo que alguns utilizadores fizeram sugestões de melhoramento da interface (por exemplo “Em caso de paths compridos, introduzir "..."no meio da path”) e de inclusão de novas funcionalidades (por exemplo “Poder pesquisar no OSMOSIS-SFS os objectos que se encontram numa dada divisão”).

Analisando os resultados destes testes, podemos afirmar que de uma forma geral o nosso sistema cumpre os principais objectivos para os quais foi desenvolvido. No entanto, foram identificadas algumas limitações a nível da interface com o utilizador e melhoramentos a realizar nas funcionalidades existentes. Estas limitações da interface eram espectáveis visto que, tal foi referido anteriormente, não foi dada muita importância ao aspecto gráfico da interface mas sim às funcionalidades fornecidas. Em relação ao funcionamento do OSMOSIS, não foram feitas muitas sugestões por parte do utilizador, apenas pequenos melhoramentos, que na nossa opinião, vão permitir aumentar a usabilidade do sistema.

5.2 Avaliação Quantitativa

Após ter sido apresentada a avaliação qualitativa do OSMOSIS, iremos agora mostrar os resultados dos testes de desempenho realizados a este sistema. Através destes testes pretendemos evidenciar as vantagens

e limitações do sistema desenvolvido, tanto a nível da duração de tarefas críticas (por exemplo pesquisas) como ao nível de memória consumida. Começamos por mostrar os testes realizados ao OSMOSIS-SFS e posteriormente os resultados do OSMOSIS-RFID.

Importante ainda referir que estes testes foram realizados num PC cujas especificações são as seguintes:

- **Sistema Operativo:** Microsoft Windows XP Home Edition, Versão 2002, Service Pack 3.
- **Processador:** Intel Pentium M, 1.60GHz.
- **Memória RAM:** 1,25 GB.

O PDA onde se encontra instalado o OSMOSIS-RFID e que foi utilizado nos testes, tem as seguintes especificações:

- **Sistema Operativo:** Microsoft Windows CE .NET, Versão 4.20.
- **Processador:** ARMV4I-EX250.
- **Memória RAM:** 48,56 MB.

5.2.1 Desempenho do OSMOSIS-SFS

Para testar o desempenho do sistema OSMOSIS-SFS foram realizados três tipos de testes. O primeiro consiste em realizar indexações (extracção automática de *Tags*) completas sobre vários conjuntos de ficheiros, registando o tempo de cada indexação e a memória RAM ocupada durante esta tarefa. O segundo teste consiste em realizar várias pesquisas, utilizando o OSMOSIS-SFS e a pesquisa do Microsoft Windows, registando o tempo para encontrar o ficheiro pretendido. Por fim, foi também realizado um teste onde se pretende observar o tempo gasto em cada fase da pesquisa no OSMOSIS-SFS.

Teste de Indexações Completas

Tal como referido anteriormente, este teste consiste em realizar indexações (extracção automática de *Tags*) completas em vários conjuntos de ficheiros, registando o tempo da indexação e a memória RAM utilizada pela aplicação. Note-se que foram indexados 8 conjuntos de 50 ficheiros, do tipo .doc, .xls, .ppt, .pdf, .txt, .jpg, .mpg, .mp3. Através deste teste pretendemos identificar quais os tipos de ficheiros levam ao aumento do tempo de conclusão da indexação, e em quais é necessária uma maior quantidade de memória RAM.

De seguida são apresentados os gráficos que mostram os resultados obtidos com este teste. Na Figura 5.1 podemos observar o tempo total para a indexação de cada conjunto, assim como o tempo médio necessário para indexar um ficheiro. Já na Figura 5.2 é apresentada a memória RAM ocupada pelo sistema, durante a indexação de cada conjunto. Por fim, a Figura 5.3 mostra a variação do espaço ocupado em disco (pela informação semântica), consoante o número de ficheiros indexados.

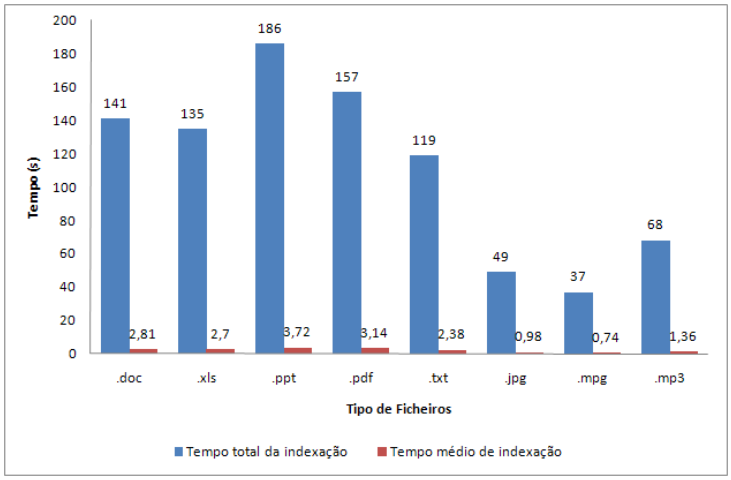

Figura 5.1: Tempos totais e médios na indexação de cada conjunto de ficheiros.

Como podemos observar na Figura 5.1, os ficheiros que levam mais tempo a ser indexados são os ficheiros de texto. Este resultado já era esperado visto que, neste tipo de ficheiros são extraídas *Tags* a partir dos metadados e do conteúdo dos ficheiros, enquanto que noutros tipos apenas são extraídas *Tags* a partir dos metadados.

Para além disso, podemos constatar que o tempo médio para a indexação de um ficheiro de texto é de cerca de três segundos, enquanto que para os restantes tipos, a indexação demora em média um segundo. Apesar de parecer demasiado tempo, podemos verificar que no contexto da nossa aplicação, este tempo é bastante aceitável. O OSMOSIS-SFS deve estar em funcionamento durante toda a sessão (de Windows) de um utilizador, e por isso sempre que um ficheiro é criado, é realizada uma indexação desse ficheiro. Podemos assim facilmente considerar que o intervalo de tempo desde que um ficheiro é criado até que o utilizador o queira pesquisar é superior a três segundos, e que por isso o tempo de indexação não afecta o desempenho global do sistema, podendo ser realizada em *background*.

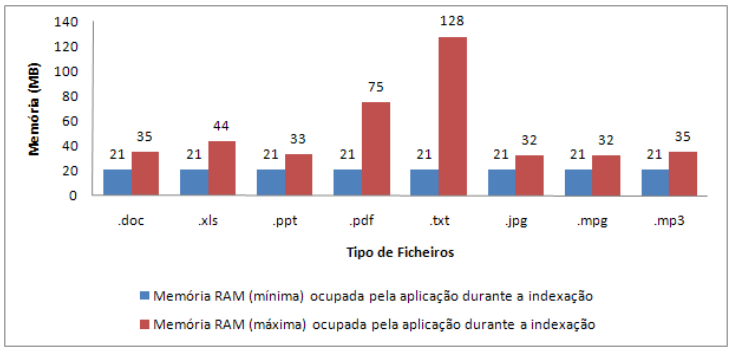

Figura 5.2: Memória RAM (mínima e máxima) ocupada na indexação de cada conjunto de ficheiros.

Observando a Figura 5.2, podemos verificar que, à excepção dos ficheiros .pdf e .txt, a indexação de todos os restantes tipos de ficheiros não consome muita memória RAM (cerca de 35 MB). A razão pela qual os ficheiros .pdf e .txt consomem uma grande quantidade de memória, tem a ver com o facto de nestes conjuntos existirem vários ficheiros com imenso texto (ficheiros .txt com mais de 6 MB).

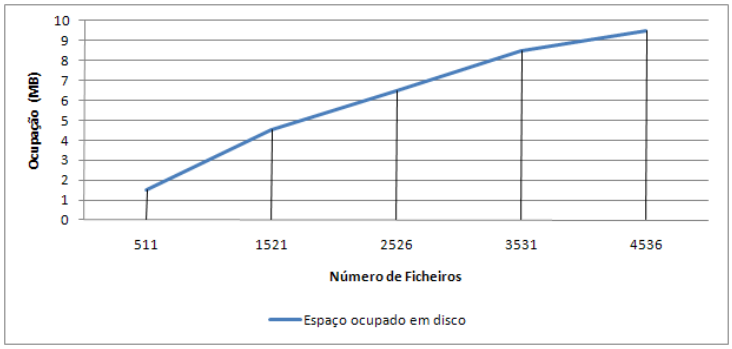

Figura 5.3: Variação do espaço ocupado em disco, pela informação semântica, consoante o número de ficheiros indexados.

Na Figura 5.3 podemos observar a variação do espaço ocupado pela informação semântica, consoante o número de ficheiros indexados. Facilmente podemos afirmar que a informação semântica de um ficheiro de texto (metadados + conteúdo), ocupa mais espaço em disco que a informação semântica de outro tipo de ficheiro (metadados). Este facto foi tido em conta neste teste, e por isso os 4536 ficheiros indexados estão distribuídos por cerca de 50% de ficheiros de texto e 50% de outro tipo de ficheiros. Importante ainda referir que estes 4536 ficheiros ocupam 9,8 GB em disco, existindo por isso uma compressão de cerca de 1000 vezes na indexação.

A partir destes dados, podemos calcular que um conjunto de 100.000 ficheiros ocupará cerca de 210 MB em disco. Se tivermos em conta a actual capacidade de memória física disponível nos PC's, este valor não afecta em nada o desempenho do computador, nem será perceptível ao utilizador.

Comparação entre a Pesquisa do OSMOSIS-SFS e a do Microsoft Windows

Este teste consiste em realizar várias pesquisas, utilizando o OSMOSIS-SFS e o Microsoft Windows, registando o tempo que cada um leva a encontrar o ficheiro pretendido. Note-se que em qualquer dos sistemas, supõe-se que o utilizador não sabe onde está guardado o ficheiro, e deste modo é sempre realizada uma pesquisa global ao sistema de ficheiros. Os ficheiros a encontrar são: "MAD aula pratica 1.pdf", "Plano de Negócios.doc", "Relatório Intermédio OSMOSIS-RFID.pdf", "Problema da Distribuicao do Calor.pdf" e "Apresentação Toyota Motor Manufacturing.ppt". Em ambos os sistemas, a procura foi realizada utilizando o nome do ficheiro como *keyword*. Com este teste, pretendemos verificar se o nosso sistema permite encontrar mais facilmente e rapidamente o ficheiro pretendido, em comparação com a pesquisa "tradicional" no Windows.

Na Figura 5.4 podemos observar o tempo total necessário para encontrar cada ficheiro a partir do OSMOSIS-SFS e do sistema de pesquisa do Windows.

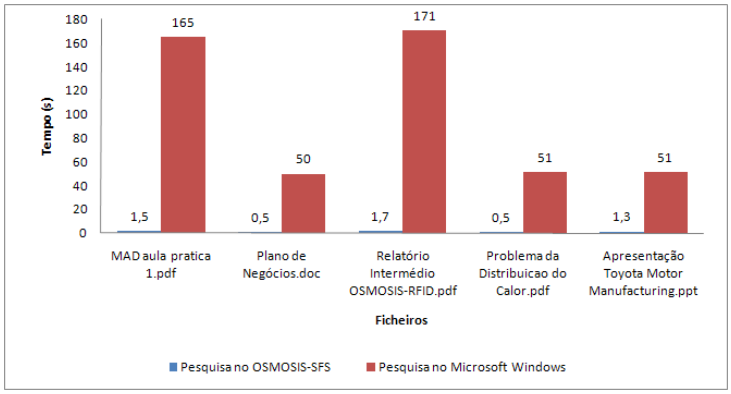

Figura 5.4: Tempo total para a pesquisa de cada ficheiro no OSMOSIS-SFS e no Microsoft Windows.

Como podemos observar a partir da Figura 5.4, o OSMOSIS-SFS demora em média pouco mais de 1 segundo a encontrar o ficheiro pretendido, enquanto que no Windows podemos esperar cerca de 1 minuto para obter os mesmos resultados. Este teste mostra-nos assim que, o nosso sistema tem um bom desempenho na pesquisa de ficheiros, sendo bastante mais rápido que a pesquisa no Microsoft Windows.

Tempo gasto em cada fase da pesquisa

Este teste consiste em realizar várias pesquisas no OSMOSIS-SFS, registando o tempo gasto em cada uma das fases (consulta da Base de Dados e ordenação de resultados + apresentação de resultados). Aqui, foram utilizadas *keywords* como "mad aula pratica", "rfid", "thesis", "relatório", "green day mp3", "ex z55 jpg", "state of the art", "mpg", "apresentação", "notas" e "sistema de ficheiros semântico". Como podemos notar, estas *keywords* foram escolhidas de forma utilizar o máximo de informação semântica possível (nomes de ficheiros, palavras mais usuais, metadados). Com este teste pretendemos identificar as fases da pesquisa onde é gasto a maioria do tempo, e logo as partes do sistema passíveis de optimização.

Na Figura 5.5, podemos verificar o tempo total, o tempo de consulta da Base de Dados e ordenação de resultados e o tempo de apresentação de resultados, obtidos em cada uma das pesquisas. Importante ainda referir que, para cada pesquisa foram retornados um diferente número de resultados como se mostra de seguida:

- "mad aula pratica" - retornou 100 resultados
- "rfid" - retornou 9 resultados
- "thesis" - retornou 3 resultados
- "relatório" - retornou 101 resultados
- "green day mp3" - retornou 100 resultados
- "ex z55 jpg" - retornou 100 resultados
- "state of the art" - retornou 105 resultados
- "mpg" - retornou 61 resultados
- "apresentação" - retornou 28 resultados
- "notas" - retornou 12 resultados
- "sistema de ficheiros semântico" - retornou 103 resultados

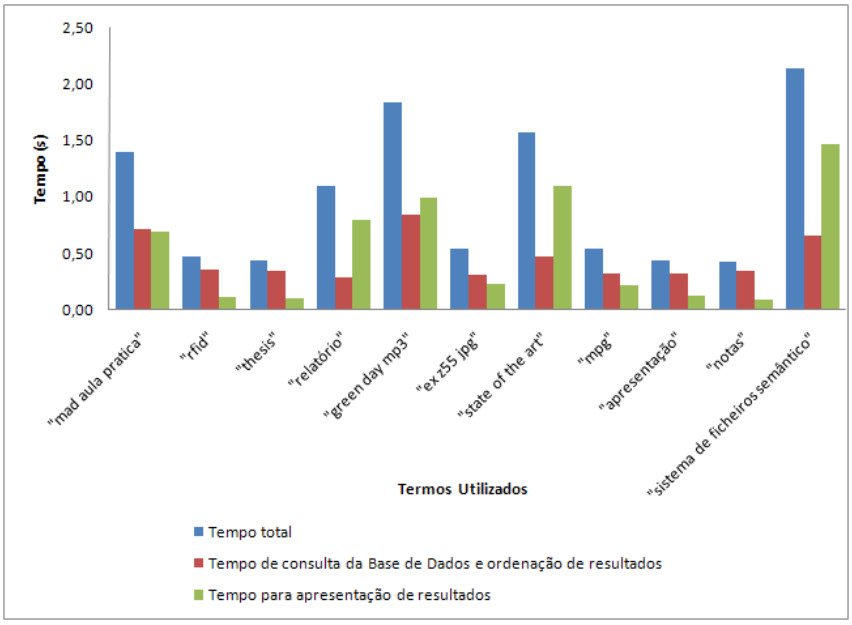

Figura 5.5: Tempos detalhados para cada pesquisa.

Analisando a Figura 5.5, podemos verificar que o tempo da fase de consulta da Base de Dados e ordenação de resultados, é menor que o tempo para a fase de apresentação, quando a pesquisa devolve um baixo número de resultados. Já quando são devolvidos um grande número de ficheiros, estes tempo são parecidos, chegando mesmo a inverter-se em certos casos.

Na Figura 5.6 estão apresentados os tempo médios das pesquisas. Como podemos observar, o tempo total é, em média, 1 segundo, o tempo de consulta da Base de Dados e ordenação de resultados é de 0,45 segundos e o tempo de apresentação é de cerca de 0,55 segundos. Estes resultados mostram que o algoritmo de pesquisa desenvolvido é bastante rápido (0,5 segundos), mas que é gasto o mesmo tempo apenas para apresentar os resultados. Este facto, deve-se à utilização do componente *ListView* da *framework* .NET e possivelmente teremos de encontrar uma alternativa para apresentar os resultados, de forma a diminuir este tempo.

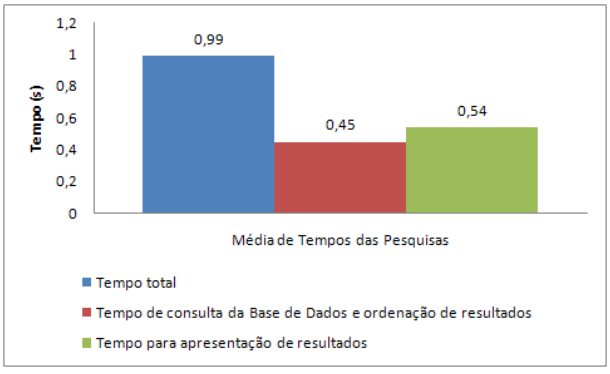

Figura 5.6: Médias dos tempos registados para cada fase das pesquisas.

5.2.2 Desempenho do OSMOSIS-RFID

De forma a testar o desempenho do sistema OSMOSIS-RFID, foram realizados dois tipos de testes. O primeiro consiste em utilizar todas as funcionalidades existentes no PDA (todas fazem pedidos ao Servidor), registando o tempo total para a conclusão de cada uma. O segundo consiste em efectuar duas

pesquisas pelos mesmos termos, registando os tempos de cada uma. Este último teste pretende medir o desempenho da cache de pesquisas.

Tempos de conclusão de cada funcionalidade

Tal como referido anteriormente, este teste consiste em medir o tempo de conclusão de cada funcionalidade do OSMOSIS-RFID. Note-se que as funcionalidades a testar são: Ler uma *Tag* RFID associada à localização "Sala de Estar", Ler uma *Tag* RFID associada ao ficheiro "C:\testes\pasta\texto.txt", Pesquisa pelos termos "mad aula pratica" (sem cache), Pesquisa pelo termo "thesis" (sem cache), associar uma localização física a uma *Tag* RFID e associar um ficheiro a uma *Tag* RFID. Com este teste pretendemos medir o desempenho global do sistema OSMOSIS-RFID.

Na Figura 5.7 podemos observar o tempo total, necessário para a conclusão de cada funcionalidade do sistema OSMOSIS-RFID. Importante referir que, como o sistema operativo do PDA não suporta o tempo em milisegundos, estes tempos registados são apenas em segundos.

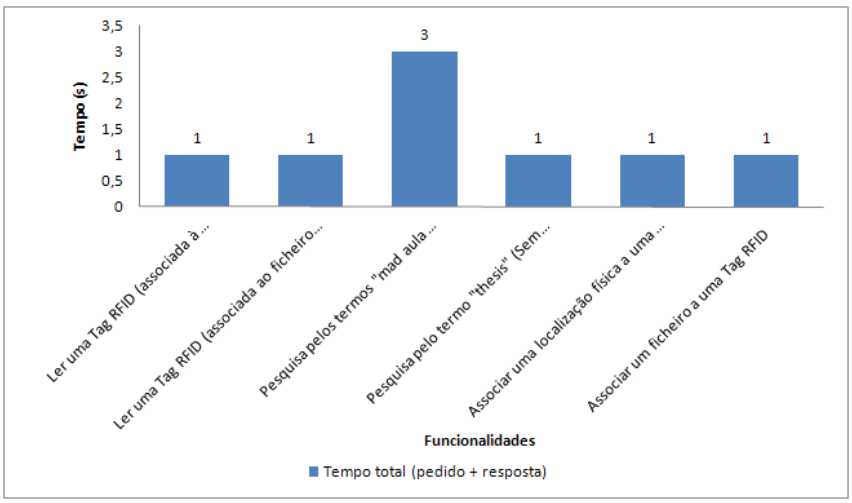

Figura 5.7: Tempo total (pedido + resposta) para a conclusão de cada funcionalidade.

Como podemos observar a partir da Figura 5.7, o OSMOSIS-RFID demora em média 1 segundo para concluir cada funcionalidade fornecida. Apenas a pesquisa pelos termos "mad aula pratica" demora 3 segundos a ser concluída. Esta situação pode ser facilmente explicada pelo facto da mesma pesquisa demorar 1,5 segundos no próprio OSMOSIS-SFS, como se pode observar na Figura 5.5. Importante ainda referir que, estes tempos medidos para cada funcionalidade incluem todas as seguintes fases:

1. Construção do pedido por parte do PDA
2. Envio do pedido ao servidor
3. Análise do pedido por parte do servidor
4. Processamento do pedido
5. Construção da resposta
6. Envio da resposta ao PDA
7. Análise da resposta por parte do PDA
8. Apresentação de resultados

Assim, como podemos verificar, 1 segundo é um valor bastante razoável para a realização de uma funcionalidade do OSMOSIS-RFID.

Tempo de pesquisa com e sem cache

Este último teste consiste em medir o tempo de uma pesquisa, sem recorrer à cache, e compará-lo com o tempo da mesma pesquisa, mas que utiliza os dados guardados em cache. Os termos utilizados nas pesquisas foram "mad aula pratica", "thesis", "apresentação" e "sistema de ficheiros semântico". Este teste pretende medir o benefício de uma cache, com resultados de pesquisas anteriores, no OSMOSIS-RFID.

Em baixo, na Figura 5.8, podemos observar o tempo total de cada pesquisa realizada no sistema OSMOSIS-RFID. Importante ainda referir que, para cada pesquisa foram retornados um diferente número de resultados como se mostra de seguida:

- "mad aula pratica" - retornou 100 resultados
- "thesis" - retornou 3 resultados
- "apresentação" - retornou 28 resultados
- "sistema de ficheiros semântico" - retornou 103 resultados

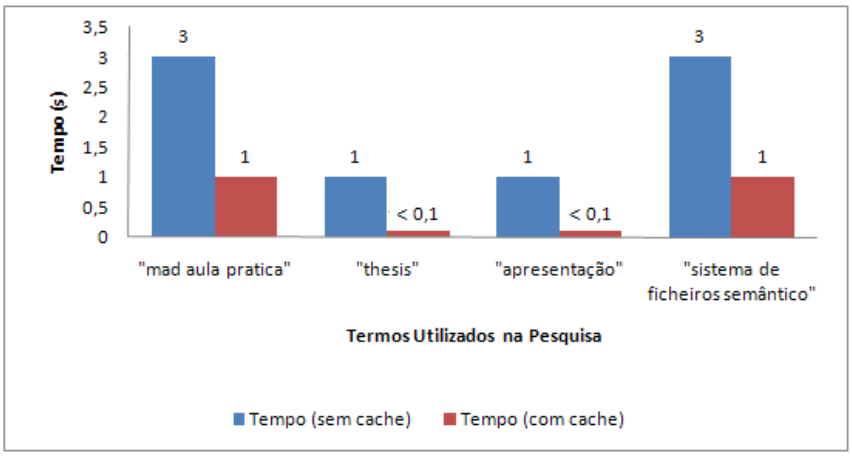

Figura 5.8: Tempo de pesquisa com e sem cache.

Analisando a Figura 5.8, podemos constatar que existe um decréscimo bastante significativo no tempo da pesquisa, quando esta é realizada pela segunda vez (recorrendo aos dados guardados). Este resultado era já expectável, visto que no caso em que são utilizados os dados em cache, o OSMOSIS-RFID apenas apresenta directamente os resultados ao utilizador.

Como também podemos verificar, no caso das pesquisas (com cache) por "mad aula pratica" e "sistema de ficheiros semântico", foi necessário 1 segundo, apenas para apresentar os resultados. Mais uma vez, é possível que a utilização do componente *ListView* da *framework* .NET, seja a razão pela qual é gasto muito tempo na apresentação de resultados.

5.3 Síntese

Neste capítulo descrevemos os testes que foram realizados ao OSMOSIS de forma a medir a sua usabilidade e desempenho. Começámos por apresentar os testes qualitativos e de seguida foram analisados os testes quantitativos realizados à nossa aplicação.

Os testes qualitativos pretendem aferir a usabilidade geral, ou seja, quão úteis são as funcionalidades fornecidas pelo OSMOSIS. Para estes testes, pedimos a vários utilizadores que realizassem um conjunto de tarefas no nosso sistema e no final registámos os comentários de cada um. Os resultados mostram que de uma forma geral o nosso sistema cumpre os principais objectivos para os quais foi desenvolvido, no entanto, foram identificadas algumas limitações a nível da interface com o utilizador e melhoramentos a realizar nas funcionalidades existentes.

Por fim, realizámos um conjunto de testes quantitativos de forma a medir o desempenho do OSMOSIS, ou seja, a duração das tarefas críticas e a memória consumida pela aplicação. Com base nos resultados obtidos para o OSMOSIS-SFS, podemos concluir que o tempo de indexação de um ficheiro não afecta o desempenho global do sistema, que não existe um elevado consumo de memória RAM, que o espaço ocupado em disco por informação semântica não afecta o desempenho do computador e que a pesquisa no nosso sistema é bastante mais rápida que a pesquisa "tradicional" do Microsoft Windows. Já analisando os resultados obtidos para o sistema OSMOSIS-RFID, podemos afirmar que 1 segundo é um valor bastante razoável para a realização de uma funcionalidade deste sistema e que a utilização de uma cache aumenta bastante o seu desempenho nas pesquisas.

Capítulo 6

Conclusões e Trabalho Futuro

6.1 Conclusões

Com o aumento da capacidade de armazenamento dos computadores, passou a ser possível guardar uma enorme quantidade de ficheiros e com isso, a necessidade por um sistema de ficheiros mais flexível, tornou-se evidente.

Nesta dissertação abordámos o tema dos sistemas de ficheiros semânticos e de que forma estes nos podem ser úteis para interacção com o nosso próprio PC. Para além disso, discutimos, implementámos e avaliámos um sistema de ficheiros semântico (OSMOSIS-SFS), assim como um sistema (OSMOSIS-RFID) que permite fazer a integração de objectos reais no mundo virtual. Estes dois "sub-sistemas" fazem parte do objectivo global do nosso trabalho, o sistema OSMOSIS.

Os sistemas de ficheiros semânticos vêm tornar a utilização do sistema de ficheiros, uma tarefa menos rígida e mais ligada com o modo como os utilizadores pensam no mundo real. Na verdade, é muito mais fácil para um utilizador lembrar-se de partes do nome do ficheiro, ou até mesmo do assunto desse ficheiro, do que o *path* completo para esse mesmo documento. Neste tipo de sistemas, os ficheiros podem ser resumidos a um conjunto de palavras-chave, que estão associadas a cada um, e que podem ser facilmente lembradas pelo utilizador. Note-se que, a existência de uma interface gráfica que permita fazer uso da informação semântica recolhida, é bastante importante neste tipo de sistemas.

No sistema OSMOSIS-SFS são associados atributos (*Tags*) aos ficheiros de modo a ser possível aceder a um em específico por mais de um caminho, ao contrário do que acontece nos sistemas de ficheiros hierárquicos. A associação de atributos é realizada de forma automática e manual. No primeiro caso, o sistema extrai atributos a partir dos metadados, de todos os ficheiros, e até mesmo do conteúdo dos ficheiros de texto. Já no caso da associação manual, o utilizador é o responsável por colocar atributos nos ficheiros, seja com uma relação explícita em dois ficheiros, ou mesmo atribuindo-lhes palavras-chave (por exemplo "tese"). Por esta razão, não se pode afirmar que este sistema se insere apenas numa das categorias descritas no capítulo 2, em vez disso, podemos considerar que faz parte de todas. Esta conclusão deriva do facto do OSMOSIS-SFS apresentado, possuir características de todas as categorias.

Mais que um simples sistema de ficheiros semântico, o OSMOSIS funciona como um localizador de objectos físicos (relacionados com ficheiros virtuais). O nosso sistema permite associar *Tags* RFID a localizações ou objectos físicos e posteriormente usar essa informação para ajudar a encontrar os objectos. Este sistema possibilita também, antecipar que ficheiros o utilizador poderá querer ver, analisando a sua localização. Importante ainda referir que, esta integração com o mundo real é realizada, recorrendo a um PDA equipado com um leitor RFID e onde está instalado o sistema OSMOSIS-RFID.

Contudo, ao longo do desenvolvimento do nosso sistema, encontrámos algumas dificuldades das quais podemos destacar:

- Decidir qual o tipo de informação semântica que seria guardada para cada tipo de ficheiros.
- Elaboração do algoritmo de extracção automática de *Tags* a partir de ficheiros de texto.
- Construção de um motor de pesquisa semântico, que permitisse fazer uso, de forma eficiente, da informação semântica recolhida.
- Decidir como seria realizada a comunicação entre o OSMOSIS-SFS e OSMOSIS-RFID, ou seja, elaborar as interfaces de comunicação.

Podemos afirmar que como pontos fortes, o sistema OSMOSIS permite adicionar facilmente *plug-ins*, de forma a conseguir extrair *Tags* de outros tipos de ficheiros de texto. Possui interfaces bem definidas, o que faz com que os detalhes de comunicação, de cada sub-sistema (OSMOSIS-SFS e OSMOSIS-RFID), estejam concentrados num só módulo. O OSMOSIS-RFID possui uma cache de pesquisas recentes, o que permite aumentar o desempenho do sistema. E por fim, o nosso sistema utiliza o conceito de que, atributos são sempre simples palavras associadas aos ficheiros.

Quanto a pontos fracos, podemos afirmar que o OSMOSIS-SFS utiliza muito poder de processamento para extrair automaticamente *Tags*, quando são criados muitos ficheiros de uma só vez (por exemplo numa cópia), podendo fazer com que o PC fique "lento" durante alguns segundos. Todas as *Tags* têm a mesma importância aquando da pesquisa, ou seja, não é dada maior importância às *Tags*, que por exemplo são pesquisadas mais vezes. E por fim, não permite detectar automaticamente quando um objecto muda de localização física, ou seja, deve ser o utilizador a fazer esta actualização no sistema.

Na fase final de elaboração do sistema OSMOSIS, foram realizados um conjunto de testes à aplicação desenvolvida, de forma a medir o desempenho e usabilidade do sistema. Os testes qualitativos mostraram que, de uma forma geral, o nosso sistema cumpre os principais objectivos para o qual foi desenvolvido, ou seja, recolha de informação semântica, pesquisa de ficheiros a partir dessa mesma informação e integração de objectos físicos no mundo virtual. Para além de testes qualitativos, foram também realizados um conjunto de testes quantitativos, de forma a medir o desempenho do sistema. Estes testes mostraram que o tempo médio para a indexação de um ficheiro de texto é de cerca de três segundos, enquanto que para os restantes tipos, a indexação demora em média um segundo. Como podemos facilmente considerar que o intervalo de tempo, desde que um ficheiro é criado até que o utilizador o queira pesquisar, é superior a três segundos, estes resultados são bastante aceitáveis e por isso o tempo de indexação não influencia o desempenho global do sistema. Com estes testes conseguimos comprovar que a informação semântica dos ficheiros não ocupa muito espaço em disco, já que a indexação de 4536 ficheiros ocupa cerca de 9,5 MB. E com a actual capacidade dos computadores, esta ocupação de memória física não afecta em nada o desempenho do PC. Mostrámos também que o OSMOSIS-SFS tem um bom desempenho na pesquisa de ficheiros, sendo bastante mais rápido que a pesquisa "tradicional" no Microsoft Windows. Em relação ao sistema OSMOSIS-RFID, podemos afirmar que o tempo registado (cerca de 1 segundo) para a conclusão de cada uma das funcionalidades é bastante razoável, dado a quantidade de fases (tarefas) de cada uma. Por fim, mostrámos o benefício do uso de uma cache, com resultados de pesquisas anteriores, no OSMOSIS-RFID.

No final do trabalho, podemos afirmar que conseguimos desenvolver e testar um sistema de ficheiros semântico, que funciona numa camada acima do sistema de ficheiros nativo do Windows. Ou seja, continuando a manter a forma como os utilizadores navegam, tradicionalmente, entre ficheiros, fornecemos uma forma de explorar semanticamente o sistema de ficheiros. Para além disso, conseguimos implementar e testar um sistema que permite associar *Tags* RFID a localizações ou objectos físicos (relacionados com ficheiros virtuais) e posteriormente usar essa informação para ajudar a encontrar os objectos ou até mesmo para visualizar o ficheiro directamente relacionado com o objecto.

6.2 Trabalho Futuro

A solução desenvolvida, apesar de ter apresentado resultados (qualitativos e quantitativos) bastante positivos, pode ser sempre alvo de melhorias. Estas podem ir desde optimizações ao sistema actual até à adição de novas funcionalidades, das quais podemos destacar:

- Evitar que quando são criados muitos ficheiros de uma só vez, estes sejam indexados todos de seguida, ou seja, provocando a sensação de que o PC está "lento". Para resolver este problema, podemos por exemplo deixar um pequeno intervalo de tempo entre a indexação de dois ficheiros, ou menos colocar a prioridade do processo com o nível "Baixo", durante a indexação.

- Actualmente, o nosso sistema não atribui diferentes níveis de importância às *Tags* de cada ficheiro, o que seria útil para o motor de pesquisa. No fundo, conseguiríamos obter uma melhor ordenação dos resultados de uma procura. Neste sentido, poderíamos por exemplo atribuir uma maior importância às *Tags*, pelas quais são realizadas um maior número de procuras, e até mesmo implementar um género de *garbage collector* para *Tags*, extraídas automaticamente, e das quais não são realizadas pesquisas durante um longo período de tempo.

- Como no nosso sistema o utilizador tem de avisar a base de dados sempre que um objecto muda de localização física, esta pode ficar mais facilmente desactualizada, simplesmente porque o utilizador esquece de registar a mudança de localização. Por esta razão, o OSMOSIS poderá detectar automaticamente quando um objecto muda de localização física, utilizando para o efeito, a informação proveniente, por exemplo, de antenas RFID colocadas à entrada de cada divisão de uma casa. Com esta funcionalidade seria então possível manter a base de dados sempre actualizada.

- Presentemente, a interface gráfica desenvolvida fornece as funcionalidades pretendidas para o sistema, contudo o seu aspecto gráfico poderia ser melhorado. Por isso, e de forma a aumentar a usabilidade, o aspecto gráfico deve ser melhorado, devem ser adicionadas novas formas de visualizar os resultados de pesquisas (por exemplo apresentando só o nome do ficheiro em vez do *path* completo) e as "ajudas", como por exemplo *Tags* para refinamento de pesquisa, devem estar melhor identificadas. Assim, a interface torna-se mais intuitiva e apelativa para o utilizador.

- Tirar partido do contexto de utilização dos ficheiros, como fonte de informação semântica, ou seja, para além de o utilizador poder relacionar explicitamente ficheiros, o sistema pode, por exemplo, colocar automaticamente este tipo de relação quando detecta um padrão de acesso aos ficheiros.

- Dar a possibilidade de definir relações de inclusão e equivalência entre *Tags* atribuídas manualmente, de forma a aumentar a usabilidade do sistema. Neste caso, o utilizador seria o responsável por colocar as relações de inclusão entre *Tags*, e por exemplo podia indicar ao sistema que a *Tag* "Música" inclui as *Tags* "Pop" e "Rock". Assim, quando fosse realizada uma pesquisa pelo termo "Música", seriam retornados todos os ficheiros com a *Tag* "Música", "Pop" e "Rock". Já as relações de equivalência poderiam ser inferidas automaticamente pelo sistema, e por exemplo, se o utilizador atribuísse a *Tag* "Férias 2008" a um conjunto de ficheiros, e a *Tag* "2008 Férias" a um outro conjunto, o sistema deveria colocar uma relação de equivalência entre estas duas *Tags*. Deste modo, no caso de uma pesquisa por "Férias 2008" ou por "2008 Férias" seriam retornados sempre os dois conjuntos de ficheiros.

- Adaptar a arquitectura do OSMOSIS de modo a poder ser utilizada num ambiente empresarial, por exemplo, no escritório de uma seguradora. Devido à grande quantidade de informação existente

neste tipo de ambientes, o nosso sistema iria permitir uma maior facilidade na procura de documentos, tanto a nível virtual (através do OSMOSIS-SFS) como a nível físico (através do OSMOSIS-RFID).

Bibliografia

[1] B. Adrian, L. Sauermann, and T. Roth-Berghofer. Contag: A semantic tag recommendation system. In *Proceedings of I-Semantics*, volume 7, pages 297–304. JUCS, 2007.

[2] A. Ames, C. Maltzahn, N. Bobb, E. L. Miller, S. A. Brandt, A. Neeman, A. Hiatt, and D. Tuteja. Richer file system metadata using links and attributes. In *Proceedings of the 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies*, pages 49–60, Apr. 2005.

[3] Apple. Apple Spotlight. Website. `http://developer.apple.com/macosx/spotlight.html`.

[4] Beagle. Website. `http://beagle-project.org/Architecture_Overview`.

[5] S. Bloehdorn, O. Gorlitz, S. Schenk, and M. Volkel. TagFS-Tag semantics for hierarchical file systems. In *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06)*, pages 6–8, 2006.

[6] C. M. Bowman, C. Dharap, M. Baruah, B. Camargo, and S. Potti. *A File System for Information Management*. Pennsylvania State University, Dept. of Computer Science and Engineering, College of Engineering, 1994.

[7] S. Fertig, E. Freeman, and D. Gelernter. Lifestreams: an alternative to the desktop metaphor. In *CHI '96: Conference companion on Human factors in computing systems*, pages 410–411, New York, NY, USA, 1996. ACM.

[8] FileSystemWatcher. Website. `http://msdn.microsoft.com/en-us/library/system.io.filesystemwatcher(VS.80).aspx`.

[9] FUSE:. Filesystem in Userspace. Website. `http://fuse.sourceforge.net/`.

[10] D. K. Gifford, P. Jouvelot, M. A. Sheldon, and J. W. O. Jr. Semantic file systems. *ACM SIGOPS Operating Systems Review*, 25(5):16–25, 1991.

[11] O. Gorter. Database file system - alternative to hierarchy based file systems. *University of Twente, the Netherlands*, 2004.

[12] D. R. Hardy and M. F. Schwartz. Essence: A resource discovery system based on semantic file indexing. In *Winter USENIX Technical Conference*, pages 361–374, 1993.

[13] IFS. Installable File System Kit. Website. `http://www.microsoft.com/whdc/devtools/ifskit/default.mspx`.

[14] D. Ingram. Insight: A semantic file system.

[15] A. W. Leung, A. Parker-Wood, and E. L. Miller. Copernicus: A scalable, High-Performance semantic file system. Technical Report UCSC-SSRC-09-06, University of California, Santa Cruz, Oct. 2009.

[16] M. Mahalingam, C. Tang, and Z. Xu. Towards a semantic, deep archival file system. In *FTDCS '03: Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*, page 115, Washington, DC, USA, 2003. IEEE Computer Society.

[17] U. Manber and S. Wu. GLIMPSE: a tool to search through entire file systems. In *Proceedings of the USENIX Winter 1994 Technical Conference on USENIX Winter 1994 Technical Conference*, pages 4–4, Berkeley, CA, USA, 1994. USENIX Association.

[18] C. Marshall. Birch: A metadata search file system.

[19] P. Mohan, V. S, and D. A. Siromoney. Semantic file retrieval in file systems using virtual directories. In *13th Annual IEEE International Conference on High Performance Computing*, 2006.

[20] MySQL. Website. `http://www.mysql.com/`.

[21] .Net Remoting. Website. `http://msdn.microsoft.com/en-us/library/system.runtime.remoting%28VS.80%29.aspx`.

[22] H. B. Ngo, C. Bac, F. Silber-Chaussumier, and T. Q. Le. Towards ontology-based semantic file systems. In *Research, Innovation and Vision for the Future, 2007 IEEE International Conference on*, pages 8–13, 2007.

[23] Oxygen. Website. `http://oxygen.lcs.mit.edu/KnowledgeAccess.html#web`.

[24] B. Schandl. SemDAV: a file exchange protocol for the semantic desktop. In *2nd Semantic Desktop and Social Semantic Collaboration Workshop at the ISWC 2006*, Athens, GA, USA, Nov. 2006.

[25] B. Schandl, S. Pomajbik, D. Todorov, and A. Amiri. Integrating file systems and the semantic web. In *Demo at the 4th European Semantic Web Conference*, Innsbruck, June 2007.

[26] C. A. N. Soules and G. R. Ganger. Toward automatic context-based attribute assignment for semantic file systems. *Parallel data laboratory, Carnegie Mellon University. June*, 2004.

[27] C. A. N. Soules and G. R. Ganger. Connections: using context to enhance file search. *ACM SIGOPS Operating Systems Review*, 39(5):119–132, 2005.

[28] Tagsistant. Website. `http://www.tagsistant.net/`.

[29] Z. Xu, M. Karlsson, C. Tang, and C. Karamanolis. Towards a semantic-aware file store. In *Proceedings of the 9th conference on Hot Topics in Operating Systems- Volume 9*, pages 31–31, Berkeley, CA, USA, 2003. USENIX Association.