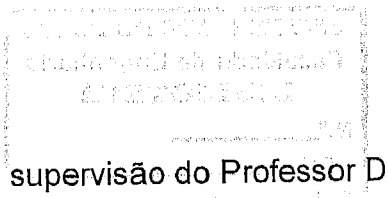
FACULDADE DE ENGENHARIA DE UNIVERSIDADE DO PORTO

Serviço de revisão bibliográfica colaborativa

Rui Humberto Ribeiro Pereira

Licenciado em Engenharia Electrotécnica pela Faculdade de Engenharia da Universidade do Porto

Dissertação submetida para satisfação parcial dos requisitos do grau de mestre em Tecnologias Multimédia



Dissertação realizada sob a supervisão do Professor Doutor Gabriel Torcato David
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Resumo

As comunidades de investigação espalhadas pelo planeta, nas suas actividades, produzem quantidades consideráveis de conhecimento. A forma mais generalizada de o comunicar é através de publicação de artigos, livros e relatórios, etc.. Estes documentos encontram-se armazenados em bibliotecas, e, cada vez mais, na própria *Internet*, o que lhes aumenta a acessibilidade. No entanto, a grande quantidade de informação disponível pode constituir um obstáculo à sua transformação em conhecimento. É portanto relevante a implementação de mecanismos que facilitem essa transformação.

Nesta dissertação faz-se uma proposta de criação de um novo serviço de informação. Trata-se de um serviço de revisão bibliográfica colaborativa, que auxilia os investigadores nessa tarefa bem como a registar os seus resultados de forma a facilitar posteriores revisões de terceiros.

O modelo desenvolvido assenta num fundo documental acessível através de um sistema de gestão de bibliotecas comercial, sobre o qual se estabelece uma "teia" de ligações semanticamente ricas que registam percursos preferenciais de exploração desse fundo, segundo múltiplos critérios, bem como anotações aos documentos e aos próprios percursos.

Apresenta-se um protótipo que implementa as ideias essenciais do modelo e que, no seu nível mais básico, constitui um interface de acesso a um repositório de documentos em formato electrónico. Assentes neste nível existem dois outros, vocacionados para o registo de informação acrescentada pelos utilizadores do serviço. Os resultados obtidos do protótipo ilustram a viabilidade do modelo.

Abstract

The investigation communities spread all over the planet create considerable amounts of knowledge while developing their activities. The most known way of communicating this knowledge is through papers publishing, books and reports, among others. These documents are stored in libraries and even more nowadays, in the Internet, allowing people to access up dated information more efficiently than ever. However, this great amount of available information can represent an obstacle to its transformation in knowledge. It is therefore relevant the implementation of mechanisms that enable this transformation.

In this dissertation a proposal for creation of a new information service is presented. It concerns a service of collaborative bibliographic revision, which supports the investigators in this specific task as well as in registering its results, thus providing future revisions by third party.

The developed model takes its grounds in a documental background accessible through a commercial libraries management system, under which a "web" of semantically enriched connections is established and that register preferential paths for exploring this background, according to multiple criteria as well as notes to documents and paths themselves.

A prototype is presented it implements the fundamental ideas of the model, which at its basic level represents an access interface to a repository of documents in electronic form. Based on this level there are two others, focussed on the registration of information added by the users of the service. The results emerged from this prototype clearly show the viability of the model.

Résumé

Les communautés de recherche disséminées sur la planète, dans leurs activités, produisent des quantités considérables de connaissances. La forme la plus généralisée de les communiquer est la publication d'articles, livres, rapports, etc. Ces documents sont emmagasinés en bibliothèques et, de plus en plus, à l'*Internet*, ce qui augmente leur accessibilité. Cependant, la grande quantité d'information disponible peut constituer un obstacle à sa transformation en connaissance. Le développement de mécanismes susceptibles de rendre cette transformation plus facile est donc important.

Dans la présente dissertation nous proposons la création d'un nouveau service d'information. Il s'agit d'un service de recherche bibliographique collaborative aidant les chercheurs non seulement dans cette tâche, mais aussi à faire l'enregistrement des résultats de façon à rendre des recherches postérieures plus faciles.

Le modèle développé est basé sur un fonds documentaire accessible par l'intermédiaire d'un système commercial de gestion de bibliothèques, sur lequel s'établit un réseau de liaisons sémantiquement riches, qui enregistre des parcours préférentiels d'exploitation de ce fonds à partir de multiples critères, aussi bien que des annotations sur les documents ou les parcours eux-mêmes.

Nous présentons un prototype qui développe les idées essentielles du modèle et qui, dans son niveau le plus élémentaire, constitue une interface d'accès à un répertoire de documents en format electronique. Basés sur ce niveau il y en a deux autres, destinés à l'enregistrement de l'information ajoutée par les utilisateurs du service. Les résultats obtenus à partir du prototype démontrent la viabilité du modèle.

Índice

Índice de figuras

Índice de tabelas

Capítulo 0 - Introdução

O recurso a citações de outros trabalhos é, regra geral, a forma utilizada para sustentar afirmações ou pressupostos efectuados num trabalho científico. Se todos os documentos envolvidos estiverem armazenados num sistema computacional, estas referências bibliográficas podem adquirir a dimensão de hiperligações entre documentos electrónicos, cumprindo a mesma função que de há muito têm vindo a desempenhar no papel, mas com a possibilidade acrescida de instantaneamente obter o documento citado. Tais referências facilitam também a navegação no tempo, para trás nos documentos citados e para a frente, nos documentos que citam o corrente, tornando-se num poderoso mecanismo de pesquisa bibliográfica. Vários estudos, projectos e até aplicações comerciais, para a utilização das citações nos documentos, têm sido elaborados como é o caso dos dois projectos abordados na presente dissertação o *Open Journal* (ver em 1.2.2) e o seu sucessor o *OpCit* (ver em 1.2.3), e uma aplicação comercial do *Institute* for *Scientific Information*[1], que disponibiliza um vasto conjunto de bases de dados de citações.

As hiperligações, quando geradas automaticamente a partir das citações introduzidas pelos autores nos seus artigos, permitem a definição de um grafo de precedências ligando todos os trabalhos relevantes para uma dada área, mas não especificam de uma forma totalmente objectiva e explícita o sentido da evolução do trabalho e da argumentação. Tal objectivo pode ser aproximado pela utilização de uma ontologia que permita classificar as relações entre documentos e os conceitos neles presentes.

A estrutura conceptual dos sistemas vocacionados para lidar com grandes quantidades de informação fortemente hiperligada, em ambientes cooperativos ou mesmo de ensino e aprendizagem, devem permitir ao leitor tornar-se autor, "tocar no texto" acrescentando o seu ponto de vista ou crítica.

A *Internet* é uma rede de computadores com uma cobertura a nível mundial. A esta rede estão ligadas praticamente todas as instituições onde há trabalho de investigação, pelo que, é utilizada como um meio de partilha e intercâmbio de informação, entre todos os que a ela

[1] http://www.isinet.com

têm acesso. Constitui portanto o meio ideal para um serviço de hiperligações baseadas em referências que se pretenda abrangente.

0.1 Objectivos da presente dissertação

Nesta dissertação faz-se uma proposta de criação de um novo serviço de informação. Trata-se de um serviço de revisão bibliográfica colaborativa, que auxilia os investigadores nessa tarefa bem como a registar os seus resultados de forma a facilitar posteriores revisões de terceiros.

O modelo que é proposto para o serviço, nesta dissertação, e o protótipo que é descrito, pretendem alcançar os benefícios atrás referidos, sobre um fundo documental acessível através de um sistema de gestão de bibliotecas comercial, estabelecendo uma "teia" de ligações semanticamente ricas.

0.2 Organização em capítulos

A presente dissertação está organizada em quatro capítulos. Ao longo destes capítulos são abordadas todas as questões conceptuais e tecnológicas, e apresenta-se um protótipo desenvolvido para testar o modelo proposto.

O primeiro deles aborda a forma como actualmente é efectuada a partilha e o acesso à informação, enumerando os problemas com que os investigadores se debatem diariamente para acompanhar a evolução do trabalho dos inúmeros colegas espalhados pelo planeta. Essa abordagem é dirigida sobre três vectores. O primeiro consiste nas soluções orientadas para a pesquisa nos próprios conteúdos e nos seus meta dados, os motores de pesquisa. O segundo, analisa as soluções orientadas para o autor. Usa para o efeito uma proposta e dois projectos, para o aproveitamento das citações colocadas pelos autores. Como forma de complementar a abordagem das citações é também abordada uma proposta de uma ontologia do conhecimento científico. O terceiro foca o espírito cooperativo entre utilizadores.

No segundo capítulo é proposto um serviço baseado num modelo de representação e partilha de conhecimento académico.

No terceiro capítulo são discutidas as tecnologias mais relevantes para a concepção de serviços baseados na *Web*. No centro desta abordagem está o XML, como um formato universal de representação e intercâmbio de informação, e o *Java*, com as inúmeras API's que já disponibiliza, assim como a sua extraordinária portabilidade.

O quarto capítulo descreve o protótipo desenvolvido para testar o modelo proposto no segundo capítulo. O seu desenvolvimento recorre às mesmas tecnologias abordadas no terceiro capítulo.

Finalmente, são apresentadas as conclusões gerais do estudo efectuado ao longo desta dissertação. Apresenta-se os resultados obtidos da aplicação do modelo no serviço proposto e conclui-se com algumas propostas de como poderão evoluir o modelo e serviço.

Capítulo 1 - Serviços de gestão e acesso à informação

O conhecimento é algo precioso, cuja disseminação deve ser feita de modo rápido, fácil e eficiente. As comunidades de investigação espalhadas pelo planeta, nas suas actividades, produzem quantidades consideráveis de conhecimento. A forma mais generalizada de o comunicar é através de publicação de artigos, livros e relatórios, etc.. Estes documentos encontram-se armazenados em bibliotecas, e, cada vez mais, na própria *Internet*, o que lhes aumenta a acessibilidade. No entanto, a grande quantidade de informação disponível pode constituir um obstáculo à sua transformação em conhecimento. É portanto relevante a implementação de mecanismos que facilitem essa transformação.

A *Internet* constitui uma plataforma de comunicações que, pelas suas potencialidades face aos custos de utilização, se tornou no meio de comunicação privilegiado para estabelecer um meio de acesso e partilha de informação entre as diversas comunidades que lhe estão conectadas. Num meio com tanta diversidade tecnológica e pessoas com saber e experiência nas respectivas áreas científicas, os sistemas colaborativos encontram o seu "habitat" natural.

Tecnologia digital, as oportunidades e os desafios

A *Internet* é o meio utilizado pelas diversas comunidades de investigação para comunicar e partilhar o seu conhecimento. Se por um lado a *Internet* facilita a partilha, também é um facto que esta coloca novos desafios. Dado este seu enorme raio de alcance e aceitação, que estabelece um contacto permanente entre praticamente todos os investigadores do mundo, leva a que hoje em dia seja muito difícil, ou até impossível, um investigador poder processar toda a informação científica a que diariamente tem acesso. Esta informação era disponibilizada através de revistas e relatórios científicos, isto é, no papel. Agora adicionalmente chega também pelas caixas de correio electrónico, listas de distribuição de mensagens, entre outros meios electrónicos, com referências a ainda mais informação. Todas estas fontes, em conjunto, fazem com que ocorra diariamente uma autêntica avalanche, que ninguém é capaz de absorver e aproveitar.

Não é correcto assumir [EF1998] que a natureza aberta e flexível do ambiente WWW, permitindo com extrema facilidade a ligação dos seus documentos, é o suficiente para facilitar a

aprendizagem ou melhorar as capacidades do utilizador. Por outro lado, a quantidade de informação disponível é enorme. Este facto constitui uma grande oportunidade [RHRS1993] para a aprendizagem e valorização pessoal mas também não é o suficiente para garantir a aprendizagem efectiva e proveitosa. Estas características sugerem que existem as condições essenciais para que sejam criados serviços que, aproveitando este vasto potencial, possam oferecer aos seus utilizadores aquilo que necessitam, para efectivamente melhorar a performance do trabalho e da aprendizagem.

Os serviços que actualmente os investigadores usam no seu trabalho diário são muito diversos. A seguir efectuar-se-á uma abordagem de alguns dos serviços que habitualmente são utilizados no trabalho diário daqueles que se dedicam à investigação.

1.1 Serviços de pesquisa WWW

O serviço *World Wide Web* constitui o modo mais simples de disseminar informação na comunidade global em que se vive hoje em dia. O facto conduz a que existam enormes quantidades de informação disponíveis na *Internet,* tendo o seu aumento uma tendência exponencial. Isto torna a gestão de toda a informação num processo bastante complexo. Actualmente os métodos mais comuns para encontrar a informação, na *Web*, consistem na combinação do recurso a motores de pesquisa com a navegação manual [KSC1998].

1.1.1 Motores de pesquisa

Um motor de pesquisa é um mecanismo de *software*, que funciona como um interface entre o utilizador e uma base de dados de índices. Estes são fruto de um processo de indexação, que pode ser automático ou manual.

Indexação automática

Os processos de indexação automáticos existentes num motor pesquisa, são designados por *spiders*, *crawlers* ou *robots*. Estes mecanismos, implementados em *software,* percorrem toda a malha de hiperligações entre documentos. Desta forma, percorrem documento a documento da vasta colecção de recursos espalhada pela *Internet*, indexando cada um deles. Alguns destes processos, são apenas capazes de actuar em sítios *Web*. Todavia, existem muitos motores de pesquisa que actuam sobre serviços como *Gopher*, FTP, grupos de notícias e listas de correio.

Os actuais volumes e dinamismo da informação disponíveis na *Internet*, em particular o serviço *Web*, colocam sérios problemas aos sistemas de indexação. Adicionalmente ao facto de em cada instante, algures, mais uma página ser publicada, os sistemas de indexação têm também de acompanhar as alterações efectuadas nos documentos já indexados. O que frequentemente acontece, é que não são capazes de eficazmente acompanhar essas alterações, nos conteúdos dos documentos, ou nas mudanças da sua localização.

Catalogação manual, serviços especializados e organização em directórios temáticos

Além dos processos automáticos de indexação, com a grande vantagem de permitir a indexação de grandes quantidades de documentos em curto prazo, existem serviços onde há intervenção humana. Nestes casos, é possível obter melhores resultados nas pesquisas, porque geralmente este tipo de indexação está organizado em directórios temáticos, com uma estrutura em árvore. Num cenário destes, a dimensão do problema do excesso de informação pode tornar-se mais reduzido, visto que o domínio dos documentos é mais pequeno. Naturalmente, nem sempre se pode esperar que o processo de catalogação esteja feito de forma que facilite encontrar aquilo que se pretende. Essa estruturação pode não ser a mais coerente, ficando o sucesso de uma pesquisa dependente do modo como a informação está organizada.

Além dos processos automáticos e manuais, atrás referidos, existe ainda a possibilidade do próprio autor do documento, ou administrador do sítio, efectuar voluntariamente o registo na base de dados do motor de pesquisa. Esta forma de catalogação, manual, permite que o responsável pelo conteúdo decida a classificação a dar ao registo. Este tipo de processo além de garantir que o registo na base de dados foi de facto efectuado, permite também que este seja feito segundo o ponto de vista do autor, que à partida deve ser o mais indicado. Assim, no caso de actualizações desses conteúdos, ou mudanças de localização não é necessário esperar que o automatismo volte a reindexar os documentos, o autor pode explicitamente fazê-lo.

Publicidade

Estes serviços têm geralmente como meio de subsistência a publicidade, dado que são de acesso livre, sem qualquer custo para o utilizador. O preço da publicidade nestes serviços cresce com o número de utilizadores. Por isso, os responsáveis de um serviço de motor de pesquisa têm de estar bastante atentos aos aspectos que mais atraem os utilizadores. Alguns desses aspectos, que requerem uma constante vigilância e investimento, são: o tamanho das bases de dados de índices; o seu conteúdo e a frequência da sua actualização; tempo de resposta a uma interrogação; conjunto de facilidades de pesquisa; o *design* gráfico do interface com utilizador; e naturalmente a sua facilidade de uso. Outras facilidades adicionais como a tradução automática dos conteúdos também são muito frequentes.

Serviço *Altavista*

Um exemplo de um destes serviços é o famoso *Altavista*, disponível através do endereço http://www.altavista.com. Este serviço, de que se mostra na Figura 1 a página inicial, permite a qualquer utilizador, sem qualquer custo, fazer todas as pesquisas que necessitar. É um serviço que, como muitos outros, apenas com uma palavra chave fornece uma imensidão de resultados.

Figura 1 - Página inicial do serviço *Altavista*

Permite vários esquemas de pesquisa, sobre vários tipos de conteúdo. Desses vários esquemas de pesquisa talvez aquele que mais frequentemente é utilizado é o que está disponível logo na página inicial. Aceita como parâmetros de pesquisa palavras chave e a língua em que os documentos estão arquivados. Assim, só referências a recursos nessa língua é que surgem como resultados. Note-se que este parâmetro não faz qualquer tradução, mas sim uma filtragem. Outro esquema de pesquisa, designado por *Power Search*, permite outro tipo de parâmetros, como:

- Qual o tipo da filtragem mediante a expressão de pesquisa que deve ser efectuada, em que as hipóteses são: qualquer uma das palavras, todas as palavras, exactamente essa expressão (como uma frase), ou ainda como uma expressão booleana.
- Se a pesquisa deve restringir-se a alguma parte do documento, ou a todo o documento.
- Quais as datas de publicação
- Língua.
- Localização geográfica.

De facto trata-se de uma pesquisa poderosa, como o próprio nome indica. Outra forma de especificar os parâmetros de pesquisa consiste no *Advanced Search*. Este método, para além de permitir a pesquisa mediante um dado conjunto de opções, destaca-se pela possibilidade de especificar o género de conteúdo a encontrar, isto é, imagens, vídeo, áudio em formato

MP3. Contudo, não é a única forma em que este serviço permite a pesquisa de outros tipos de conteúdos para além do texto.

Depois de submetida uma pesquisa de qualquer um dos tipos atrás descrito, o serviço fornece um conjunto de resultados, sob a forma de hiperligações aos documentos que se enquadram nos requisitos dos parâmetros fornecidos.

Para além destes resultados, o serviço fornece também um conjunto de sugestões alternativas para a pesquisa que foi efectuada. Por exemplo, quando é fornecida a palavra chave "*sendmail*"[1] o serviço *Altavista* sugere outras pesquisas relacionadas com este assunto, tais como: "*sendmail configuration*", "*sendmail tutorial*", "*sendmail linux*", "*sendmail NT*", etc..

Outra forma de facilitar a pesquisa consiste numa organização temática que permite ao utilizador efectuar pesquisas dentro de um dado domínio de assuntos. Um exemplo dessa organização é o seguinte: Artes/Música/Educação. O resultado de tal pesquisa consiste num conjunto de hiperligações a escolas de música, professores de música, aulas de música, etc. Mas quando se entra no directório "professores", os resultados alteram-se, restringem-se apenas a documentos relacionados com professores de música. Em qualquer nível desta estrutura arborescente, é possível efectuar uma pesquisa por palavra chave. Porém, essa pesquisa não se restringe apenas a esse nível, mas sim a toda a árvore, o que naturalmente constitui algo de limitador.

Adicionalmente a tudo isto, o serviço *Altavista*, permite também a tradução automática dos documentos encontrados, entre muitas outras facilidades, cuja descrição exaustiva seria demasiado longa. Com esta pequena descrição, pretendeu-se fazer uma abordagem das características regra geral encontradas nos motores de pesquisa, usando para essa análise um caso particular.

1.1.2 Meta motores de pesquisa

Muitos dos serviços de motores de pesquisa têm bases de dados pequenas face à quantidade de informação existente. Este facto obriga os utilizadores a recorrer a mais de um deles sempre que não há, ou são insuficientes, as referências sobre o que pretendem. Os meta motores de pesquisa [CTRCORP1997] conseguem automatizar o processo de pesquisa em vários motores. Serviços como estes permitem que o utilizador, através de uma única página *Web* central, possa efectuar várias pesquisas em simultâneo. Obviamente soluções deste género conduzem a uma substancial economia de tempo, quando é necessário recorrer a mais do que um destes serviços, para encontrar o que se pretende.

Existem vários serviços deste género, com mais ou menos sofisticação. Os mais sofisticados medem a performance relativa entre os diversos motores a que recorrem. São ainda capazes de combinar resultados, eliminando duplicados, agrupá-los por rubricas temáticas e de os ordenar, segundo uma taxa de sucesso, relativamente à interrogação submetida no formulário

[1] *Sendmail* é o nome de uma aplicação servidora de correio electrónico.

de pesquisa. Alguns renovam a pesquisa alertando o utilizador automaticamente por correio electrónico sempre que surja uma novidade [CR1998].

Os vários serviços deste tipo podem ser classificados em três categorias [CTRCORP1997]. As características e alguns exemplos desses serviços são os seguintes :

- Os mais básicos têm apenas a vantagem de permitir a reutilização da expressão de texto para a interrogação. Fornecem o serviço mínimo, o de permitir a pesquisa em vários motores. Mas o maior benefício, o da economia de tempo, em virtude da pesquisa ser feita em simultâneo neste caso não acontece. O utilizador tem de, um a um, submeter nova pesquisa e guardar os resultados de cada uma.

  Um exemplo deste tipo de serviço é o *StartingPoint*, que está disponível através do endereço http://www.stpt.com.

  A `Figura 2` ilustra o interface com o utilizador. Nesta página inicial submeteu-se uma pesquisa na caixa de texto "*Search Other Resources*", com a expressão pesquisa de apenas uma palavra chave, "*sendmail*". Como resultado, surge uma nova página, ilustrada na `Figura 3`, com hiperligações para outros motores, ao contrário dos convencionais motores de pesquisa, que fornecem de imediato as hiperligações directas. Estas hiperligações estão devidamente parametrizadas para efectuarem uma nova pesquisa directamente num dos motores que fornecem os meta dados de indexação. Desta vez, o resultado já contém as presumíveis hiperligações, pretendidas pelo utilizador. Na prática, este serviço limita-se a estruturar as diversas hiperligações para uma interrogação nos vários motores. O valor acrescentado por este serviço aos diversos serviços que usa é o de poupar ao utilizador a abertura das diversas páginas de introdução de parâmetros de pesquisa, e naturalmente o de evitar a que tenha de repetidamente introduzir a mesma expressão de pesquisa, com as palavras chave. No caso do utilizador usar expressões de pesquisa com algum nível de complexidade, este serviço faz a tradução para as diversas sintaxes de pesquisa, o que já constitui uma vantagem. Será de salientar que este serviço, na sua fase final, passa todo o controlo para o motor que fornece as hiperligações ao utilizador, isto é, o navegador *Web* perde a conexão com o servidor do serviço *StartingPoint*.

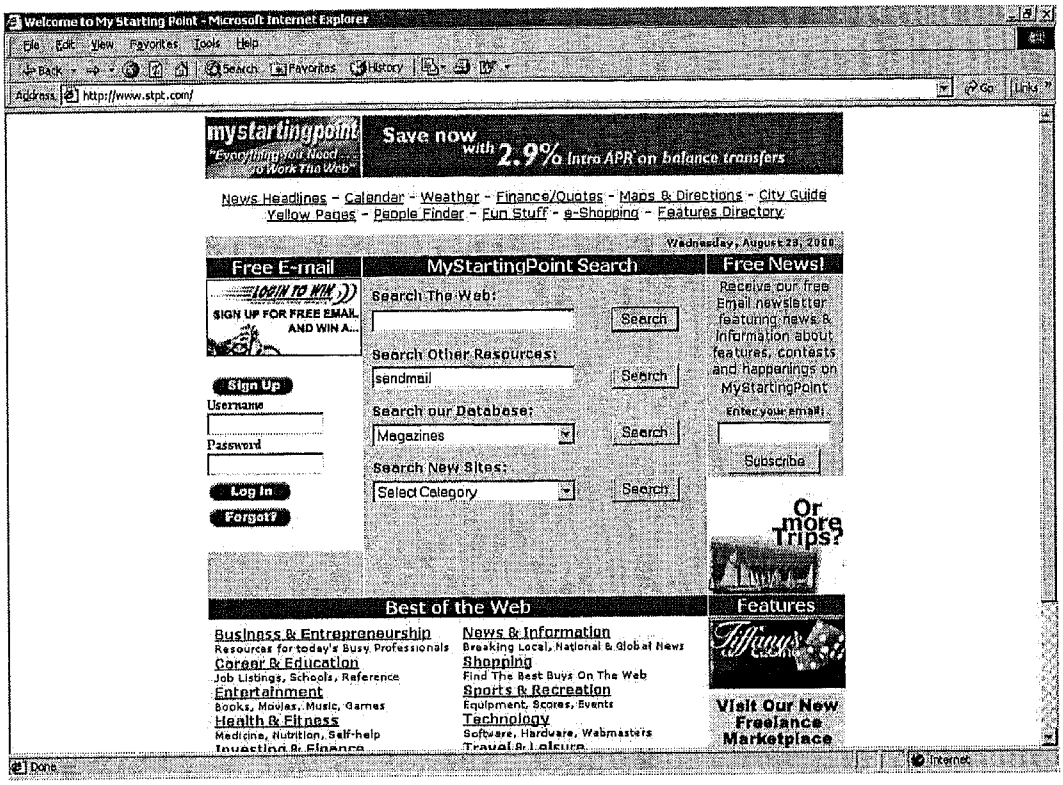


Figura 2 - Página inicial do *StartingPoint*

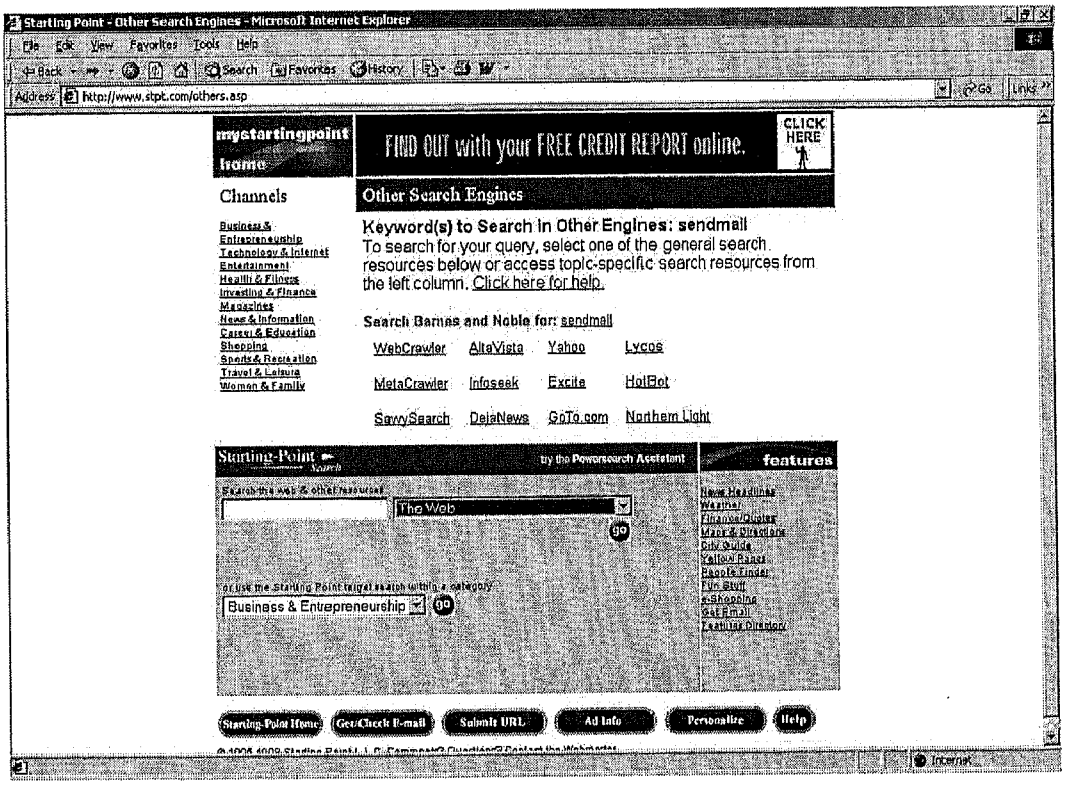


Figura 3 – Página de hiperligações a outros motores de pesquisa

- Outro tipo de serviço do género, mais sofisticado, permite que de facto a pesquisa seja feita em simultâneo. Estes serviços, já permitem ao utilizador uma substancial economia de tempo, visto que não tem um tempo de pesquisa correspondente ao somatório dos tempos de resposta dos vários serviços consultados, se o utilizador pretender todos os resultados. Um exemplo deste género de serviços é o *All 4 One*, disponível através do endereço http://www.all4one.com. Porém, estes serviços ainda têm a limitação de obrigar o utilizador a tratar individualmente os resultados das diversas pesquisas, agora já efectuadas em simultâneo. As duas figuras a seguir apresentadas ilustram o resultado dado por este serviço quando lhe é submetida a mesma pesquisa que foi efectuada no *StartingPoint*.

  Este motor submete a pesquisa efectuada pelo utilizador aos motores *Altavista*, *Lycos*[1], *HotBot*[2] e *Excite*[3].

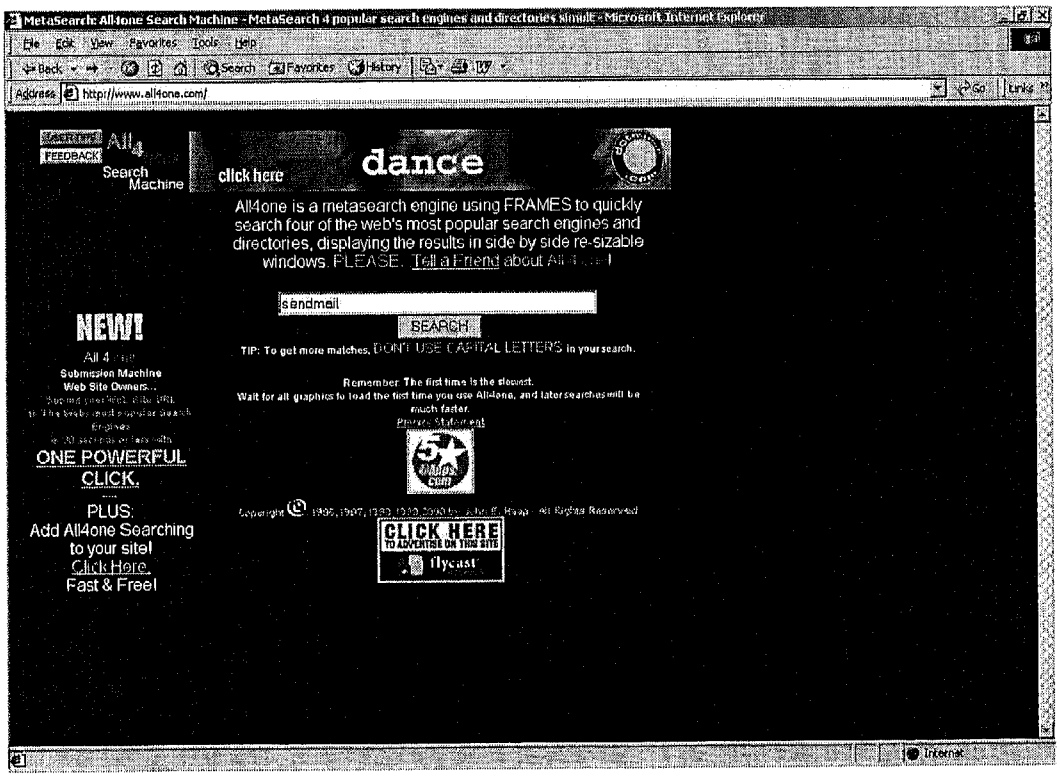


Figura 4 – Página inicial do *All4One*

[1] http://www.lycos.com
[2] http://www.hotbot.com
[3] http://www.excite.com

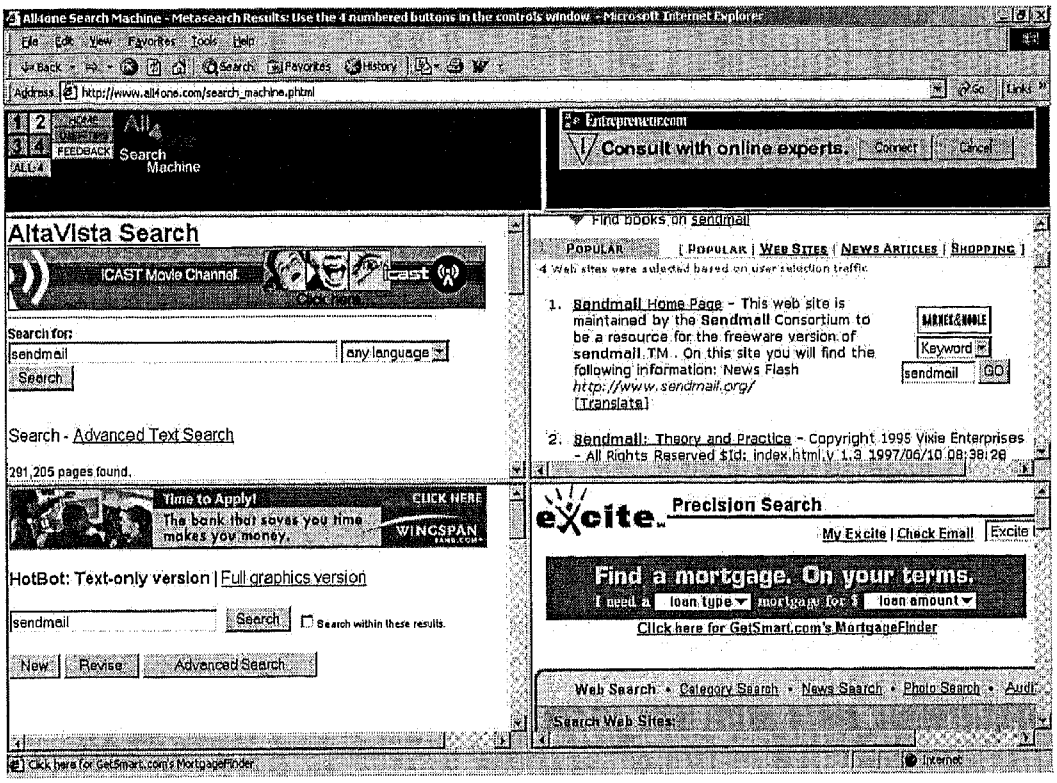


Figura 5 – Página de resultados da pesquisa nos 4 motores

- Nesta categoria, os meta motores de pesquisa já permitem a combinação dos diversos resultados numa única página. Um serviço a título de exemplo para esta categoria, é o *Highway61*, disponível através do endereço http://www.highway61.com. Este serviço, adicionalmente às categorias e exemplos atrás referidos, é capaz de combinar os diversos resultados numa única página. É também capaz de os ordenar e estabelecer uma pontuação para cada uma das hiperligações. Juntamente com a expressão de texto da pesquisa, as palavras chave, o utilizador pode definir alguns parâmetros adicionais. O tempo que o está disposto a esperar, a quantidade de resultados, operadores booleanos, etc.. As figuras que se seguem ilustram uma pesquisa, utilizando a mesma palavra chave nos dois exemplos anteriores.

  Este motor submete a pesquisa efectuada pelo utilizador aos motores *Yahoo*[1], *Lycos Webcrawler*[2], *Infoseek*[3], e *Excite*. Dado que são vários os fornecedores de meta dados, o mecanismo utilizado por este serviço, o *DigiWeb*[4], tem de tratar os diversos resultados parciais com formatos diferentes.

[1] http://www.yhaoo.com

[2] http://www.webcrawler.com

[3] http://www.infoseek.com

[4] Mais informações sobre este mecanismo em http://www.digiweb.com

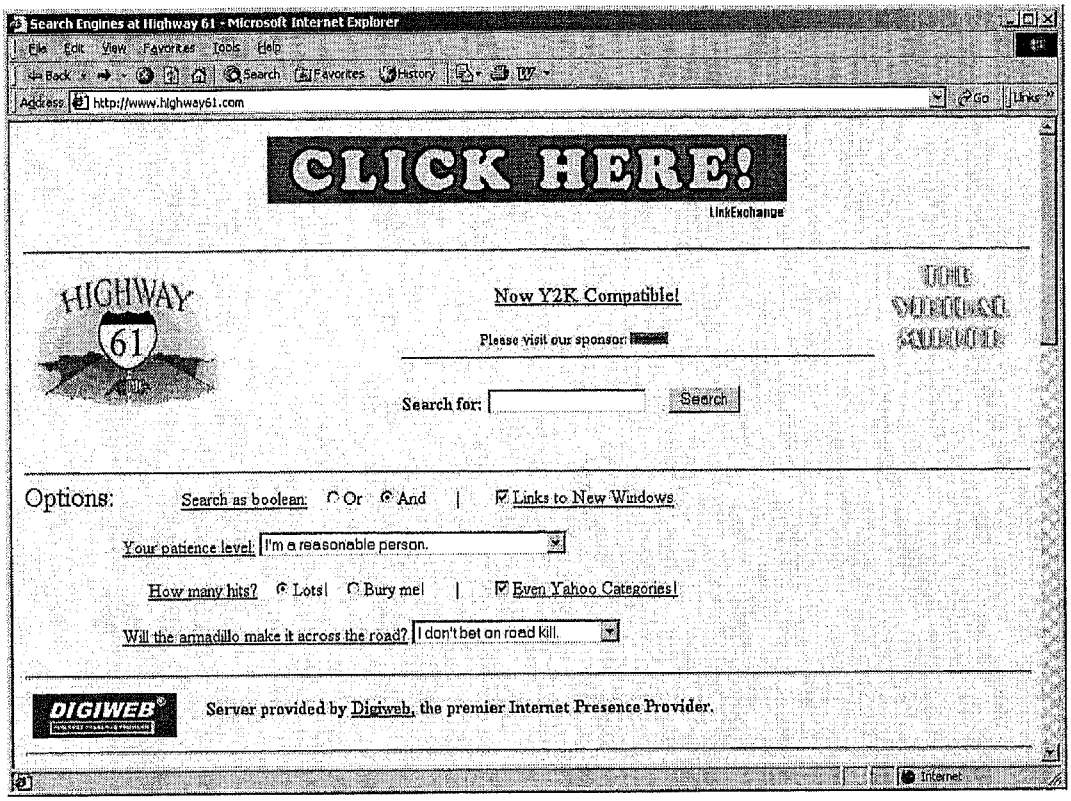


Figura 6 - Página inicial do *Highway61*

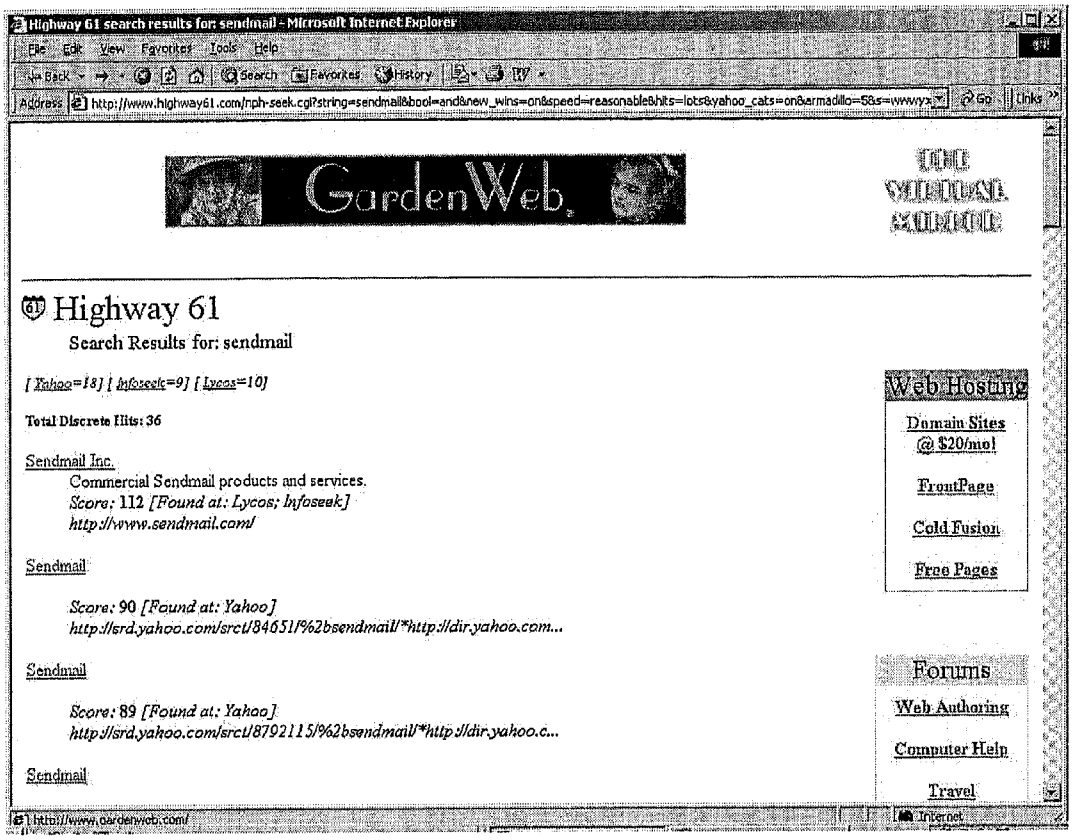


Figura 7 - Página de resultados combinados

Adicionalmente aos meta motores das três categorias anteriormente descritas, de primeira geração, os de segunda geração também designados por *off-line* são um prolongamento lógico instalados no computador do utilizador [CR1998]. O princípio de funcionamento é exactamente o mesmo. Para cada pedido, interrogam vários motores de pesquisa e eliminam as repetições. Mas a sua originalidade é o facto de não haver necessidade de uma ligação directa à *Internet* para os fazer funcionar. De facto, pode definir no estado *off-line* a sua estratégia de pesquisa, escolhendo as palavras-chave apropriadas e os motores que deseja utilizar para cada missão. Pode também definir a hora e a periodicidade da sua pesquisa tal como o local onde deseja que o agente guarde os resultados da pesquisa. Deste modo, no momento escolhido, o agente activa a ligação à *Internet*, interroga os motores com as palavras-chave que lhe foram fornecidas, funde os resultados e elimina as repetições, guarda-os no disco local, e por fim, termina a ligação à *Internet*. Depois de concluído o processo de pesquisa e desligado de *Internet* os resultados podem ser consultados a partir do disco local.

Actualmente, podem ser encontrados vários meta motores *off-line* como: *Copernic*[1], *NetAttaché Light*[2], *WebFerret*[3], etc..

[1] Mais informações disponíveis em http://www.copernic.com

[2] Mais informações disponíveis em http://www.tympani.com

[3] Mais informações disponíveis em http://www.ferretsoft.com

1.1.3 A gestão da informação baseada em motores de pesquisa

O ruído e o silêncio

Qualquer utilizador que pretenda usufruir da informação que existe na *Internet*, recorrendo aos motores de pesquisa não tem grande dificuldade em encontrar referências para a informação pretendida. Embora o número de referências possa ser tão elevado que obrigue o utilizador a efectuar uma nova pesquisa na tentativa de encontrar, de entre a imensidão de resultados, aquilo que pretende. O conjunto de resultados que surgem juntamente com aqueles que de facto são relevantes para o utilizador, são "ruído" que escondem a informação que tem interesse. A tarefa de refinamento não é fácil, porque os filtros de refinamento das pesquisas não eliminam esse ruído sem correr o risco de eliminar também referências com interesse para o utilizador, aquelas que correspondem à necessidade do utilizador: um artigo, um programa, um relato de uma experiência de alguém que já passou por um problema igual àquele que tem para resolver, etc.. A precisão de uma pesquisa é tão mais elevada quanto menor for o ruído do seu resultado, por outras palavras, quanto mais precisa for uma pesquisa mais relevantes são os resultados. Assim sendo, a precisão com que uma pesquisa é efectuada é bastante importante e pode auxiliar na redução do ruído. Este conceito de precisão de uma pesquisa é frequentemente utilizado em motores de pesquisa. Quando a expressão de pesquisa contém várias palavras, o número destas contido num documento pode ser utilizado como critério da sua relevância. O mesmo se diga da sua ordem pela qual aparecem as palavras na expressão e no documento.

Um exemplo de um motor que usa uma estratégia deste género está disponível no sítio *Web* da *Sun* dedicado à linguagem de programação *Java*, através do endereço http://www.javasoft.com. A figura a seguir apresentada ilustra como os resultados são classificados para duas palavras chave cuja ordem e adjacência é relevante, e permite obter uma maior precisão da pesquisa recorrendo à relevância de cada resultado.

Como é possível constatar, na figura, no topo do resultados estão os dois mais relevantes, um com 57% e outro com 55%, referem documentos onde as duas palavras chave foram encontradas. Os dois porcento de diferença que separam o primeiro do segundo, devem-se ao facto do primeiro conter as duas palavras com a mesma disposição (tal como numa frase) com que foram colocadas na caixa de texto do formulário de pesquisa.

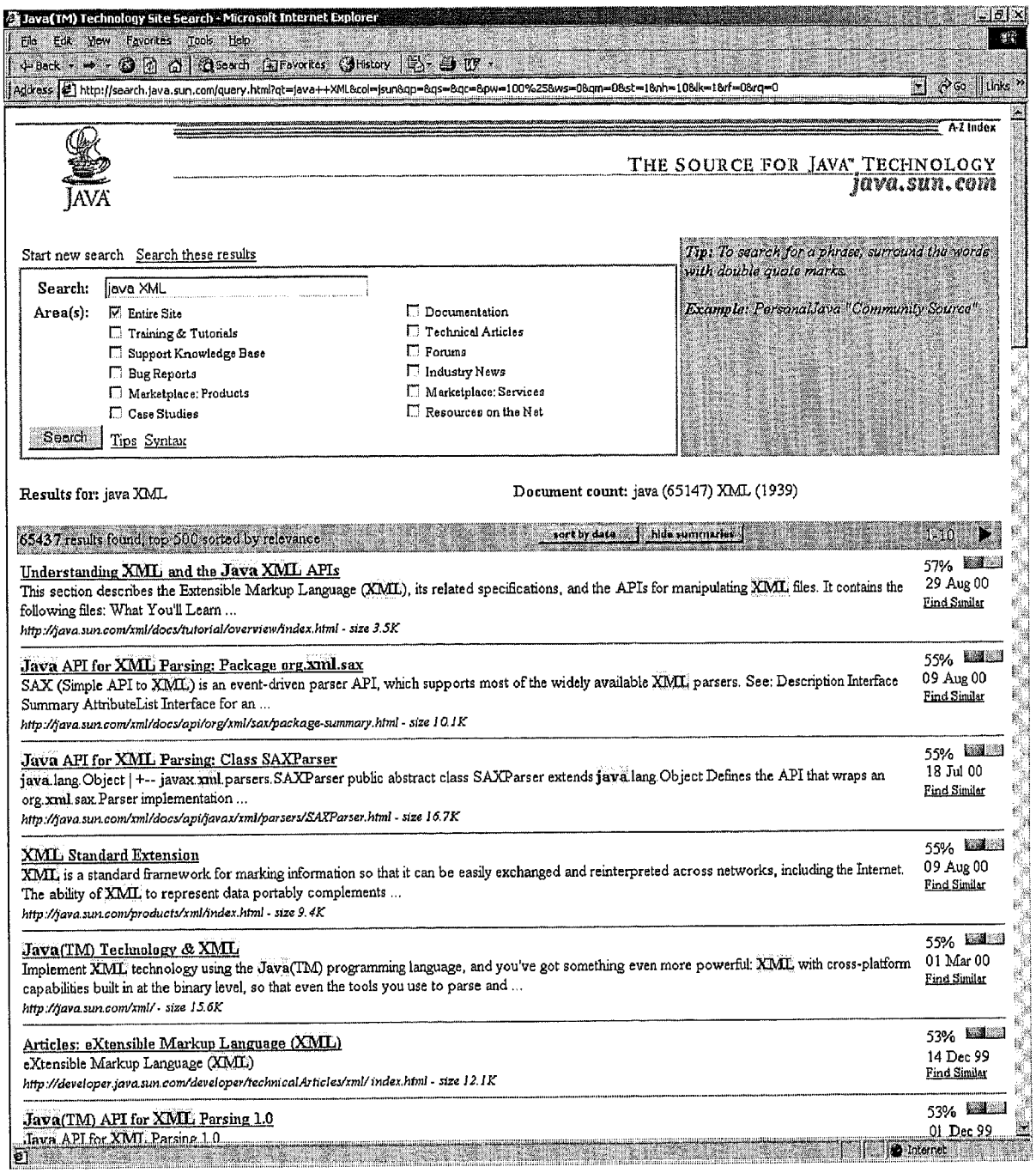


Figura 8 - Página do motor de pesquisa da Sun

Esta forma como a gestão da informação disponível na *Internet* é efectuada, regra geral baseada em motores de pesquisa em texto integral e outros com informação de catalogação, não é suficiente para dar respostas eficazes. Os resultados obtidos numa pesquisa, muitas vezes nada têm a ver com o que se pretendia. Este ruído em torno dos resultados de pesquisas, pode ser devido a vários factores: porque uma das palavras chave utilizadas na pesquisa tem um sinónimo algures no texto, ou então porque a palavra surge no texto, mas está totalmente desenquadrada do resto do contexto. Os sistemas de pesquisa em texto integral limitam-se a

apurar quais os documentos, onde constam as palavras chave da pesquisa, recorrendo a meta dados, resultado de uma pré indexação automática. E se surgem respostas totalmente fora do âmbito pretendido, o "silêncio" também acontece se: as palavras utilizadas na pesquisa não constam no documento, mas o documento tem total interesse; o tempo verbal não é o mesmo; a palavra chave está no singular mas no documento está no plural. Nestes casos, não surge como resposta da pesquisa o que obviamente constitui um sério problema.

Depois de se chegar a um conjunto mais reduzido de referências, há ainda um outro nível de pesquisa, onde não há nenhum processo automatizado que faça essa tarefa. O utilizador tem de, manualmente, carregar documento a documento, analisá-lo, e determinar se o documento em causa se enquadra no que pretendia. É claro que é aqui que o consumo de tempo é maior.

Genericamente uma solução baseada em motores de busca tem uma forte dependência relativamente ao tamanho da sua base de dados. Estes sistemas existem em bastante quantidade, espalhados pela *Internet*, e muitos deles têm bases de dados pequenas, comparativamente com a informação disponível. Portanto nem sempre o utilizador encontra o que pretende logo nos primeiros. Desta forma, os utilizadores acabam por recorrer a meta motores de pesquisa que, naturalmente, simplificam essa tarefa. Mas a consequência é obvia: o ruído. Como usam motores de pesquisa bastante diferentes, não permitem efectuar pesquisas avançadas muito fiáveis. Com efeito, geralmente, cada ferramenta de pesquisa possui opções avançadas que lhe são características. É, por isso, quase impossível os mesmos operadores de pesquisa para ferramentas diferentes. E mesmo quando possível os resultados não são tão precisos como numa interrogação individual a cada motor. Concluindo, a sua principal vantagem consiste em "desbravar rapidamente o terreno" [CR1998], fornecendo um panorama geral sobre os documentos disponíveis. Permitindo um bom compromisso entre o número de motores interrogados e o tempo necessário.

Uma forma possível de o atenuar o ruído será a de recorrer a serviços especializados no respectivo assunto. Além dos meta motores, motores generalistas (*Altavista*, *HotBot*, *Lycos*, *Excite*, etc.), dos catálogos temáticos (*Yhaoo*, Sapo[1]) um categoria de motores de pesquisa, em plena expansão, são aqueles especializados por sector de actividade específico. São motores, ou meta motores especializados unicamente num único domínio preciso (finanças, educação, literatura, etc.). Frequentemente são fundos documentais fora do alcance dos robôs dos motores generalistas, disponíveis apenas através de serviços pagos.

Depois desta pequena abordagem sobre os motores de pesquisa, pode-se constatar que estes apresentam grandes limitações, conduzindo a um consumo elevado de tempo. A pesquisa baseada em palavras chave não é suficiente pois estas são apenas uma aproximação de conceitos e do significado das palavras. Por outro lado, com um universo de informação tão vasto como o da *Internet* e a necessidade de haver sempre uma intervenção manual, a pesquisa torna-se bastante complexa e demorada. Neste contexto, o papel de outros formatos alternativos ao HTML [DRAHIJ1999] (ver em 3.1.1.2) e a aplicação de técnicas de inteligência artificial podem certamente resolver muitos dos actuais problemas.

[1] http://www.sapo.pt

Vantagem da representação da informação em XML

A forma de representação da informação, baseada em HTML, também não contribui de forma favorável para a resolução dos problemas até aqui referidos. O HTML, sendo actualmente o suporte da maioria da informação na *Internet*, é uma linguagem baseada em marcadores, cujas origens estão no SGML [SGML1986] (ver em 3.1.1.1). Dado o facto de esta linguagem não ter sido estruturada para permitir separar as marcas ligadas à apresentação, da informação propriamente dita, os resultados de pesquisas, em texto integral, sejam imprecisos. Por exemplo: Uma página HTML poderá comportar a indicação: "`<B> Livro de matemática = 100 páginas A4 </B>`", isto significa apenas que o texto entre as marcas, <B> e </B>, deverá ser apresentado em negrito. Em contrapartida, a indicação não revela nada sobre a natureza dessa informação (número e formato das páginas do documento). Quem consulta a página HTML lerá a expressão: "**Livro de matemática = 100 páginas A4**" apresentada num tipo de fonte carregada, e, provavelmente deduzirá que o documento em causa tem 100 páginas e que o formato dessas páginas é o formato A4 (210 x 297 mm). Mas, apesar desta dedução ser possível para um ser humano, um computador não tem a mesma capacidade. Este tipo de limitação do HTML, o não possuir a indicação do significado da informação, leva a que, por exemplo, pesquisar na *Web* quais as livrarias que têm livros de matemática com mais de 100 páginas, no formato A4, seja difícil ou até mesmo impossível de encontrar.

A linguagem XML [TBJPCS1998] (ver em 3.1.1.3) dá resposta a necessidades deste género, dado que é uma linguagem que marca o tipo de dados, e lhes dá significado, por forma a torná-los compreensíveis, mesmo por computadores diferentes, em qualquer plataforma. A forma como o XML organiza e estrutura a informação, vai permitir aos sistemas de pesquisa, a procura não por palavras mas por conceitos, o que naturalmente é muito mais poderoso.

Para além do HTML, outros formatos também largamente usados, como o PDF, PS, *LaTeX*, entre outros, são bastante utilizados, especialmente em artigos científicos. O facto agrava a situação, porque a generalidade dos motores de pesquisa não suporta estes formatos de uma forma directa ou mesmo os ignora no processo de indexação, o que naturalmente impossibilita a posterior pesquisa. Porém, existem inúmeros sítios onde grandes repositórios de documentos nestes formatos são pesquisáveis, mas não de uma forma directa, isto é, torna-se necessário recorrer a esses sítios e utilizar motores por eles fornecidos.

Aplicação de técnicas de inteligência artificial

O passo que actualmente já está a ser dado é o de introduzir técnicas de inteligência artificial no tratamento dos resultados.

Este género de sistemas começam a surgir no mercado[1], tipicamente desenvolvidos por departamentos de investigação em universidades.

O objectivo destes sistemas é o de exaustivamente procurar pelos conceitos, utilizando nessas pesquisas frases em linguagem natural, em vez de palavras chave, como actualmente é feito.

O uso de serviços de pesquisa inteligentes, constitui uma forma de pesquisar bastante promissora, onde a maior parte dos aspectos negativos que os actuais sistemas de pesquisa

[1] Um conjunto de referências sobre projectos nesta área estão disponíveis a partir do endereço http://www.cybion.fr/intelligence ou http://www.veille.com

têm poderão ser colmatados. A inteligência artificial constitui uma ferramenta bastante poderosa para lidar com os actuais, e futuros, volumes de informação existentes na *Internet* e *Intranets*. Para as pesquisas em conteúdos multimédia estas técnicas também terão bastante importância.

1.2 Serviços baseados em informação colocada pelo autor

Reconhecendo o facto de que não bastam bons mecanismos de indexação, desenvolveram-se serviços complementares que constituem um poderoso leque de possibilidades de acesso à melhor informação.

Actualmente o recurso a citações a outros trabalhos, regra geral é a forma utilizada para sustentar afirmações ou pressupostos efectuados. Se as referências e citações a outros documentos no domínio do papel já são bastante utilizadas, com muitas vantagens, esse interesse pelas citações no domínio dos documentos electrónicos não diminuiu. Pelo contrário, fez com que todo esse potencial das citações se multiplicasse, multiplicando também o interesse.

Ao utilizar as citações os investigadores interligam os seus trabalhos. Essas interligações para os investigadores têm bastante interesse, pois é graças a elas que podem seguir todo o trabalho desenvolvido numa dada área, isto é, encontrar nova informação e fazer pesquisa bibliográfica

Na *Internet*, grande parte do sucesso do serviço WWW deve-se às hiperligações entre documentos. No mundo académico, os investigadores usam as citações formais como método de ligar os seus documentos. Se actualmente ao nível do papel já existem grandes vantagens na navegação rudimentar que é feita ao seguir as citações de artigo em artigo, então, com esses documentos em formatos electrónicos, essas citações facilmente se transformam em hiperligações. A rapidez com que a navegação poderia ser efectuada seria extremamente mais elevada. Esta é uma das várias vantagens no uso das citações como meio de ligar a informação académica. Essa e outras vantagens serão abordadas no decorrer do presente capítulo.

Vários estudos, projectos e até aplicações comerciais[1] para a utilização das citações nos documentos, têm sido feitos nesta área alguns dos quais serão aqui abordados.

1.2.1 Uma base de dados universal de citações

Objectivos da proposta

Robert D. Cameron no seu trabalho "*A Universal Citation Database as a Catalyst for Reform in Scholarly Communication*" [RDC1997] propôs a constituição de uma gigantesca e universal base de dados de informação bibliográfica e de citações, baseada num serviço

[1] *Keycite*, 1997 e *Institute for Scientific Information*, 1997

Internet. Esta base de dados de citações que interligaria todo o tipo de trabalhos: artigos, teses, relatórios técnicos, etc. alguma vez escritos, que fosse diariamente actualizada e de livre acesso, sem qualquer custo para o utilizador, via *Internet*, certamente revolucionaria a forma como as instituições, particularmente as de ensino, manteriam a sua informação. A forma de localizar e acompanhar a evolução dos trabalhos levados a cabo por cada uma das instituições alterava-se completamente [RDC1997]. Isto teria resultados bastante benéficos [RDC1997], tendo em conta que muitos jornais e artigos científicos estão disponíveis electronicamente em formatos como o PDF.

Actualmente a *Web* é o serviço *Internet* que faz chegar essa informação aos interessados. Mas, apesar do formato PDF possibilitar a inserção de hiperligações e até meta dados, essa informação não é eficientemente utilizada pelos motores de pesquisa. Como foi referido em [1.1.3], só motores de pesquisa específicos são capazes de efectuar tais pesquisas. Esta situação leva a que o acesso a muita informação com valor seja difícil.

Descrição da proposta - Modelos

O autor propõe também, três modelos para diferentes aspectos da concepção de bases de dados universais de citações.

O primeiro, é um modelo operacional relativo à forma como a informação bibliográfica é introduzida na base de dados. Seriam as próprias instituições que produzem a literatura que manteriam a sua informação bibliográfica num formato normalizado e disponível para as outras instituições via *Internet,* num cenário totalmente distribuído.

É claro que tal modelo só teria sucesso com o empenho dos autores, pelo facto de terem o trabalho adicional de formatar devidamente as referências. Mas se, quando o artigo é aceite para publicação, existe também o trabalho adicional de formatar essas referências segundo as convenções da organização publicadora, e se actualmente, os autores têm de seleccionar e formatar as referências para todos os artigos que escrevem, com a facilidade que a base de dados poderá oferecer de permitir, com o recurso a *software* específico, a geração automática das referências bibliográficas, então, haverá certamente aqui uma economia de tempo que facilmente os convence a adoptar o serviço prestado pela base de dados, para o registo das novas referências bibliográficas.

É ainda proposta, a criação de um identificador canónico, único da referência bibliográfica. Assim, através deste identificador, todas as referências da base de dados estariam facilmente acessíveis e facilmente seriam convertíveis para qualquer outro formato de referência bibliográfica, com o auxilio de *software* como o *EndNote*.

Um segundo modelo descreve aspectos técnicos, ligados com a minimização dos acessos a base remotas (e naturalmente o tráfego na rede), propondo a replicação da base de dados, e, assim, apenas trabalhos mais recentes seriam acessíveis remotamente.

O terceiro modelo proposto, prende-se com questões ligadas aos objectivos iniciais e realistas, para um projecto de uma base de dados universal de citações. O autor propõe o conceito de *semi universalidade*, isto é, que inicialmente não estejam todas, mas apenas as ligações a partir de uma determinada data. Trabalhos mais antigos estariam disponíveis na base de dados apenas através da sua informação bibliográfica. Com esta proposta realista [RDC1997], apenas duas tarefas têm de ser garantidas: a retro expansão da colecção da base de dados

referente a todo o trabalho anterior; e o estabelecimento de um conjunto de procedimentos que garantam a catalogação apropriada. Logo após a inicialização da base de dados semi universal já teria muitos dos benefícios de uma totalmente universal [RDC1997].

Resultados deste trabalho

Uma tal base de dados teria um valor considerável na pesquisa bibliográfica e na avaliação do trabalho académico, contribuindo como um forte catalisador para a reforma no modo da comunicação académica.

Os modelos sugerem soluções operacionais e técnicas para um possível projecto.

Outra possibilidade é que alguma forma de base de dados universal de citações naturalmente surja, e cresça, a partir dos desenvolvimentos da *Web*. Baseada em robôs, que sistematicamente explorem o espaço da *Web*, o protótipo já está criado, apenas existem várias limitações [RDC1997]. Uma delas é a forma como na *Web* são acedidos os objectos, via *Universal Resource Locators* - URL[1], a actual implementação das citações da *Web*. Esta forma é demasiado técnica, sendo preferível um identificador mais abstracto, como o *Universal Resource Name* – URN [KSLM1994], que poderá vir a servir como fundamento de uma base de dados de citações, no universo da *Web* [RDC1997].

1.2.2 Projecto *Open Journal*

Muitos aspectos da publicação de jornais científicos foram completamente alterados com o aparecimento da *Web* e a facilidade com que se processa a distribuição da informação electrónica. O formato PDF e as hiperligações são dois marcos importantes no processo de transformação. Porém, têm ainda de se adaptar [SH1998b].

O uso da *Internet* para fazer chegar jornais e artigos científicos a todos que deles necessitam, introduz grandes vantagens. Desse conjunto de vantagens, as hiperligações multimédia são o mais importante. O projecto *Open Journal*[2] [SH1998b], utilizou ferramentas que permitiram a hiperligação dos textos, acessíveis de uma forma totalmente aberta. Este projecto, teve início em Maio de 1995 e conclusão em Maio de 1998, e foi criado através do programa *Electronic Libraries*[3] - *eLib*. Durante o período de três anos, o número de jornais electrónicos disponíveis passou de umas poucas dezenas para uns poucos milhares. O mesmo se passou com as funcionalidades do serviço, que também foram evoluindo durante o mesmo período.

Objectivos do Projecto

O objectivo do projecto foi o de proporcionar a estrutura necessária à publicação de jornais científicos, num ambiente de rede, em particular a *Web*. O serviço a criar teria de utilizar a

[1] O protocolo utilizado, o endereço do computador e porto, e a localização no sistema de ficheiros.

[2] http://journals.ecs.sotan.ac.uk

[3] http://www.ukoln.ac.uk/services/elib/

informação de ligação contida nos tradicionais artigos, de modo a aumentar a capacidade dos leitores em seguir essas ligações, pesquisar e ter acesso a literatura numa dada área, usando para isso o máximo de recursos disponíveis. A esses recursos, documentos nos formatos HTML e PDF, foram adicionadas ligações de hipertexto utilizando um serviço designado por *Distributed Link Service* – DLS [SH1998b].

Descrição do projecto

O serviço DLS, tem como intuito estabelecer uma relação entre os diversos documentos, de modo a tornar os recursos disponíveis não como elementos isolados, mas em vez disso, cooperantes. O serviço teve uma evolução durante os três anos de modo a que no fim desse período estava implementado um conjunto de módulos [SH1998b], os quais serão a seguir descritos.

- *Citation Link*

O serviço DLS não faz qualquer alteração dos documentos originais, em vez disso, é capaz de, em tempo real, isto é, no momento da visualização do documento, estabelecer a hiperligação. Para o utilizador as ligações surgem como numa normal página *Web*. Durante o processo de visualização, um mecanismo automático retira de uma base de dados toda a informação referente às ligações do documento em causa, e adiciona-as ao documento em visualização. Sendo todo este processo realizado em tempo real, o factor velocidade de processamento é crítico. Então, foram tomadas medidas de modo a garantir a eficiência do processo: o código do mecanismo foi escrito em linguagem C; procedeu-se à migração das bases de dados em texto, para sistemas de gestão de bases de dados; e ainda o pré processamento, de modo a obter-se versões pré processadas de páginas conhecidas. Assim, os utilizadores, no caso de requisitarem uma dessas páginas não teriam de esperar, bastaria encontrá-la numa *cache* de páginas já pré processadas.

Outra faceta importante do serviço é a sua capacidade de automaticamente extrair todas as citações a outros documentos, jornais ou não, e assim gerar a informação necessária ao estabelecimento das respectivas ligações.

O reconhecimento dos diferentes estilos de citação, foi uma das questões a levar em consideração. Existem poucos tipos de referências largamente aceites em jornais científicos, mas nesses estilos, existe uma grande quantidade de variantes específicas de cada jornal. Enquanto o agente automático usado para reconhecer as citações depender da forma correcta e consistente das mesmas, é possível definir regras e técnicas baseadas em métodos heurísticos, para identificar estilos específicos.

- *Keyword Link*

Outro tipo de hiperligação, consiste no relacionamento entre documentos a partir de palavras chave. O processo consiste em extrair palavras e frases do documento, e aplicar o serviço de hiperligações para determinar, a partir de informação na base de dados, quais as hiperligações que podem ser aplicadas ao documento. Esta funcionalidade foi implementada no módulo para processamento de documentos PDF e estendida às hiperligações baseadas em citações,

extraídas na parte final do documento PDF, onde geralmente está a bibliografia ou lista de referências.

- Hiperligações em documentos PDF

As funcionalidades suportadas pelo módulo HTML, foram estendidas ao formato PDF, dando origem a um módulo específico.

O formato PDF demonstrou ser complexo (ver em 3.1.2.2) para o tipo de funcionalidades que foram implementadas. É um formato optimizado para a visualização. As hiperligações estão associadas às coordenadas na página, e não a palavras ou frases. Como consequência, é necessário um profundo domínio do formato para a criação de estruturas necessárias à adição de hiperligações associadas a palavras ou frases [SH1998b].

Apesar dos inconvenientes do formato PDF, este tem a seu favor o grande número de jornais electrónicos que usam o formato, cerca de 80% [SH1998b]

Demonstradores, opiniões e críticas dos utilizadores

Para que as funcionalidades de hiperligação do serviço fossem testadas, foram criados três demonstradores contendo literatura nas áreas das Ciências Cognitivas, Biologia e Ciências da Computação.

Graças a estes demonstradores foi possível efectuar inquéritos aos utilizadores no sentido de se averiguar a fiabilidade, facilidade de utilização, e propostas para melhoramentos. De entre as várias formas de contacto com os utilizadores, foi elaborado um inquérito (ver anexo A) sobre a demonstração das hiperligações baseadas em citações, do *Open Journal* de Ciências Cognitivas. O questionário esteve disponível numa página *Web*.

Dos testes efectuados pelos utilizadores, surgiram resultados dos quais alguns dos mais importantes são:

(a) Os utilizadores são bastante exigentes - maior rapidez e mais hiperligações.

(b) Hiperligações directas – desnecessário saltar para o fim do documento (lista de referências), para depois saltar novamente para o documento citado.

(c) Ícones que discriminem o tipo de destino (apenas *abstracts* ou texto integral).

(d) Tecnologias transparentes para os utilizadores, estes não deverão ter de instalar nenhum *software* para o acesso ao serviço, ou alterar parametrizações do computador.

(e) As ligações ajudam a navegação, mas os utilizadores também necessitam de orientação.

Resultados do projecto e continuação do trabalho desenvolvido

Este projecto foi a primeira demonstração em larga escala de hiperligações baseadas em citações existentes em documentos. Como resultado, e segundo os seus utilizadores, foi produzido com sucesso *software* para serviços do género.

Os factos de nem todos os cenários aplicacionais terem sido totalmente explorados, e de ter sido desenvolvido *software* bastante diverso, constituíram algumas das limitações do projecto.

Os editores puderam constatar a importância das citações, como hiperligações, mesmo não sabendo como fazê-las.

Não houve o envolvimento suficiente no projecto por parte dos editores.

Para projectos futuros, fica o dilema, continuar com um formato estático como o PDF, ou migrar para formatos mais flexíveis e menos complexos. Na base deste dilema estão dois pontos de vista [SH1998b], colocados pelas pessoas que participaram no desenvolvimento.

- O PDF está bastante optimizado para a apresentação, mas provou ser um formato bastante complexo, o que coloca bastantes dificuldades na adição de hiperligações.
- A opinião daqueles que desenvolveram os ambientes de publicação é reticente relativamente ao uso de bases de dados de hiperligações externas. Em sua vez preferem a geração de documentos a partir de *scripts* específicos.

Por outro lado, muitos dos actuais projectos em curso envolvem o formato PDF, e os editores dominam as ferramentas deste formato. Isto indica que ainda não existem todas as condições para que os jornais científicos, em formato electrónico, sejam distribuídos como recursos dinâmicos, num formato nativo da *Web*. Mas tudo isto pode alterar-se rapidamente, pois os editores já estão mais familiarizados com os vários formatos multimédia, e sentem-se atraídos pela facilidade de reutilização da informação possível com o SGML (ver em 3.1.1.1). Alguns dos editores já têm planos para gerar páginas no formato HTML, para a *Web*, a partir de documentos SGML. Sendo assim, rapidamente os editores poderão começar a utilizar o novo descendente do SGML, o XML (ver em 3.1.1.3).

No sentido de dar uma continuação ao trabalho desenvolvido neste projecto, foi iniciado em Setembro de 1999 o projecto *OpCit - Open Citation* .

1.2.3 Projecto *OpCit*

O *OpCit* [SHLC2000] é um projecto de investigação e desenvolvimento, que teve o seu inicio oficial, em Setembro de 1999. É um dos projectos que a *National Science Foundation - NSF*[1] e a JISC[2] *International Digital Libraries Research Programme* têm em conjunto. Este projecto tem os seus centros de investigação e desenvolvimento em *Southampton University* no Reino Unido, *Cornell University* e *Los Alamos National Laboratory* nos Estados Unidos.

[1] http://www.nsf.gov

[2] http://www.jisc.ac.uk

Objectivos do projecto

O *OpCit* tem como missão três objectivos:

- A dimensão - hiperligar os 100.000 artigos do arquivo electrónico de Física de *Los Alamos*.
- A interoperabilidade – para desenvolver e integrar uma família de ferramentas genéricas de ligação de documentos, tanto para interfaces de autoria como para interfaces de leitor, para que seja facilitada a adopção por outros arquivos, serviços e editores.
- A universalidade – para promover o poder desta nova forma de navegação em documentação científica (jornais científicos), e induzir os autores de outras áreas a criarem arquivos, como o de *Los Alamos Eprint Archive*[1].

  Esses arquivos poderão abranger todas as disciplinas, estar espalhados pelo mundo e todos eles interligados.

Com esta parceria, as três universidades pretendem estudar as tecnologias necessárias à criação de um protótipo capaz de estabelecer ligações entre o material escolar, das mais diversas disciplinas existentes em repositórios, começando por *Los Alamos*.

Descrição do projecto

O *Eprint Archive* do *Los Alamos National Laboratory* – LANL, é um notável repositório público, pela sua dimensão, da literatura actual em Física. Está a tornar-se rapidamente na primeira opção de consulta dos 35.000 utilizadores diários da comunidade da Física, a nível mundial. O gráfico a seguir apresentado reporta a evolução dos acessos ao arquivo até Junho de 2000[2].

[1] http://arxiv.org e também http://xxx.lanl.gov

[2] Pode ser obtido um gráfico actualizado a partir do endereço http://xxx.lanl.gov/cgi-bin/show_weekly_graph

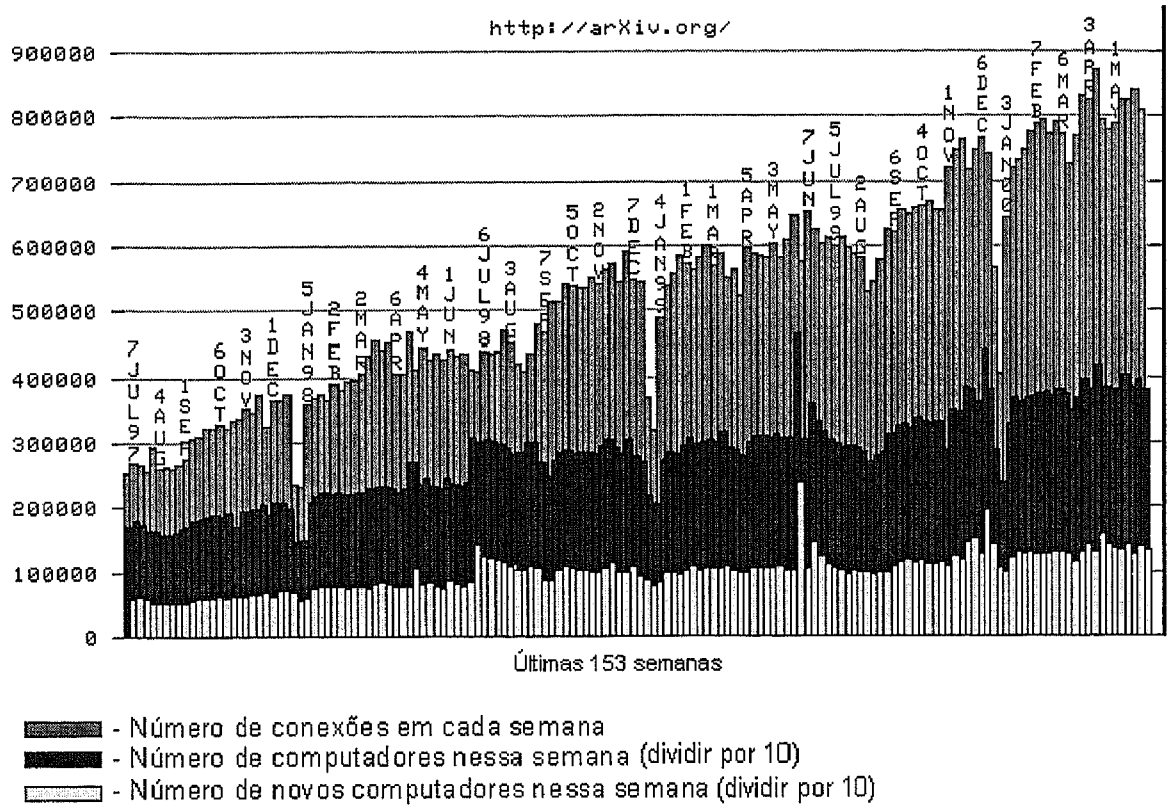


Figura 9 - Acesso ao arquivo de *Los Alamos*

Actualmente, não só existem ainda formas de tornar o arquivo muito mais poderoso e útil aos seus utilizadores, assim como estender o mesmo serviço para outras disciplinas para além da Física no arquivo de *Los Alamos,* ou em quaisquer outros arquivos com a mesma arquitectura. A forma encontrada para melhorar as actuais funcionalidades do LANL e a sua extensão às restantes áreas da ciência e ensino, consiste nas hiperligações com base nas citações.

O projecto *OpCit* faz uso da experiência adquirida nos anteriores projectos *Open Jornal Project*[1] e *CogPrints*[2] na Universidade de *Southampton*. Nestes projectos, foram desenvolvidas ferramentas necessárias à ligação dos documentos, utilizando as citações neles contidas. Estas ferramentas foram desenvolvidas para estes projectos, cujo repositório de documentos é muito mais pequeno que o de *Los Alamos*, mas interdisciplinar. Agora podem ser reutilizadas, e melhoradas no sentido de as tornar mais potentes, e permitir a completa interligação dos documentos do LANL. Para que haja de facto benefício com as ligações baseadas em citações, tanto o interface das ferramentas de autor como de leitor, do LANL, têm de se tornar universais para todas as disciplinas.

Sabe-se, à partida, que as características [SHLC2000] [RDC1997] do recurso ideal, *on-line*, para estudantes e cientistas seriam: a existência de documentos, com todo o tipo de conteúdo

[1] http://journals.ecs.soton.ac.uk/

[2] http://cogprints.soton.ac.uk/

(áudio, vídeo, base de dados, etc.) de todas as áreas, sistematicamente interligados, facilmente acessíveis e racionalmente navegáveis por qualquer investigador de qualquer ponto do mundo. Mas a opção foi considerar apenas artigos científicos, por haver consciência da inviabilidade [SHLC2000] dessa concretização a curto prazo, no contexto do projecto *OpCit*. Por outro lado, seria necessário ter todo o material disponível numa forma única, em formatos digitais.

O arquivo LANL, contém uma substancial quantidade de literatura em Física, Matemática e Ciências da Computação. O *Computing Research Repository* – CoRR[1] pode ser directamente acedido pelo LANL. No entanto, os textos estão em diversos formatos: HTML, *LaTeX*, PS, PDF, HTML e DVI, e o primeiro problema que surge, é como conseguir a navegação pelos documentos dos arquivos, sem haver mudanças de ferramentas, ambientes, ou outro tipo de interrupção, isto é, num ambiente integrado.

Nos recentes projectos *Open Journal* e *CogPrint*, já referidos, foram utilizadas com sucesso as "hiperligações" inseridas pelos próprios autores através das suas citações. O trabalho já desenvolvido permite que de uma forma pacífica, seja resolvido o problema de automaticamente reconhecer e estabelecer a ligação para o conjunto diverso de formatos de citações existente no meio do texto. Este mecanismo, designado por *Citation Linking* [SH1998b], é baseado num componente de *software* designado por *Citation Agent*. Este agente reconhece as referências do documento e cruza essa informação com outra, resultado de uma pré indexação existente numa base de dados de resumos e hiperligações. Todo este processo decorre em tempo real.

Da experiência adquirida com estes anteriores projectos, a opinião dos utilizadores foi bastante favorável relativamente à navegação baseada nas citações dos textos, porém, e dado que apenas disponibilizaram os resumos em vez dos textos de uma forma integral, houve alguma frustração por esse facto. Essa foi uma das conclusões [SH1998b] [SHLC2000] que a experiência permitiu obter: disponibilizar apenas os resumos reduz o enorme potencial deste tipo de ligação de documentos. Para uma plena exploração desse potencial, é necessária a sua realização com os textos completos em vez de apenas os resumos. É isso que no projecto *OpCit* se pretende fazer [SHLC2000], graças ao arquivo LANL.

Resultados

O projecto *OpCit* está ainda em curso, portanto ainda não existem conclusões. Porém, o facto de o acesso ao arquivo de *Los Alamos* ser um serviço que não tem qualquer restrição de ordem financeira, com acesso integral, público e global, constitui logo à partida uma vantagem [SHLC2000] comparativamente a outros serviços do género. O *Commercial Journal Publishers*[2], em conjunto com o serviço de índices e *abstracts* proposto pelo *Institute for Scientific Information - ISI*[3], são sistemas que interligam documentos científicos. Estas iniciativas são muito fechadas e severamente limitadas pelo facto de impossibilitarem a

[1] http://www.acm.org/repository/

[2] http://www.lib.uchicago.edu/Annex/pcaplan/reflink.html

[3] http://alerting.isinet.com/isi/products

navegação livre pelos documentos e suas citações, sem que antes se subscreva o serviço numa modalidade do tipo *pay-per-view*.

Este repositório, o *arXiv*, é actualmente um dos signatários da *Open Archives Initiate*[1] que tem em definição normas de interoperabilidade entre arquivos disponíveis *on-line*.

1.2.4 A gestão da informação com base nas citações

A utilização das citações em documentos electrónicos como hiperligações, conferem um valor acrescentado, a possibilidade de instantaneamente obter o documento citado. Estas hiperligações podem funcionar como uma forma poderosa de relacionar documentos e, assim, interligar a informação independentemente da data da sua publicação, esteja ela onde estiver, não só a possibilidade de registar a referência bidireccionalmente (o que cita e o citado) constitui uma vantagem relativamente ao papel e possibilita a navegação no tempo, para trás (nos documentos citados) como para frente (os documentos que citam o corrente), tornando-se num poderoso mecanismo de pesquisa de literatura.

Pesquisa bibliográfica

Utilizando as referências bibliográficas e citações existentes nos textos, estabelece-se uma ligação, eficiente, coerente e interdisciplinar entre os documentos [RDC1997]. Desta forma aproveita-se o trabalho que os próprios autores tiveram ao introduzir as suas citações.

Os autores referenciam outros trabalhos como forma de sustentar muitas das afirmações e pressupostos, reutilizando trabalho efectuado anteriormente [EG1994]. Havendo uma referência a trabalhos anteriormente efectuados, que regra geral são da mesma área, torna-se possível estabelecer uma linha no tempo que intercepta todos os trabalhos dessa área. Como consequência, benéfica, a pesquisa bibliográfica é facilitada. Recorrendo a serviços como os atrás referidos, que contêm uma base de dados de citações, torna-se possível efectuar pesquisas como: Que continuidade foi dada a um trabalho iniciado por alguém? Quais os trabalhos no qual o presente se baseia? Quais são todos os trabalhos da uma área? entre muitas outras possíveis pesquisas.

Pelo facto de serem os próprios autores a introduzirem estas "hiperligações", e estas serem susceptíveis de automaticamente serem reconhecidas e de imediato arquivadas numa base de dados de citações, torna-se possível reduzir imenso o tempo que ocorre desde a publicação até à referência do trabalho em catálogos.

[1] A OAi – *Open Archive Initiative* promove e encoraja o desenvolvimento de soluções que permitam ao próprio autor a colocação dos seus trabalhos no arquivo on-*line*. Em Outubro de 1999, na Convenção de Santa Fé, foram definidos os princípios organizacionais e especificações técnicas para facilitar o mínimo mas potencialmente grande nível funcional de interoperacionalidade entre arquivos de documentação académica. Em Setembro de 2000, decorreu um encontro técnico, do qual se espera um acordo mais estável, com uma especificação mais aprofundada. Os resultados só estarão disponível no fim de 2000. Mais informação pode ser obtida a partir do endereço http://www.openarchives.org

Estas vantagens na utilização das citações em documentos, não são propriamente algo de novo, pois desde sempre a navegação manual, no papel, foi possível e utilizada. O que este tipo de serviços tem de inovador, é o facto de permitirem aumentar extraordinariamente esse potencial, reduzindo os tempos e aumentando os horizontes sobre o que é publicado.

Com a implementação deste tipo de serviços, poderão ser criados outros níveis de funcionalidades como, por exemplo, o aviso por mensagem de correio electrónico para alguém interessado na evolução de um dado trabalho. Por outras palavras, sempre que surja na base de dados de citações uma nova referência, que se enquadre num conjunto de parâmetros predefinidos, de imediato poderá ser enviado um *email* alertando sobre o facto.

Análise das citações

Para além dos benefícios para a pesquisa bibliográfica, uma análise dos conteúdos de uma base de dados de citações pode fornecer informação sobre a evolução do trabalho desenvolvido por instituições, grupos de trabalho e pessoas. Estes resultados da análise das citações, têm sido utilizados de uma forma directa para a tomada de decisões de contratações, promoções e salários [LLH1990] em departamentos académicos. Muitos responsáveis de departamentos académicos têm todos os anos a ingrata tarefa de decidir os aumentos salariais. O recurso à análise do número de citações e onde são citados, pode contribuir para a tomada dessas decisões [NW1975] [PHG1993]. As medições das citações têm sido também utilizadas para a criação de rankings dos mais citados de um departamento ou instituição.

A análise de citações tem também outros efeitos indirectos na avaliação das instituições e pessoas, através da avaliação das revistas científicas mais citadas. O *Journal Citation Reports*[1], publica todos os anos, pelo Verão em CD-ROM[2] e na *Web*[3], uma avaliação das revistas científicas, baseado em critérios como o factor de impacto (média do número de citações por artigo publicado na revista [EG1972]), o número de anos abrangidos pela metade mais recente das citações da revista, entre outros. A avaliação de instituições e pessoas, pode também ser efectuada através do prestígio das revistas em que os seus artigos são publicados. Essas medidas são ainda frequentemente utilizadas por bibliotecários aquando da necessidade de tomada de decisão do cancelamento ou subscrição de publicações.

Contudo, muitos dos estudos efectuados são críticos relativamente às metodologias utilizadas nas avaliações, recorrendo a resultados da análise das citações. Segundo esses estudos, estas são sensíveis às modas repentinas e transitórias, às fraquezas e às tendências populares da ciência [DL1989], e que simples contagens de citações são frequentemente utilizadas sem uma correlação desses valores com outra informação relevante [ASTB1993]. Por outro lado, o uso dos existentes índices de citações tende a enfatizar o papel das revistas que constam dos índices, desvalorizando todas as outras formas de publicação. Porém, se a base de dados de citações for universal e de livre acesso, estas críticas perdem a razão de existência [RDC1997]. A universalidade da base de dados, certamente valorizará igualmente todas as formas de publicação, permitindo que o impacto de trabalhos possa ser avaliado, sem que a inclusão ou não inclusão na base de dados interfira nessa avaliação. O facto do acesso à

[1] http://www.isinet.com/isi/products/citation/jcr

[2] http://www.isinet.com/isi/products/citation/jcroncd.html

[3] http://www.isinet.com/isi/products/citation/jcr/jcrweb.html

informação da base de dados ser livre, fará com que facilmente essa informação se correlacione com outra informação relevante para a avaliação do trabalho efectuado.

Diversos tipos de análise bibliométrica serão possíveis a partir da informação sobre as citações em base de dados [RDC1997]. Este tipo de análise é importante para uma melhor compreensão, previsão e direccionamento do desenvolvimento deste tipo de sistemas.

1.2.5 Projecto de uma ontologia de conhecimento académica

A *World Wide Web* foi o primeiro sistema global de hiperligação de documentos a emergir e fornece uma infra-estrutura rudimentar para publicação de documentos hiperligados. Este nível de sofisticação de representação por si só já constitui uma forma extremamente útil para o melhoramento, e até a transformação da publicação de trabalhos académicos [SSEMJD1999a]. Porém, e como já foi referido, a *Web* não proporciona mecanismos para estruturar, pesquisar ou analisar a informação que disponibiliza.

Serviços baseados em bases de dados de citações proporcionam apenas mecanismos de hiperligação de documentos, utilizando a informação colocada pelos autores quando fazem citações a outros trabalhos. A pesquisa bibliográfica e a análise das ligações, sendo benéficas (ver em 1.2.4), não fornecem contudo ao leitor mais nenhuma informação para além da própria ligação. Outros níveis de informação, são extremamente importantes para a fácil e rápida interpretação dos conteúdos dos documentos por parte do leitor. Para os autores, este nível de informação funciona como uma ferramenta bastante útil para a argumentação e defesa dos seus pontos de vista. A pesquisa que já era beneficiada com a simples ligação, agora, com outros níveis de informação, capazes de definir os conceitos e relações, poderá ser ainda mais facilitada.

Objectivo

Esta proposta deverá permitir a criação de ontologias suficientemente simples para o fácil entendimento, sem se tornarem simplistas. Naturalmente, esta especificação deverá ser amplamente aceite pela comunidade de investigação. Essas ontologias poderão ser depois utilizadas como mecanismos de classificação de conceitos e relações entre documentos, que permitam aos próprios autores a classificação dos documentos que publicam em repositórios.

Descrição do projecto

A proposta do projecto *ScholOnto* [SSEMJD1999b], assenta no uso de serviços de classificação para submissão de documentos a arquivos *on-line* (por exemplo: *Los Alamos Eprint Archive* e *CogPrints*). Este nível de informação associado aos documentos hiperligados, permite a criação de uma ontologia [SSEMJD1999b] capaz de reflectir de uma forma amplamente aceite, um ponto de vista sobre a conceptualização de um dado domínio ou fenómeno. Para isso, os autores desta proposta sugerem a especificação de um conjunto relativamente pequeno de conceitos e tipos relacionais que sejam capazes de representar

adequadamente a maioria das argumentações de ideias e pontos de vista. Este conjunto limitado de termos classificam tanto os conceitos presentes nos documentos como as relações entre esses conceitos. Com isto, facilita-se a adição de hiperligações de argumentação entre conceitos ou documentos.

A Figura 10 que se segue ilustra como poderá ser efectuada a classificação dos documentos e as suas relações.

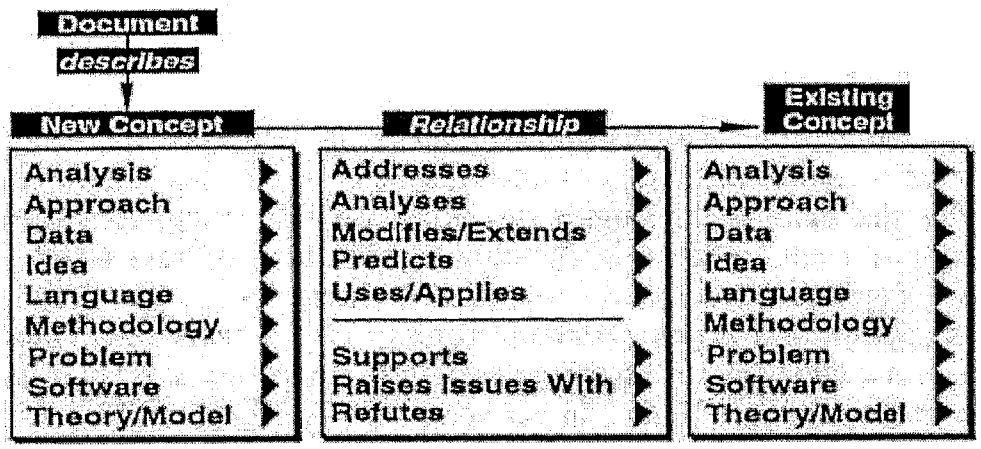


Figura 10 - Conceitos e relações [SSEMJD1999a]

Esta proposta deu origem a uma implementação. A aplicação desenvolvida usa uma ontologia e tem como infra-estrutura uma tecnologia de modelização de conhecimento, desenvolvida e testada noutros domínios, a linguagem de modelização OCML [EM1998]. Esta linguagem suporta a construção de modelos de conhecimento recorrendo a vários tipos de construção. Permite a especificação e operacionalização de funções, relações, classes, instâncias e regras. Inclui também, mecanismos de definição de ontologias. A linguagem OCML fornece o formalismo necessário à definição da ontologia que serve para o debate e interpretação. Esta ontologia é designada por *ScholOnto*.

A aplicação *WebOnto* [SSEMJD1999a] permite que num ambiente colaborativo seja possível a navegação e edição de modelos de conhecimento através da *Web*. A sua arquitectura é constituída por um servidor central e um *applet Java*. O servidor central do *WebOnto* assenta num servidor HTTP específico, o *LispWeb* [ARMR1999] e usa a OCML como linguagem de modelização.

O servidor *LispWeb*, adicionalmente ao interface HTTP que proporciona, fornece as bibliotecas dum conjunto de funções *Lisp*[1], de alto nível, necessárias à geração dinâmica de páginas HTML, de imagens e de métodos de comunicação entre servidores.

O *applet* proporciona múltiplas visualizações dos modelos OCML e formulários para criação de novas estruturas. Proporciona também funcionalidades de trabalho em grupo (*groupware*),

[1] *Lisp* é uma linguagem de programação de alto nível geralmente utilizada no desenvolvimento de sistemas de inteligência artificial

as quais suportam tanto a construção síncrona como assíncrona, por equipas de engenheiros de conhecimento.

Resultados

O texto é um meio poderoso para a publicação e discussão de ideias em detalhe. Porém, tem como desvantagem o tempo que é necessário à sua leitura e ainda o facto de ser complexa a análise computacional.

O desenvolvimento de estruturas de informação que permitam a modelização das ideias e pontos de vista constitui um nível de informação que, adicionado aos textos, lhes poderá dar um valor acrescentado bastante elevado. A ontologia *ScholOnto* especifica uma forma de estruturar essa informação adicional, a acrescentar aos textos. Os autores desta proposta defendem que uma ontologia deste tipo, facilita a compreensão dos textos, e mais importante ainda, permite a criação de serviços inteligentes, capazes de responder a questões como *"Qual o impacto da teoria T ?", definindo "impacto"* como o número de artigos subsequentes que usaram ou modificaram a teoria, número de diferentes domínios em que foi aplicada, número de problemas que trataram, etc.. A pesquisa de documentos relevantes para questões do tipo *"o método X usa o método Y, o método X foi posto em causa, o método Y também deverá ser posto em causa ?"*.

O género de estruturas de informação propostas permite o enriquecimento dos textos e das suas ligações. A pesquisa, a interpretação e a análise computacional desses textos torna-se mais fácil.

O *WebOnto*, a implementação que usa a ontologia *ScholOnto*, já permite a geração de formulários para a adição de novas instâncias das classes da ontologia. Porém, estes formulários têm alguma complexidade, os quais poderão ser utilizados apenas por pessoas mais especializadas na área da gestão do conhecimento. Existe a necessidade de criar outros formulários que ajudem o utilizador no processo de criação de novas estruturas de classificação. Será também necessária a criação de interfaces que permitam ao autor publicar e classificar os seus artigos. Estes interfaces deverão facilitar este processo de classificação, permitindo a reutilização de conceitos existentes. Pois os autores não são especialistas em gestão do conhecimento ou bibliotecários, que conhecem todas as técnicas de classificação.

Uma aplicação como o *WebOnto*, poderá ter um papel determinante na submissão e disseminação de documentos em arquivos como o de *Los Alamos Eprint Archives* ou *CogPrints*.

1.3 Serviços de Cooperação

Outra forma que frequentemente se utiliza para encontrar a informação pretendida são os grupos de notícias (serviço de "*news*") e as listas de distribuição de mensagens electrónicas. São sistemas onde os utilizadores podem partilhar o seu conhecimento. Trata-se de sistemas colaborativos com bastante interesse. No entanto, este tipo de solução é muito dependente do nível de conhecimentos do grupo. Quanto maior for o nível de conhecimentos dos membros do grupo, mais valiosa é a informação. Estas soluções têm bastante interesse pela colaboração

entre os utilizadores, e o seu relacionamento no que toca à partilha dos conhecimentos. Sistemas como estes demonstram a viabilidade dos sistemas colaborativos, como forma de vencer alguns dos desafios da gestão da informação. Apresentam-se nesta secção alguns trabalhos em que a colaboração, por vezes com papéis diferenciados, é importante.

1.3.1 *Theseus* – Um modelo de conectividade

O nome dado ao modelo, *Theseus* [RHRS1993], é o de uma personagem da mitologia grega que derrotou o Minotauro, uma criatura meio homem meio touro. Segundo a lenda, o Minotauro estava preso num labirinto tão complexo que até o próprio arquitecto se perdeu lá dentro. *Theseus* entrou no labirinto, para matar o monstro. Para não se perder, utilizou um fio para marcar o caminho de regresso até à saída. *Theseus*, depois de matar o monstro, esse o seu objectivo, só poderia sair do labirinto com a preciosa ajuda do fio com que tinha marcado o caminho de regresso até à saída.

Objectivos

Da mesma forma que a personagem da mitologia grega conseguiu entrar no labirinto sem se perder, com o modelo *Theseus*, os seus autores, *Harden e Stringer,* pretendem que o utilizador de uma vasta colecção de objectos documentais possa efectuar um percurso coerente, sem se dispersar por caminhos menos produtivos e acabar por se "perder" na imensidão de informação. Sendo um sistema vocacionado para o ensino o utilizador, enquanto aluno, segue um caminho ditado pelo tutor, o autor do caminho do assunto.

Com base neste modelo deverá ser possível a criação de serviços [RS1992] que, graças ao espírito de colaboração entre os seus utilizadores acrescentem um nível de informação estruturada, que funcione como um "fio" capaz de orientar o utilizador.

O modelo deverá permitir formatar e organizar novas formas de informação de uma forma controlada e coerente, tornando o meio em algo dinâmico, com a sua estrutura e conteúdos em constante mutação.

Um dos requisitos para este modelo é o de permitir a rápida compreensão e a fácil exploração da estrutura da informação. De uma forma linear ou sequencial, deverá ser possível navegar pelos objectos documentais, as entidades que compõem a base de recursos documentais.

Descrição do modelo

O modelo *Theseus*, de *Harden e Stringer*, propõe a existência de um conjunto de informações estruturadas que ajudem o utilizador (enquanto aluno) a não se perder no labirinto de hiperligações possíveis, que unem todos os objectos documentais que tem à disposição no repositório. Essa informação estruturada que os autores (utilizadores enquanto tutores) propõem tem exactamente a mesma função do fio utilizado por *Theseus*.

A metodologia utilizada parte da constatação que a existência de grandes quantidades de informação de facto oferece imensas oportunidades para a aprendizagem e valorização pessoal, mas o facto por si só não é suficiente para garantir a aquisição efectiva de

conhecimentos. Encontrar um caminho rápida e facilmente, através de tal volume de informação, até justamente aquela que é relevante para as necessidades e interesses do utilizador, e então encontrar o caminho de regresso à área de estudo da qual o utilizador se tinha perdido para tão longe, coloca grandes problemas.

É um modelo vocacionado para o ensino, que propõe uma possível forma de resolver muitos dos problemas da conectividade da informação. Assenta num modelo de hipermédia, cujo sucesso a longo prazo depende da utilidade para as instituições e pessoas, que o utilizam e ao mesmo tempo contribuem para a evolução da base de recursos. A sua eficácia, e seus atractivos, tanto para professores como alunos, fomenta a riqueza e diversidade de uma colecção de objectos hiperligados.

A um nível operacional, o modelo caracteriza-se por ter duas camadas distintas de informação, a base de recursos - *Mediabase* e os caminhos de assuntos - *Subject Paths* [RHRS1993]

- *Mediabase*

Esta camada é constituída por uma base de objectos. Cada item discreto dessa base de objectos, é uma entidade [RHRS1993]. As entidades tanto podem ser um simples ficheiro de texto, como uma imagem ou até um qualquer sistema de representação virtual, de um complexo sistema, ao qual podem ser aplicadas também as mais complexas operações de simulação, decomposição, etc.. Uma entidade deve ser interactiva. Os termos dessa interactividade devem ser definidos de uma forma intrínseca, e não por um conjunto de regras genéricas aplicadas a toda a *Mediabase.*

As entidades que constituem esta camada, a *Mediabase,* devem ser tão categóricas e objectivas quanto for possível. Quaisquer dados de informação, fornecidos pela entidade, deverão ser factos genericamente aceites, acima de qualquer dúvida quanto à sua veracidade ou valor científico.

- Caminho de Assunto *(Subject Path)*

Os caminhos de assunto consistem em informação exterior às entidades. Esta informação permite que sejam criados caminhos, "fios", por entre as entidades, que dêem ao utilizador a orientação necessária à rápida e fácil compreensão das matérias abordadas. O caminho de assunto representa uma argumentação ou ideia do seu autor, portanto, a importância das entidades é algo que não é previamente definido, mas ajustado a um dado contexto. O papel de uma entidade é o de uma ilustração ou até simplesmente o de um adorno.

Um leitor poderá a qualquer momento interrogar uma entidade, no sentido de determinar quais os caminhos que a referenciam, e assim tomar contacto com as diferentes argumentações que a interceptam. Deste modo, o leitor ao seguir um determinado caminho poderá desviar-se ortogonalmente para outros.

É a combinação destas duas camadas distintas que constitui o dinâmico e evolutivo Hipermeio *(Hipermedium)* [RHRS1993]. Através dos caminhos de assuntos que cruzam o Hipermeio num dado momento, o utilizador pode efectuar uma navegação pela *Mediabase,* e percorrer os

pontos de vista, opiniões subjectivas e pessoais daqueles que contribuíram para a criação do Hipermeio.

Resultados

Apesar da simplicidade do modelo, este fornece uma estrutura conceptual para que seja possível lidar com grandes quantidades de informação. Toda a estrutura do Hipermeio está em constante mutação. É capaz de exprimir tanto a experiência de uma comunidade (*Mediabase*), como uma experiência individual e única (Caminho de Assunto).

Os utilizadores, como autores, podem marcar o seu ponto de vista ou tese, ou então, como leitores, podem seguir e reflectir sobre o ponto de vista ou tese de outra pessoa. O modelo dá ao utilizador, enquanto leitor, o poder de "tocar no texto" [JDB1991], isto é, introduzir na obra do autor as suas anotações e ligações e disseminá-las livremente.

Outra característica muito importante deste modelo é a reutilização dos recursos. Estes podem ser utilizados inúmeras vezes (em mais do que um caminho), em contextos completamente diferentes.

O *UWA Theseus Project*[1] é um dos projectos em curso na *University of Western Australia – UWA,* que em conjunto com a *Liverpool John Mores University*, no Reino Unido, desenvolve este modelo. O modelo tem sido utilizado num protótipo de um centro cultural virtual, o *Virtual Cultural Centre- VCC*[2], um projecto conjunto entre *Library and Information Service of Western Australian* – LISWA, o *Western Australian Museum, Western Australian Gallery* e a *University of Western Australian.*

1.3.2 Centro de Ensino Cooperativo

O modelo *Theseus,* anteriormente descrito, propõe uma estratégia para a concepção de serviços que permitam aos seus utilizadores a rápida compreensão e fácil exploração de uma estrutura constituída por grandes volumes de informação.

O presente modelo tem como ponto de partida essa proposta, estendendo-a em alguns dos seus conceitos.

Objectivos

O modelo proposto para a concepção de um Centro de Ensino Cooperativo [CMO1999], deverá permitir o desenvolvimento de bases de dados distribuídas, com conteúdos multimédia para o ensino e investigação. Esta proposta pretende tirar proveito das excelentes oportunidades para o ensino e investigação, devido à existência de grandes quantidades de

[1] http://www.arts.uwa.edu.au/TheseusWWW/Theseus.html

[2] http://www.arts.uwa.edu.au/External/TheseusWWW/VCC.html

informação na *Internet*, permitindo a pesquisa de informação relevante e útil, assim como a navegação coerente e controlada.

Descrição do serviço

Com raízes no modelo *Theseus*, que à sua semelhança propõe a interligação de objectos documentais através de caminhos temáticos, esta proposta permite a representação de várias ideias ou anotações, associadas aos conteúdos da base documental.

O modelo *Theseus* tem como objectivo a definição de um ambiente de ensino à distância convencional no método de apresentação sequencial da informação, embora permitindo facilidades de comutação entre "cursos" (caminho de assunto) e a reutilização de objectos. O Centro de Ensino Cooperativo é um modelo com vocação para a auto formação, contrariamente ao modelo *Theseus*, vocacionado para a orientação baseada em "tutores", e define uma estrutura funcional baseada em quatro níveis de informação, que compõem o Hipermeio [RHRS1993] neste caso.

O primeiro nível contém todos os objectos que constituem o repositório, a informação primária. Estes objectos, recorrendo a um segundo nível, são agregados em unidades lógicas, o documento multimédia. Note-se que os documentos que constituem este nível podem conter vários tipos de meios: texto, animações, vídeo, ficheiros *MS-PowerPoint*, etc., isto é, são multimédia, contudo é lhes conferido o estatuto de "monomédia" para reforçar a ideia de documento lógico e a reutilização de conteúdos. Os objectos do primeiro nível de informação, os conteúdos, deverão estar catalogados num sistema de catalogação.

O terceiro, o caminho de assunto, na realidade são hiperligações, que permitem o estabelecimento de um caminho temático através da informação dos dois níveis inferiores. Trata-se do mesmo conceito introduzido por *Harden* e *Stringer* no modelo *Theseus*.

O quarto nível, a camada de anotações, complementa o anterior apoiando o aluno no percurso indicado. São anotações que podem exprimir opiniões e pontos de vista pessoais do autor que as inseriu. Estas anotações estão associadas aos nós do Hipermeio, os documentos do segundo nível.

A combinação destes quatro níveis de informação resulta no Hipermeio [RHRS1993], que aplicado a este modelo concreto tem o seguinte diagrama:

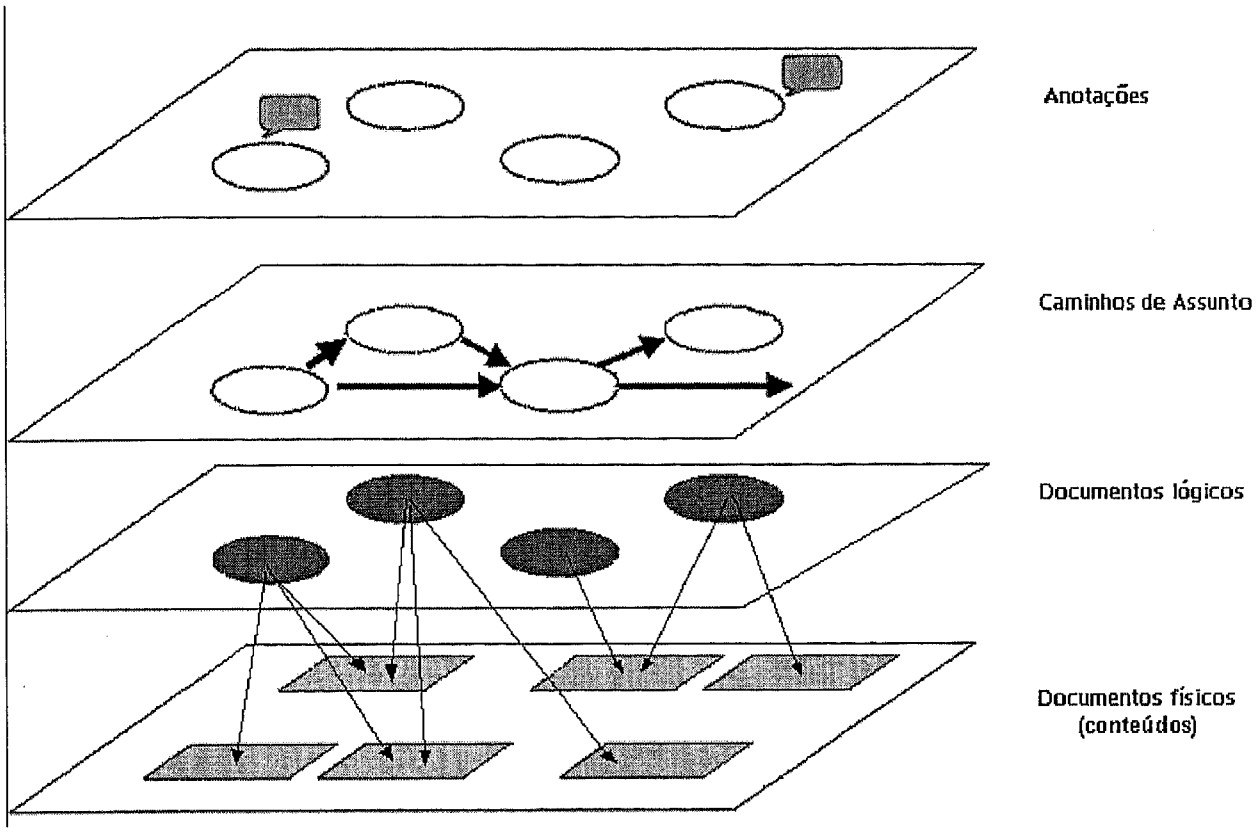


Figura 11 – Modelização do conhecimento utilizado no Centro de Ensino Cooperativo

O sucesso deste modelo, como um sistema de ensino, depende exclusivamente do forte envolvimento dos seus utilizadores, multiplicando as entidades [RHRS1993] e os Caminhos de Assuntos [RHRS1993] que lhes estão associados.

Os direitos de autor relacionados com as anotações inscritas nos caminhos de assunto, estão protegidas pela natureza do Hipermeio, assim como qualquer entidade que seja extraída de sistema, perde muitas das suas qualidades [RS1992].

Esta proposta considera ainda a existência de mecanismos de controlo que façam a contabilização de acessos, e até a sua aplicação para fins comerciais, a facturação do serviço. A segurança que estes mecanismos fornecem é a segurança necessária aos serviços de edição e distribuição dos conteúdos. A análise estatística é outra funcionalidade possível de obter a partir desses mecanismos.

Resultados

A proposta deste modelo, relativamente à proposta que lhe deu origem, introduz alguns aspectos os quais podem ser considerados inovadores. No modelo *Theseus* é proposta a existência de dois níveis de informação, este trabalho faz uma divisão do primeiro em dois subníveis. A subdivisão do primeiro nível dá origem a dois subníveis que em conjunto, têm precisamente a mesma funcionalidade daquele que lhes deu origem. O nível inferior da

mediabase descrito no modelo *Theseus* contém as entidades (físicas), os documentos do repositório. O segundo nível do presente modelo, contém as entidades (lógicas), elas próprias constituídas pela agregação dos elementos físicos, os documentos do repositório. Esta agregação dos elementos do repositório (objectos monomédia) em objectos lógicos (objectos multimédia), poderá ser considerada com um factor de redução da complexidade dos caminhos de assunto que cruzam o hipermeio, visto que, a própria agregação de alguma forma já hiperliga os diversos elementos que constituem o documento lógico. A camada adicional, o quarto nível do presente modelo, introduz a possibilidade de serem introduzidas anotações que tanto podem ser simples comentários informais de âmbito pessoal, ou críticas formais com algum tipo de argumentação. Estas anotações indiscutivelmente favorecem a compreensão dos assuntos representados pelos respectivos caminhos.

Esta proposta aproxima o modelo original de uma possível implementação do modelo, contemplando aspectos bastante práticos.

1.3.3 *Pharos* – infra-estrutura *colaborativa* para partilha de conhecimento

O serviço *Web* como meio disseminador de informação é amplamente aceite e utilizado. No entanto, encontrar a informação que se pretende, aquela que é relevante, evitando tanto o ruído como o silêncio (ver em 1.1.3) é um exercício bastante difícil. Os actuais mecanismos de pesquisa são muito orientados para a pesquisa por palavra-chave, e pouco para os conceitos, entre outras limitações [VBOD1999]. Embora, e tal como foi abordado (ver em 1.1.3), existam motores de pesquisa temáticos, ou recorrendo a técnicas mais elaboradas e até utilizando inteligência artificial, que possam aproximar-se da pesquisa através de conceitos.

Os autores desta proposta, consideram que existe a necessidade de haver serviços que, por um lado, integrem funções de motores de pesquisa, e que ao mesmo tempo forneçam informação valiosa como aquela que é trocada em grupos de notícias ou listas de distribuição de mensagens electrónicas [VBOD1999].

Objectivos

O serviço *Pharos* [VBOD1999] tem como objectivo permitir que os seus utilizadores facilmente encontrem a informação que procuram na *Web*. *Pharos* é um serviço que está a ser desenvolvido no sentido de permitir aos seus utilizadores a partilha entre si, do conhecimento que têm acerca de documentos que considerem interessantes. Portanto, é um serviço que assenta numa infra-estrutura colaborativa, que permite a indexação e avaliação dos documentos em tópicos específicos.

Descrição do serviço

Este serviço pressupõe que os seus utilizadores, no decorrer do trabalho de pesquisa que efectuam, sempre que encontrem um documento, na *Web*, que considerem importante, lhe associem uma anotação no "canal" [VBOD1999] apropriado. Um "canal", consiste numa base

de dados de anotações dedicadas a um tópico específico. Estes canais funcionam autonomamente e podem estar distribuídos por rede de computadores. Assim, em vez de se terem grandes quantidades de informação concentrada, sobre vários tópicos, obtém-se uma solução mais flexível e eficiente [VBOD1999].

A função da anotação é a de descrever o documento ao qual se refere. A publicação da anotação tem de ser efectuada explicitamente pelo utilizador. Neste serviço, uma anotação consiste numa estrutura de dados à qual está associada uma URL. As anotações têm uma estrutura que depende do canal a que pertencem, contém informações como: o título, um coeficiente (avaliação do documento), um comentário e algumas palavras-chave. O coeficiente consiste num valor real de –1 a +1 que traduz a apreciação do utilizador sobre o documento. No entanto, este coeficiente no interface com utilizador toma valores discretos representados por ícones ou texto, definidos pelo administrador do canal. O comentário é livre, e consiste num pequeno texto sem qualquer rigidez de estrutura que o utilizador pode introduzir. Contrariamente, as palavras-chave podem ser escolhidas de um conjunto de palavras organizadas de uma forma hierárquica e extensível. Este conjunto de palavras pode ser encarado como um simples *thesaurus*. Este *thesaurus*, proporciona um vocabulário comum com uma estrutura em árvore que os utilizadores podem utilizar para efectuar as anotações dos documentos. Desta forma, evita-se o uso de palavras chave semanticamente similares mas que lexicalmente são diferentes, por exemplo, *browser* e navegador. Obviamente que o *thesaurus* permite uma pesquisa de documentos bastante mais simples. Por outro lado, este permite obter uma vista global sobre a organização dos tópicos do canal. A estrutura deste *thesaurus* só pode ser alterada por utilizadores autorizados.

O serviço *Pharos* prevê ainda que existam classes de canais para a melhor adaptação a necessidades específicas. Todas as classes de canais derivam de uma classe base designada por *BasicChanel* [VBOD1999] que já incorpora as funcionalidades básicas para fins genéricos.

Ainda outro aspecto importante acerca das anotações, é a definição pelo utilizador, no momento da sua criação, de uma data em que esta se torna obsoleta.

Com a utilização do serviço, o número de anotações crescerá, naturalmente isso tornar-se-á mais difícil para os utilizadores explorarem tão grande quantidade de anotações. Para contornar este inconveniente, o excesso de informação, existe um processo que de uma forma personalizada, de acordo com o perfil de utilizador, gera uma síntese de todas as anotações relacionadas com um único documento. Esta síntese é uma recomendação. As recomendações são obtidas a partir do conjunto de anotações ao documento feitas por qualquer utilizador. Sendo assim, é necessário que possam ser personalizadas de acordo com um conjunto de parâmetros. Se não for assim, as recomendações tornam-se em simples médias, onde as opiniões em minoria são diluídas na imensidão de anotações. Por outro lado, existem determinados utilizadores que pelo facto de serem especialistas na área, deverão ter opiniões com um peso maior, expressas nas anotações. A poluição é outro alvo da personalização da obtenção das recomendações, por exemplo, evitar que certos utilizadores que para fins comerciais possam inundar o serviço com anotações que favorecem os seus produtos ou tecnologias. A estas anotações, é dado um peso nulo para a obtenção da recomendação. Isto pode ser feito explicitamente pelo utilizador, ou então, automaticamente recorrendo a um algoritmo de correlação [VBOD1999].

Ainda para a obtenção da recomendação, os restantes atributos das anotações também são processados, como o coeficiente, título, comentários, palavras-chave, usando para isso algoritmos específicos [VBOD1999].

O navegador Assistente é o componente do sistema que funciona como um *proxy*[1] entre o navegador *Web* e os servidores (*Web e Pharos)*. É também um navegador, com *plugins*[2] que lhe conferem a capacidade de funcionar como *proxy*. Um desses *plugins* tem a designação de *Pluxy* [OD1998] e funciona como *proxy* HTTP. Outros *plugins* funcionam como *proxy* de pedidos de anotações. São classes *Java* que remotamente invocam os seus pedidos no servidor do canal, usando para o efeito métodos RMI[3]. A Figura 12 ilustra esta interacção entre os diversos elementos do sistema. O servidor do canal também aceita pedidos directamente, mas à custa da perda de algumas das funcionalidades.

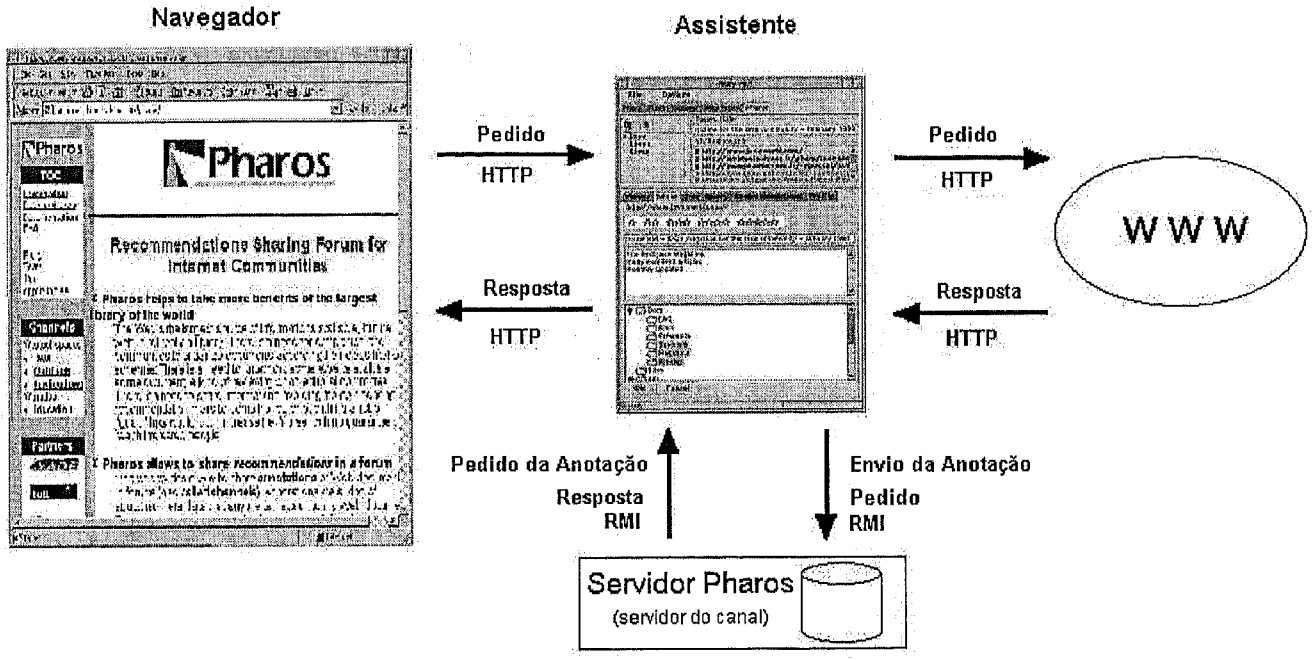


Figura 12 - Interacção Navegador/Assistente/Servidor [VBOD1999].

Utilização do serviço

Enquanto o utilizador vai navegando na *Web*, as anotações relacionadas com o documento visualizado no Navegador, são exibidas num navegador Assistente.

[1] Um *proxy* é intermediário entre clientes e servidores. Consiste num programa ao qual são feitos pedidos (por exemplo, páginas HTML). Depois de lhe ser feito um pedido, o *proxy* submete esse pedido ao servidor, o servidor responde-lhe e ele envia a resposta ao cliente.

[2] Um *plugin* é um componente de *software* que adiciona à aplicação em que é embebido, uma ou mais funcionalidades.

[3] RMI – *Remote Method Invocation*, consiste numa API *Java* que implementa um mecanismo de RPC (*Remote Procedure Calls*)

Ao seleccionar da lista dos autores de anotações um deles, a sua anotação é exibida. Adicionalmente, as recomendações também são exibidas como se fossem anotações de um autor. Cada um destes pseudo autores representa um dos algoritmos utilizados para a obtenção de recomendações.

Os utilizadores podem interrogar as recomendações, preenchendo um formulário de critérios de pesquisa. Estes critérios são utilizados para efectuar as pesquisas na base de dados. Os resultados das pesquisas não são exibidos no navegador assistente, mas sim no próprio navegador, sob a forma de hiperligações.

Outra forma de consultar as anotações, consiste num processo muito parecido com os tradicionais favoritos. As anotações ficam disponíveis através de um menu em árvore, com a mesma estrutura hierárquica do *thesaurus* já referido. As folhas desta árvore são as recomendações.

A autenticação é um processo que pode em certos canais ser necessário e em outros não, assim como, o processo de registo e criação de novos utilizadores. Os aspectos ligados à autenticação são dependentes do canal. Um utilizador depois de fazer a autenticação no serviço, está autorizado a um determinado conjunto de operações, tais como: acesso às anotações, publicação de anotações, edição da estrutura de palavras chave, o *thesaurus* e até funções de administração.

Resultados

Pharos proporciona um espaço virtual onde comunidades de diversos interesses podem trocar conhecimentos.

Relativamente aos sistemas de indexação, pelo facto de *Pharos* se basear num ambiente colaborativo, existe a troca de informação de carácter subjectivo, o que não é possível nesses sistemas.

Relativamente aos sistemas de notícias e distribuição de mensagens electrónicas, é capaz de representar a heterogeneidade de um largo número de utilizadores, sintetizando automaticamente e de uma forma personalizada as recomendações de cada um dos utilizadores. Comparativamente a outros sistemas de recomendações (por exemplo: as recomendações de livros na *Amazon*[1]), é bastante mais flexível, dado que uma anotação consiste numa estrutura de dados e não apenas num simples valor ou pequeno texto.

Os autores desta proposta acreditam que sistemas como *Pharos*, serão amplamente utilizados, e acrescentam: "os seres humanos são os agentes inteligentes da *Internet*" [VBOD1999].

[1] http://www.amazon.com

Capítulo 2 - Proposta de um novo serviço

A *Internet* coloca à disposição dos seus utilizadores um conjunto de potencialidades bastante vasto, onde a facilidade de intercâmbio de informação se destaca. Mas, esta facilidade de comunicação ao permitir a partilha global da informação entre todos os investigadores e entre todas as áreas científicas, acaba por levantar alguns desafios.

Actualmente, uma comunidade de uma dada área de investigação pode encontrar a informação de que necessita (artigos científicos, por exemplo), de diversas formas: por via dos meios tradicionais, em revistas e jornais científicos; ou electrónicos, na *Web* ou em CD-ROM. O investigador que pretende, a partir de uma dada publicação, conhecer os comentários que suscitou ou até a sequência dada por terceiros ao trabalho nela contido, vê-se em dificuldades dada a diversidade de formas e lugares possíveis para essas publicações. Em geral, torna-se difícil aos investigadores acompanhar a evolução das suas especialidades.

Como consequência da disseminação da informação e de não existir um único mecanismo de pesquisa suficientemente eficaz, a pesquisa bibliográfica torna-se muito complexa e demorada, sendo muitas vezes necessária a intervenção de um especialista de pesquisa de informação.

Outro aspecto é o ritmo com que surge nova informação. Esse ritmo é bastante superior à capacidade dos investigadores em processá-la. Dada esta impossibilidade, naturalmente o resultado é o desperdício de informação, podendo esta ser bastante importante.

Portanto a questão que se coloca é: como acompanhar, estruturar e filtrar a informação de carácter científico? O serviço que se propõe neste capítulo tem como objectivo a minimização destes desafios. Começa-se por referir as suas características operacionais para em seguida apresentar o modelo que descreve a metodologia e a estrutura a dar à informação, de modo a concretizar essas características.

2.1 Características do serviço

Organização da informação em centros temáticos.

A sua característica distintiva será a de construir centros temáticos com documentação filtrada, interligada e comentada por um especialista e, cumulativamente, outros utilizadores. Pretende-se portanto, oferecer algo mais que a simples publicação dos artigos. Uma consequência benéfica desta organização temática é a filtragem, que facilita a pesquisa bibliográfica.

Estimular os utilizadores a usar o serviço.

O sucesso de um serviço deste género depende totalmente do interesse dos seus utilizadores em o usar. O espírito de colaboração entre utilizadores é outro aspecto muito importante. Desta forma, é imperativo torná-lo suficientemente atractivo para que seja utilizado. E esse interesse, pode ser conseguido com: facilidade de utilização e interfaces agradáveis; eficiência; integração com outros serviços; qualidade dos documentos que constituem o repositório. Havendo interesse por parte dos utilizadores, o serviço tornar-se-á dinâmico e a colaboração entre utilizadores surgirá espontaneamente.

- Partilha da informação

  O serviço a implementar deverá permitir aos seus utilizadores o fácil acesso e partilha da informação disponível num repositório de documentos (artigos com o devido valor científico). Este repositório, poderá ainda estar distribuído por vários sistemas, geograficamente dispersos.

- Fácil acesso ao serviço

  Quanto à infra-estrutura de telecomunicações de suporte, será de toda a conveniência que as aplicações a desenvolver se baseiem num dos serviços de larga aceitação, como é o caso da *Web*. Assim, consegue-se também minimizar todas as questões ligadas à parametrização das aplicações, reduzindo-se ao máximo os recursos (apenas um navegador *Web)*, e facilita-se o acesso ao serviço (basta o acesso à *Internet*).

- Facilidade de utilização

  Outro requisito importante do serviço a desenvolver é a facilidade de utilização. As aplicações deverão ser o mais intuitivas possível. A falha na concretização deste requisito pode comprometer o sucesso de uma implementação do serviço, pois os utilizadores não querem perder muito tempo no processo de aprendizagem.

  O facto das ferramentas a desenvolver se basearem num simples navegador *Web,* conduz a que muito desse desenvolvimento passe pela utilização do HTML. Devido à existência de algumas

limitações (ver em 3.1.3) do HTML, podem surgir dificuldades acrescidas à concretização dessas ferramentas fáceis de usar, se as exigências de interface GUI[1] forem elevadas.

- Integração com outros serviços

  Deverá integrar-se ao máximo com outros sistemas, nomeadamente com sistemas de pesquisa, tanto bibliográfica como em texto integral. Evita-se assim, a duplicação de esforços e obtêm-se uma maior rentabilização de recursos. A integração com outros sistemas, como foros de discussão, têm também bastante interesse.

  Para além das vantagens da reutilização de recursos, uma integração bem conseguida, constitui outra forma de cativar os utilizadores, pois tendo à sua disposição um conjunto amplo de ferramentas, estes certamente se sentirão mais cativados para utilizar o serviço.

Controlo dos acessos

O serviço a implementar deverá comportar funcionalidades complementares relativamente ao objectivo principal, como o controlo de acessos. O serviço proposto é uma ferramenta cuja essência é o trabalho colaborativo. Contudo, deverá também estar prevista a possibilidade de um utilizador qualquer poder optar por restringir o acesso à informação que produziu. Assim, caso os utilizadores optem por esse estilo, podem também desenvolver um trabalho mais individualista.

Registo da actividade dos utilizadores

Outra funcionalidade secundária, é o registo das actividades efectuadas pelos utilizadores. Porém, esta informação tem aplicações cujo interesse é fundamental. A informação relativa à actividade dos utilizadores pode ser utilizada de várias formas:

- Obtenção de perfis de utilizadores

  Poder-se-á definir o perfil do utilizador, isto é, conhecer as suas preferências e interesses. Desta forma, é possível dar ao serviço alguma inteligência e elevar a qualidade geral do serviço prestado, fazendo sugestões ao utilizador e até o aviso (por correio electrónico) da publicação de novos documentos na sua área de interesses. Por outro lado, sendo possível saber quais os documentos mais consultados, é possível definir comunidades de interesses e quantificar a actividade que efectuam. Para os administradores do serviço, esta informação é bastante importante. É com base no conhecimento da existência destas comunidades específicas de interesses, que os administradores do serviço lhes podem dar mais atenção. Essa atenção especial pode ser efectuada por exemplo, com o reforço do repositório com novos documentos.

- Funcionalidades de navegação

  Para o utilizador, o registo do trajecto efectuado até um dado momento, o histórico, permite a criação de mecanismos adicionais para a navegação e pode constituir em si mesmo um elemento informativo valioso para terceiros menos experientes.

[1] *Graphic User Interface*

- Manutenção do serviço

  Para fins operacionais, a informação da actividade dos utilizadores tem interesse numa fase posterior ao arranque, em que graças a um possível sucesso do serviço, com eventuais sobrecargas, se pretenda optimizar a distribuição dos documentos. Conhecendo os documentos que são acedidos, e de que local, o administrador do serviço poderá criar réplicas dos documentos de forma a minimizar as distâncias, e obviamente o tempo de transferência do documento, do sistema servidor ao cliente.

- Segurança

  A segurança é outro aspecto beneficiado com a existência de um registo de actividades. Existindo um processo de autenticação certamente também haverá todo o interesse em controlar a actividade dos utilizadores dentro do serviço.

- Taxação

  A taxação é outra aplicação a dar à informação gerada, resultante dos acessos a documentos e funcionalidades do serviço. Portanto, deverão ser criadas as condições para o caso da tomada da opção de tornar o serviço financeiramente auto-suficiente, ou até, a sua viabilidade comercial.

Estes mecanismos de registo de actividades são de extrema importância, porém, o uso desta informação tem de ser feito de tal forma que os utilizadores não se sintam espiados, e como resultado percam a confiança no serviço e deixem de o usar.

2.2 Modelo

A montagem de um serviço com as características do atrás especificado pressupõe um conjunto de decisões articuladas num modelo da estrutura de informação e da colaboração que se pretende facilitar. O modelo a seguir descrito poderá ser utilizado para a criação do referido serviço, permitindo que todos os objectivos possam ser realizados.

O presente modelo é baseado no modelo *Theseus* [RHRS1993], este com o objectivo de definir uma metodologia que permita estruturar a informação na perspectiva do desenvolvimento de serviços de apoio à aprendizagem. O modelo aqui proposto não se destina ao desenvolvimento deste tipo de serviço, visa antes serviços de apoio a comunidades de investigação.

2.2.1 Objectivos do modelo

Facilitar o acesso e partilha da informação

O modelo define uma metodologia de acesso e partilha de recursos informação, recursos esses que podem ou não estar distribuídos por vários sistemas. Pretende-se que possam ser

construídas ferramentas de trabalho de investigação, baseadas neste modelo. Os utilizadores de tais ferramentas poderão construir a sua própria base de informação, estruturada segundo o modelo apresentado.

Estruturar a informação

O modelo define a estrutura a dar à informação, tendo em vista a revisão bibliográfica colaborativa de documentos disponíveis num repositório, para comunidades de investigadores, numa ou mais áreas. Os serviços que poderão ser criados com base neste modelo deverão permitir aos seus utilizadores a inserção de meta informação, que complemente os conteúdos já existentes num repositório de documentos (artigos científicos). Os documentos têm uma estrutura própria, independente da estrutura global da informação gerida pelo serviço.

Promover o espírito colaborativo entre utilizadores

O modelo pretende também tirar proveito do espírito colaborativo entre os investigadores, utilizadores de serviços baseados no presente modelo. É graças à colaboração entre utilizadores que outros níveis de informação de valor acrescentado podem surgir associados aos objectos que constituem a base do repositório.

Flexibilidade da estrutura

Os serviços baseados neste modelo deverão permitir que os seus utilizadores tirem o máximo de proveito da informação colocada à disposição. O modelo não deverá colocar qualquer restrição quanto ao número de documentos, ou quanto à ligação desses documentos para o exterior. Outro aspecto, é o fácil estabelecimento de relações interdisciplinares, facilitando o trabalho de pesquisa. Assim, possibilita-se aos utilizadores a descoberta de novas áreas, com associações bastante ricas à área específica da cada um.

Eficiência

Num conjunto bastante amplo de documentos, colocado à disposição dos seus utilizadores, o modelo tem de facilitar a pesquisa e, depois de obtidos os resultados, deverá ser rápido o processo final de filtragem, sem que o utilizador seja obrigado a fazer um exame exaustivo de todos os apontadores.

Independência quanto à plataforma

Tal modelo deverá também garantir que os serviços nele baseados, possam com a maior facilidade possível operar com vários sistemas independentemente da plataforma. Assim, facilita-se integração com outros sistemas.

2.2.2 Planos de assuntos

É central ao modelo proposto o conceito de plano de assuntos. Este é uma extensão ao conceito de caminho de assunto [RHRS1993] proposto no modelo *Thesus*. O caminho de assunto é uma visão geométrica de um conjunto de segmentos, unindo os nós do hipermeio [RHRS1993], formando um "fio" que conduz o utilizador por um dado percurso. O plano de assunto também consiste num conjunto de nós interligados, contudo existem dois aspectos que o distingue de um caminho de assuntos.

Enquanto que o caminho de assuntos é um "fio" que estabelece um caminho por onde o utilizador deve seguir, o plano de assunto pressupõe que possam existir bifurcações, dando origem a dois ou mais caminhos alternativos, os quais podem mais à frente fundir-se e continuar num único caminho.

Outro aspecto que distingue um caminho de um plano de assunto, é o objectivo para o qual cada um deles foi definido. O caminho de assunto tem como objectivo proporcionar um processo de aprendizagem sobre um dado assunto, dar orientação. Um plano, permite a representação das ligações entre diversos documentos relacionados com um dado assunto. Desta forma, comunidades de investigadores nesse assunto, podem ter facilitado o acesso e a partilha do conhecimento nessa área. Por outro lado, o plano de assunto, poderá constituir um meio de representação do conhecimento nessa área.

A figura que se segue ilustra a distinção entre os dois conceitos.

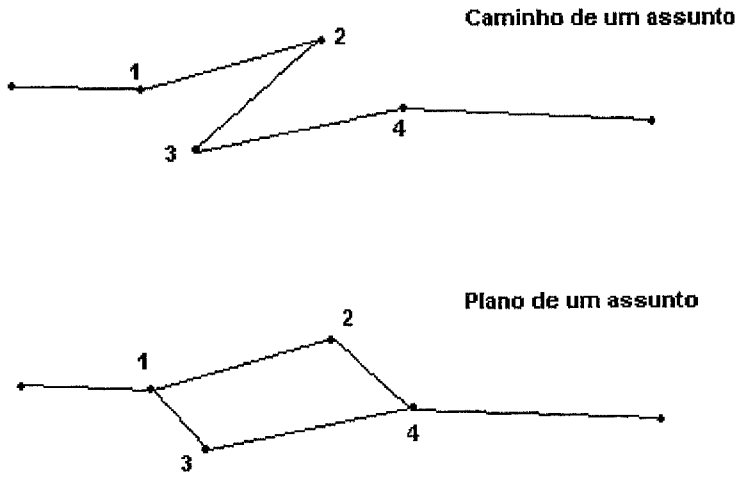


Figura 13 - Caminho e plano de assunto

A linha que une os pontos 1,2,3 e 4, permite que o utilizador (como aluno) siga um caminho único pelos quatro nós. Um exemplo de quatro documentos hiperligados desta forma é o seguinte: o documento representado pelo ponto 1, é um ficheiro de texto que descreve um processo químico de electrólise. O ponto 2 é um ficheiro de vídeo, demonstrando como proceder à montagem de todos os elementos necessários à electrólise: encher a tina com água, adicionar sal, colocar os eléctrodos e alimentá-los com uma tensão eléctrica contínua, o vídeo

termina com a observação das bolhas de oxigénio e hidrogénio. O documento 3, é um ficheiro de animação, que explica em detalhe todas as reacções químicas, ao nível molecular, que se passam durante o fenómeno observado. No final, para que o utilizador consolidasse a sua aprendizagem, um último ficheiro de vídeo exibe algumas demonstrações da aplicação da electrólise na indústria. Neste exemplo, é perfeitamente óbvio que a navegação pelos quatro documentos sem esta ordem não permite ao utilizador a mesma facilidade de compreensão do fenómeno.

No caso do plano de um assunto, o utilizador quando chega ao nó representado pelo ponto 1, confronta-se com dois caminhos alternativos, cada um deles representando o ponto de vista, opinião ou comentário de um ou mais autores (apenas dois no exemplo). Um exemplo para um plano de assunto como o apresentado na `Figura 13` é o seguinte: o documento representado pelo ponto 1 refere a extinção dos dinossauros e aponta duas teorias possíveis, uma abordada no documento 2 e a segunda no documento 3. Como a extinção dos dinossauros é um facto, os dois documentos são hiperligados ao quarto, que descreve toda a fauna após a extinção. Mas se o percurso for iniciado no documento 4, a relação que este tem com os documentos 2 e 3, é a de justificar quais as razões do tipo de fauna naquela época, e porque razões teriam essas espécies evoluído dessa forma. Nesse caso, a explicação estava nas duas teorias que tinham justificado o desaparecimento dos dinossauros. Portanto, o plano de assunto prevê ainda a possibilidade de ligações bidireccionais.

Enquanto que o caminho de assuntos fornece ao utilizador (como aluno) uma linha ("o fio" aplicado no modelo *Theseus*, um modelo para o ensino) que o ajuda a não se perder no hipermeio, o modelo aqui proposto tem como objectivo a criação de serviços de apoio a investigadores, com requisitos de outro tipo, como por exemplo, a liberdade de escolha do trajecto. Um aluno necessita que lhe seja indicado um caminho, para que não se perca na imensidão de trajectos que ligam todos os nós do hipermeio. Alguém que está a fazer um trabalho de pesquisa não deve ter qualquer restrição, deve sim, ter à sua disposição o maior número de pontos de vista de outros investigadores.

Sendo grande a possibilidade de escolha em cada nó, o utilizador (como investigador) deverá dispor de mecanismos que o ajudem a encontrar rapidamente o que pretende. Assim, é necessária a existência de meta dados associados aos trajectos. No exemplo, os dois papéis que a ligação bidireccional permitia simultaneamente, entre o documento 3 e o 4 (assim como, entre o 2 e 4) poderia ser clarificada com esses meta dados.

Sendo a metodologia utilizada um processo colaborativo produz resultados cuja qualidade depende dos colaboradores. Desta forma, o modelo, ao permitir a filtragem por planos de assunto ou autor, facilita ao utilizador o processo de filtragem da informação reduzindo este inconveniente.

Nos serviços criados com base no presente modelo, naturalmente poderão existir vários planos de assunto. Um documento pode pertencer a vários planos. Ao utilizador deverá ser permitido saltar de um plano para outro, enquanto tem acesso a um documento que pertence a mais do que um plano de assunto. Então, deverá existir um identificador do plano de assunto. Este identificador, assim como um conjunto de informações adicionais, como o seu autor, data de criação, título, um documento inicial (ponto de acesso ao plano), entre outras, permitem ao utilizador uma melhor identificação desses planos. Poderá ainda existir um menu com todos os planos de assunto disponíveis no serviço.

2.2.3 Documentos

Tal como foi anteriormente referido um plano de assunto consiste numa malha de hiperligações relacionando entre si os documentos, os nós do hipermeio. Um único documento, um artigo científico por exemplo, pode abordar diversos assuntos, e sendo assim existe toda a conveniência em associar a parte que é relevante a um dado assunto ao respectivo plano, em vez do documento todo. Consequentemente, também são possíveis hiperligações dentro do próprio documento, interligando duas partes distintas desse mesmo documento.

A especificação dos pontos de contacto entre documentos (capítulo, secção, página, parágrafo, etc.) deverá ser a mais flexível possível, ficando esta apenas dependente das restrições tecnológicas colocadas pelos formatos de representação dos documentos.

A hiperligação da parte do documento que é relevante ao assunto abordado facilita extraordinariamente a compreensão deste. Com vantagem de permitir ao utilizador a identificação quase imediata das duas ideias que o autor da hiperligação pretende relacionar.

2.2.4 Descrição do modelo

Um plano de assunto cobre um assunto. Sendo assim, será de prever a existência de um plano por cada assunto. Paralelamente ao plano, às hiperligações e aos documentos, poderá ainda existir outro tipo de informação, isto é, anotações associadas ao documento.

Os diversos planos que podem existir, assim como toda a estrutura de informação adicional, podem ser representados por um modelo baseado em três níveis de informação. O diagrama que se segue pretende ilustrar este modelo.

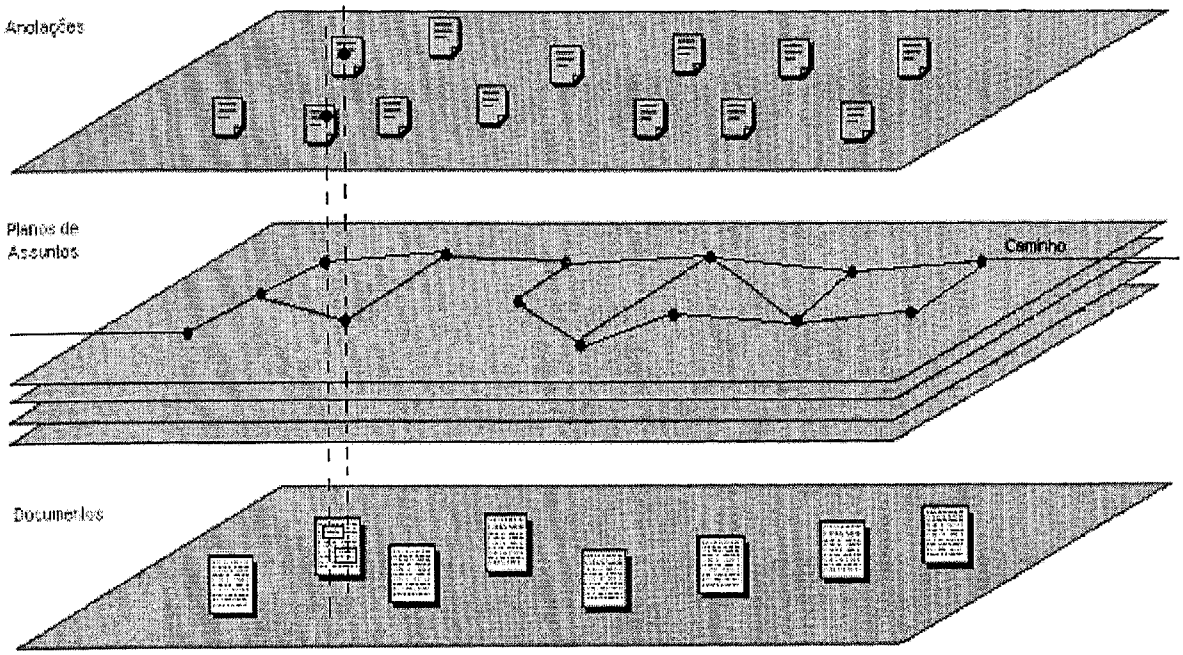


Figura 14 - Modelo de representação de conhecimento de 3 níveis

O primeiro nível, é constituído pelos documentos que constituem o repositório. A estes documentos, é associada meta informação, o que permite a pesquisa dos documentos segundo os métodos clássicos para este tipo de catalogação.

A estrutura deste nível é, e pretende-se que seja precisamente a mesma de um simples repositório de documentos, em formato digital, com informação de catalogação associada. Desta forma, às implementações deste modelo poderão ser acopladas quaisquer bibliotecas digitais. Uma estrutura deste tipo, permite ainda que os diversos documentos estejam dispersos por mais do que um lugar, e seja da responsabilidade do sistema de catalogação fornecer a sua localização.

O segundo nível, são hiperligações que permitem dar a um dado conjunto de documentos, uma ordem e um relacionamento coerente, através de planos de assunto. Esta malha de hiperligações, um plano de assuntos, consiste num conjunto de percursos bidireccionais que o leitor poderá seguir. Cada um destes percursos, segundo a pessoa que os definir, deverá ser o melhor, para que um ponto de vista, ideia ou tese, do seu autor, possa ser abordado ou defendido. Pretende-se que este percurso ligue o que mais relevante (segundo o ponto de vista do seu autor) há na respectiva área. Naturalmente, ligações com outras áreas são possíveis e necessárias para que haja também ligações interdisciplinares.

O utilizador terá à sua disposição um conjunto de propostas de quais os melhores "caminhos" a tomar. Assim, ele tem menos probabilidades de se dispersar por outros caminhos menos frutuosos. Cada nó, um documento, pode ou não estar referenciado noutros planos. Se assim acontecer, o utilizador poderá conhecer as ligações a outros planos de assuntos, assim como as ligações por autor, ou outros a quaisquer critérios.

Como forma de elevar o valor destes trajectos, em cada um deles deverá existir um conjunto de meta dados que define a relação entre os dois documentos, e que conceitos estão abordados

no documento. Esta informação deverá ser constituída por um conjunto de propriedades que especificam a relação e os documentos. Na prática, esta informação serão anotações efectuadas em linguagem controlada, associadas à ligação e ao documento. Os termos desta linguagem controlada terão de ser definidos por um utilizador com permissões de administração. Este mecanismo é de total interesse, pois é graças a ele que se torna possível uma considerável economia de tempo. O utilizador sempre que se depara com uma ligação a um documento pode antes de o abrir, já ter uma noção do que vai encontrar no outro extremo da hiperligação. Por outro lado, essa informação que descreve a ligação e os documentos, pode ser utilizada por mecanismos de pesquisa, e desta forma o processo pode ser automatizado.

O seguinte diagrama complementa a `Figura 14`, e pretende ilustrar como, segundo o modelo, deverá ser feita a classificação do relacionamento entre os documentos. São os mesmos meta dados que permitiriam a clarificação dos papéis tomados por cada um dos documentos, na hiperligação bidireccional, do exemplo dado em [2.2.2].

As anotações em cada um dos extremos da ligação, pretendem classificar os conceitos ligados aos objectos em cada extremo da ligação, enquanto a anotação relativa ao tipo de relação classifica o papel de cada um dos documentos relativamente ao outro. Consiste numa estrutura de informação em tudo idêntica a uma ontologia com a utilizada no projecto *WebOnto* [SSEMJD1999a] abordado no capítulo anterior..

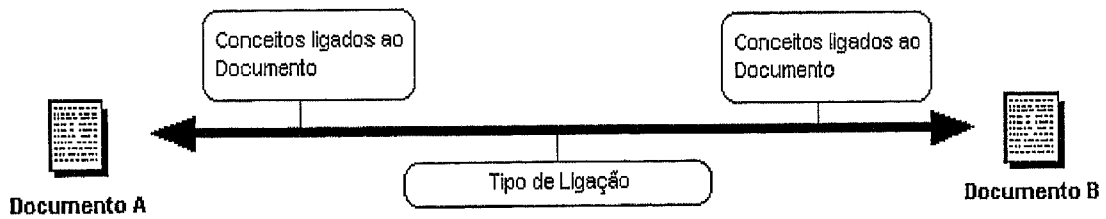


Figura 15 – Trajectos anotados com linguagem controlada

O terceiro nível, é constituído por um outro tipo de informação mais subjectiva. São anotações de âmbito pessoal, que a pessoa que estruturou o plano do assunto achou conveniente acrescentar. Estas anotações poderão expressar opiniões do âmbito pessoal, onde o autor da anotação pode colocar os seus pontos de vista sobre as matérias que estão a ser abordadas.

Cada um destes níveis dá ao nível imediatamente inferior um valor acrescentado, valorizando a informação.

2.2.5 Módulo temático

Definindo a distância entre dois documentos como o número de hiperligações para chegar de um documento a outro, constata-se que os documentos sobre um mesmo assunto têm uma tendência a manterem-se próximos [LAA1999]. Limitando a distância entre documentos e simultaneamente considerando aqueles com um maior número de ligações entre si, obtém-se uma "nuvem" de documentos que cobre um assunto ou até uma área científica, dependendo naturalmente do "raio" da "nuvem". Desta forma, podem ser obtidas outras associações entre os documentos, estas num sentido mais amplo do que o plano de assunto.

O módulo é então, um conjunto de documentos fortemente ligados e correlacionados. Naturalmente, não é obrigatório que todos os documentos sejam da mesma área científica, ou plano, prevê-se apenas que, a ligação seja mais forte entre documentos da mesma área. Como consequência, o módulo acaba por se tornar em algo difícil de delimitar, onde trabalhos de diversas áreas podem estar relacionados, pois todas as áreas da Ciência também se interceptam, apesar de constituírem unidades de conhecimento autónomas.

Um módulo pode englobar a totalidade ou parte de vários planos de assunto, pois, o que permite a definição de um módulo é a distância entre documentos e o número de ligações entre eles.

Considera-se como parte integrante do módulo não só a informação base, os documentos do arquivo e respectivos meta dados, mas também informação adicional, composta pelas ligações e anotações.

2.2.6 Comparação de resultados

O conceito de hipermeio, definido por *Harden e Stringer* no modelo *Theseus* [RHRS1993] pressupunha a existência de um conjunto de objectos hiperligados por "fios", cada um deles um caminho de assunto.

O presente modelo, como já foi referido, é baseado no modelo *Theseus*, tendo naturalmente as suas diferenças. A topologia das hiperligações é a diferença mais relevante. Enquanto que, no modelo *Theseus* a estrutura das hiperligações que representa os caminhos de assunto é um "fio", o presente modelo sugere uma estrutura em malha para representar a estrutura dos planos de assuntos. Esta é a característica mais distintiva entre os dois modelos, a extensão ao conceito de caminho de assunto, que resultou no novo conceito, o plano assunto. Desta forma, as duas estruturas do hipermeio dos modelos não terão diferenças significativas entre si. Outra característica, é a vocação do próprio modelo. Este novo conceito o plano de assunto, é mais flexível, estando melhor adaptado às necessidades dos investigadores, os destinatários dos serviços baseados no modelo. Um plano de assunto é capaz de exprimir as várias perspectivas de um assunto, o que não é tão facilmente atingido com um caminho de assunto, que tem como objectivo principal a orientação e o encaminhamento do utilizador enquanto aluno.

O modelo proposto acrescenta também as anotações livres. Acrescenta-as como um espaço de opinião pessoal, onde o autor do plano de assunto pode expressar as suas críticas, sugestões ou simplesmente tomar os seus apontamentos. Mas, como estas anotações não trazem um contributo relevante à pesquisa, por se tratarem de conteúdos não estruturados, é proposta adicionalmente a existência de outro tipo de anotações, mais objectivas, com termos em linguagem controlada. Estas anotações objectivas estando associadas às ligações e aos documentos, permitem a classificação relacional dos objectos documentais, com grandes vantagens para a pesquisa. As mesmas vantagens que uma ontologia académica como aquela abordada em [1.2.5].

O hipermeio com a sua riqueza de ligações entre documentos, pode funcionar como uma fonte de informação de pesquisa bibliográfica. Havendo uma separação em três camadas, dos diversos tipos de categorias de meta dados associados ao fundo documental, torna-se possível a implementação de poderosas ferramentas de pesquisa. Na primeira camada, os documentos do repositório, é possível a pesquisa em texto integral, por palavra-chave ou pesquisa em catálogo. A segunda camada do hipermeio, é uma base de dados de hiperligações e classificações relacionais, como tal, permite obter os mesmos benefícios possíveis com as bases de dados de citações (ver em 1.2.4) e um sistema de indexação colaborativa (ver em 1.3.3).

2.3 A necessidade de autenticação

O acesso ao serviço deverá requer uma validação de um utilizador através de um processo de autenticação. Cada utilizador terá de ter um *username* e uma *password* que lhe garante o acesso ao seu ambiente de trabalho.

A necessidade dos acessos serem assegurados por um mecanismo de autenticação, prende-se ao facto de permitir que, outras funcionalidades de valor acrescentado possam ser adicionadas ao sistema. O fornecimento de informação sobre os acessos ao fundo bibliográfico pode ser utilizado para vários fins como os referidos em [2.1]. Mas por outro lado, torna-se possível que todos os objectos (anotações e hiperligações) de cada utilizador estejam sob o seu controlo, e não se tornem de domínio público, caso o seu autor assim o deseje. Cada um desses objectos tem um conjunto de propriedades associadas, com vista à definição de permissões, que lhes condicionem o acesso. Isto é, o acesso a um dado objecto poderá ser exclusivo ao próprio utilizador (o autor), ou então, de um grupo de utilizadores, ou ainda, se é um objecto de âmbito público, a que todos os utilizadores têm acesso.

Este mecanismo de autenticação pretende proteger não os documentos, mas toda a informação de valor acrescentado, que lhes está associada. Esta informação acrescentada aos documentos, constituída pelas hiperligações, anotações, etc., terá de estar sob o controlo do utilizador que a criou. Só o utilizador, proprietário dessa informação, poderá fazer essa gestão. Pretende-se com isto, que os utilizadores possam usar o sistema como uma ferramenta de trabalho individual, e se assim o quiserem, disponibilizarem o seu trabalho a um dado conjunto de utilizadores.

2.4 Aplicação do modelo no serviço

O modelo proposto descreve a forma como deverão ser estruturadas as diversas categorias de informação. Porém, a criação de um serviço com as características referidas, com base no modelo descrito, levanta algumas questões.

Localização dos documentos

Como foi referido na descrição do modelo, existem três níveis distintos, com tipos de informação bem definidos. O nível inferior, os documentos mais os respectivos meta dados de catalogação, podem constituir uma unidade autónoma do serviço. Para que tal seja possível, deverá ser especificada uma interface e protocolo de comunicação entre este sistema que implementa este primeiro nível do modelo, e os restantes sistemas, que implementam os outros dois níveis superiores. Desta forma, a implementação do nível inferior (os documentos) poderá estar distribuída por vários sistemas. Um exemplo de um possível protocolo, poderá ser o seguinte:

O sistema de catalogação fornece uma URL, que localize o documento, quando lhe é fornecido um identificador do documento em causa. Uma função do género:

URL=*função*(identificador).

Na prática, o sistema de catalogação seria um mero tradutor identificador/URL. Assim, de uma forma totalmente objectiva e eficiente, o documento pode ser encontrado, onde quer que esteja. Desta forma, o acesso ao documento é totalmente transparente, e fica o serviço de catalogação responsável por fornecer a localização (URL) do documento. Por outro lado, o facto das ligações estarem definidas através de identificadores em vez da própria URL, permite que o documento possa mudar de localização e não obrigar a uma actualização da URL. O próprio sistema de catalogação também deverá poder estar distribuído.

Esta solução não seria a única, mecanismos como o *Document Object Identifier* - DOI[1], ou o *Persistent Uniform Resource Locators* – PURL[2], poderão fornecer soluções alternativas para a resolução da localização dos documentos.

Qualquer uma das soluções tem condições de garantir a independência do sistema de catalogação e até fornecer as condições necessárias para que aspectos de optimização, como a redução dos tempos de transferência de documentos, possam ser aplicados.

Autenticação

A autenticação é outro processo beneficiado com a integração deste serviço num sistema central, onde uma base de dados com os utilizadores já esteja definida. Se a implementação de um serviço baseado no modelo proposto, utilizar uma base de dados de utilizadores, partilhada por outros sistemas, o serviço ganha com isso, pois não obriga à criação de uma nova base de dados de utilizadores, com o inconveniente para os utilizadores de estes terem

[1] http://www.doi.org

[2] http://www.purl.org

de subscrever um novo serviço. Adicionalmente, seriam obrigados a gerir mais uma senha de acesso. Naturalmente, este benefício só tem sentido numa organização em que existem outros serviços com autenticação.

Falta de conteúdos na fase inicial

O problema de qualquer serviço, cujo sucesso depende do espírito colaborativo, é a falta de conteúdos na fase de arranque do serviço. Para os utilizadores se interessarem pelo serviço, este tem de oferecer conteúdos. Mas, se esses mesmos utilizadores são os responsáveis pela existência dos conteúdos, logo surge o problema da falta de conteúdos na fase inicial. Desta forma, haverá toda a conveniência que logo de início existam alguns planos de assunto, preferencialmente criados de forma automática. Um desses planos poderia ser construído automaticamente, com base nas citações disponíveis nos documentos do repositório. Outros planos de interesse obtidos automaticamente a partir da informação de registo da actividade dos utilizadores, é o percurso efectuado por determinados utilizadores que pode ter bastante interesse para utilizadores menos experientes.

Capítulo 3 - Tecnologias relevantes

A apresentação das tecnologias relevantes para o desenvolvimento de serviços de revisão colaborativa está organizada em duas partes. Começa-se pelas tecnologias utilizadas para a representação e apresentação de documentos, referindo-se a seguir as tecnologias relacionadas com a publicação na Web.

3.1 Representação e apresentação de documentos

Um documento consiste num conjunto de dados ao qual é dada uma estrutura, lógica e regras. Estes três aspectos, que modelam os dados e caracterizam um documento, são definidos através de formatos e linguagens. A independência quanto à plataforma, a interoperabilidade, a universalidade e a eficiência são alguns dos aspectos que ditam a aceitação e o sucesso do formato ou linguagem. Nesta secção apresentam-se linguagens de representação de documentos baseadas em marcadores e linguagens de descrição de página. Estes dois tipos representação de dados e informação têm características e objectivos distintos. Na parte final desta secção é feita uma análise comparativa dos diversos formatos de representação de documentos, anteriormente descritos.

3.1.1 Linguagens baseadas em Marcas

3.1.1.1 SGML - *Standard Generalized Markup Language*

A *Standard Generalized Markup Language* – SGML [SGML1986], foi adoptada como ISO 8879, em 1986. Esta fornece um modo consistente e preciso, aplicando esquemas de marcas, de descrição das partes que compõem um documento. Dada a sua extrema flexibilidade constitui uma linguagem de definição de linguagens de marcas.

A norma SGML resultou do trabalho de pesquisa em codificação genérica e linguagens baseadas em esquemas de marcas, no início da década de 1970.

Os vários trabalhos de pesquisa convergiram no *ISO subcommittee Text Description and Processing Languages*, o qual deu origem à norma em 1986. A norma SGML é uma especificação totalmente aberta e genérica, que reflecte a diversidade das necessidades da indústria. Esta especificação não define directamente qualquer género de tipo de dados, nem restringe o tipo de dados contidos num documento.

O SGML tem a flexibilidade suficiente para descrever qualquer conjunto de dados logicamente estruturados, quer seja um formulário, uma carta, livro, relatório, enciclopédia, dicionário ou até, uma folha de cálculo ou base de dados.

Este standard também não define nenhum sistema nem esquema de marcas, para qualquer tipo de documento em particular.

A partir do SGML, muitos esquemas baseados em marcas podem ser desenvolvidos, um por cada tipo de documento ou classe de objectos. Isto é uma vantagem, mas também pode constituir uma desvantagem.

A sua flexibilidade para especificar estruturas de documentos para qualquer tipo de aplicação, foi a razão pela qual foi amplamente adoptado em todas as áreas como um método de codificação e intercâmbio de documentos. Adapta-se a qualquer plataforma, em qualquer nível de complexidade. O SGML cria condições para que seja possível separar a informação que descreva a apresentação do conteúdo e da estrutura.

Mas o SGML vai muito mais longe, para além do objectivo de permitir o intercâmbio de informação entre múltiplas plataformas; esta norma permite dar um valor acrescentado à informação e fomenta a reutilização da informação.

De facto, a grande vantagem da utilização do SGML é a flexibilidade que oferece, ao tornar possível a reutilização para diferentes propósitos da informação. Esta forma de estruturar a informação permite interagir com sistemas em qualquer plataforma, mesmo que tenham sido desenvolvidos antes do aparecimento da norma.

Porém, dado que o SGML é extremamente aberto, este facto levou a que muitas organizações desenvolvessem os seus próprios meios de o utilizar, os quais têm sido usados como *standards ad-hoc*. Naturalmente, esta situação leva a alguma confusão e duplicação de esforços. Actualmente esta situação já atingiu um estado de alguma estabilidade e consenso entre os grupos de trabalho.

Objectivos e Aplicações

A norma SGML disponibiliza uma forma coerente e não ambígua de descrever seja o que for, dentro de um documento.

É uma meta linguagem desenvolvida a partir dos seguintes requisitos [SGML1986]:

- Que uma larga variedade de sistemas de processamento de texto fossem capazes de aceitar documentos SGML;

- Possibilidade de escolha do conjunto de caracteres e da língua nacional;
- Uma ilimitada possibilidade de escolha da organização dos ficheiros, ou do fluxo de bytes;
- Coexistência de dados descritos por marcas com outro tipo de dados;
- Que seja perceptível às pessoas, bem como às máquinas;

As aplicações usam o SGML para definir esquemas de marcas. Por exemplo, um grupo de editores poderia definir um desses esquemas de marcas, para descrever os livros de texto. Cada editor poderia posteriormente implementá-lo de forma a adaptar-se ao seu próprio sistema. Se os documentos estiverem em conformidade com o esquema, eles poderão ser transferidos. Qualquer autor ao produzir um trabalho que esteja em conformidade com esse esquema de marcas, poderá submeter os seus artigos a qualquer editor. Os artigos poderão ser automaticamente formatados, usando o estilo do próprio editor.

Em ambientes SGML, a formatação dos documentos é efectuada no momento em que é feita a apresentação, em vez do momento da entrada dos dados que constituem o conteúdo. O *Document Style Semantics and Specification Language* - DSSSL pode ser usado para controlar a forma como é feita a apresentação da informação codificada em SGML.

O SGML está bem adaptado não só para descrever documentos multimédia, como também revistas, jornais ou livros. A sua facilidade de aplicação tem levado muitos fabricantes de conteúdos em CD-ROM, documentos electrónicos, hipertextos e hipermédia a usar o SGML na sua preparação.

Estrutura de um documento SGML

Um documento SGML contém dois tipos de informação: estrutura e conteúdo (mais a informação ligada à apresentação que poderá estar ou não fora do documento). A estrutura é determinada pelas marcas e o conteúdo fica delimitado entre elas. As marcas no documento definem a estrutura lógica dos diversos elementos que o constituem (por exemplo: capítulos, secções, parágrafos, etc.) e dá toda a informação necessária à manipulação do documento. Ao usar-se SGML, os documentos são estruturados hierarquicamente, como uma árvore. Na raiz dessa árvore está, por exemplo, o Livro, que é dividido em Capítulos, subdividido em Secções, e assim sucessivamente.

Cada documento SGML consiste num prólogo SGML e numa instância do documento. O prólogo contém uma declaração SGML e uma definição do tipo do documento (DTD).

A declaração SGML, normalmente não é visualizada pelo utilizador, serve apenas para indicar ao processador SGML informações diversas, como a versão do SGML a ser utilizado. O SGML possui uma declaração por omissão, para o caso de esta não lhe ser fornecida.

O DTD também não é visualizado, serve apenas para indicar ao processador como deve entender e processar os elementos do documento.

A instância é o documento em si, com a sua estrutura e conteúdo, organizada com base nos elementos descritos no DTD. O elemento contém uma marca de início, o seu conteúdo e uma

marca de terminação. Dentro das marcas, de início e de terminação, está o identificador genérico com o nome do elemento.

O exemplo que se segue, ilustra um possível elemento de um documento SGML. O elemento `obra` representa um tipo de documentos de obras literárias. Este elemento contém ainda outros três elementos interiores, o `titulo`, `autor` e `texto`, contendo cada um deles respectivamente a informação relativamente ao titulo, autor e o próprio texto, da obra literária. O elemento tem um atributo que define o género de obra literária. Os atributos dos elementos são compostos por um par [nome, valor], os quais ajudam a descrever o elemento.

```
<obra genero="poesia">
     <titulo> Titulo da obra </titulo>
     <autor> Autor da obra </obra>
     <texto> ... texto do documento... </texto>
</obra>
```

***Document Type Definitions* - DTD**

Sendo a SGML uma linguagem de definição de linguagens de marcas, é utilizada para criar DTDs. Cada DTD é uma definição da estrutura de uma classe de documentos. Os DTDs, escritos com a linguagem SGML, caracterizam a lógica da estrutura dos documentos e os seus elementos e atributos.

O DTD contém a lista de elementos, atributos, entidades e outras declarações, pelas quais as marca que estruturam o texto são distinguidas. Naturalmente qualquer marca incluída na instância do documento que não esteja definida no DTD, não será reconhecida como tal.

Porque a SGML não define nenhuma marca, a especificação das marcas fica ao cuidado dos utilizadores, ou do grupo de entidades com interesses comuns. Assim sendo, cada um desses grupos de utilizadores, poderá ter terminologias diferentes, isto é, a marca que contém um capítulo, poderá ser definida de várias formas possíveis: <CAP>; <CHAPTER>; <CAPITULO>; etc.. Naturalmente ter-se-á de fazer uma especificação, recorrendo-se a um DTD, que evita este tipo de ambiguidades, na forma como os elementos são definidos.

Entre o DTD e a instância do documento existe uma diferença muito importante, a instância do documento não contém nenhuma definição de marcas, nem o DTD contém informação.

Um exemplo de um DTD será apresentado mais à frente, quando uma das aplicações do SGML, o XML for abordado (ver na página 66).

Processadores (*Parsers*)

O propósito de um processador ou analisador SGML é bastante simples, o de reconhecer as marcas nos documentos SGML. São programas que analisam a sintaxe do documento, que interpretam as marcas e posteriormente entregam a informação às aplicações.

Se por um lado, um documento SGML deverá respeitar as regras de sintaxe, por outro lado, um documento válido e bem formado também tem de obedecer a uma estrutura de descrição, o DTD.

Um processador SGML não tem obrigatoriamente a função de detectar erros. Este deverá apenas processar o documento livre de qualquer erro de sintaxe. Porém, o processador com essa capacidade (analisador validante), deverá alertar sobre eventuais falhas e alertar sobre a existência de condições com potencial para causar erros. Estes podem ser de sintaxe, ou ambiguidade na estrutura ou no conteúdo.

Este conceito de processador, naturalmente, aplica-se aos documentos SGML e seus derivados tais como o XML [TBJPCS1998] e o HTML [DRAHIJ1999].

3.1.1.2 HTML – *Hypertext Markup Language*

A *Hypertext Markup Language* - HTML [DRAHIJ1999], é uma linguagem baseada num esquema de marcas, especialmente adaptada à publicação de hipertexto na *World Wide Web*. É um formato aberto que derivou de outra linguagem, também baseada em esquemas de marcas, tornando-se numa subclasse dessa linguagem, o SGML. O propósito do SGML é o de tornar possível a partilha de documentos criados em qualquer sistema, tendo em conta que esses sistemas poderiam ter características, tanto de *hardware* como *software*, muito diferentes. O HTML herdou do seu predecessor essa importante característica.

Dada a sua simplicidade, um documento HTML, pode ser facilmente criado a partir de um simples editor de texto. Por outro lado, este tipo de documentos é totalmente independente da plataforma na qual são criados e visualizados. Dado que os documentos HTML são também documentos SGML com uma semântica apropriada à representação da informação dos mais variados tipos, nas mais recentes versões do HTML foram definidos mecanismos que permitem representar praticamente todo o tipo de informação. Porém, o HTML tem grandes limitações no modo como descreve a apresentação dos documentos, isto é, na aparência e no posicionamento dos elementos. Para isso, agrega-os em unidades semânticas tais como: cabeçalhos, parágrafos, listas e tabelas. Aqueles que diariamente usam o HTML, frequentemente são obrigados a usar os mais diversos e complexos truques para colmatar esta falha. Sendo este facto evidente, têm sido feitos esforços para reduzir a inadequada capacidade de controlo da aparência dos documentos. Uma prova disso são as folhas de estilo CSS [HLBB1996] e as camadas [DRAHIJ1997].

O HTML tem sido utilizado pelo serviço *World Wide Web* da *Internet* desde 1990, os passos mais recentes na evolução desta linguagem em relação às suas versões anteriores são:

- HTML foi originalmente desenvolvido por *Tim Berners-Lee,* um engenheiro de *software* no Laboratório Europeu de Partículas Físicas, o CERN, 1989. Foi popularizado pelo browser *Mosaic* desenvolvido na NCSA.
- Em Novembro de 1995 surge a versão 2.0, descrita na RFC1866 [TBDC1995], desenvolvida pelo *Internet Engineering Task Force - IETF HTML Working Group*, sendo

concluída em 1996, a qual definiu todo o conjunto de funcionalidades nucleares do HTML.

- A versão 3.0 [DR1995] em 1995, nunca obteve o consenso, e com o esforço do W3C, surge a versão 3.2.
- A versão 3.2 [DR1997] resultou de uma recomendação do W3C datada de 27 de Janeiro de 1997. Esta recomendação veio acrescentar às funcionalidades básicas da anterior versão as tabelas, *applets*[1], texto capaz de envolver imagens, mantendo total compatibilidade com a anterior versão 2.0.
- O HTML 4.0 teve a sua primeira recomendação em 18 de Dezembro de 1997 [DRAHIJ1997]. Quatro meses depois surge uma segunda versão, em 24 de Abril de 1998 [DRAHIJ1998], com alterações pouco relevantes relativamente à primeira.
- Em 24 de Dezembro de 1999 surge uma nova recomendação, o HTML 4.01 [DRAHIJ1999], que vem corrigir alguns erros[2] da anterior especificação 4.0.

A evolução do HTML tem sido no sentido de em cada nova versão serem implementadas novas marcas. Os aspectos ligados à extensibilidade e a separação da informação relativa à apresentação daquela que é conteúdo, têm sido alguns dos pontos chave nos trabalhos levados a cabo em áreas com o XML e *eXtensible Hyper Text Markup Language* – XHTML [W3C2000].

Objectivos e Aplicações

O HTML dá aos autores de conteúdos meios de:

- Publicar *on-line* documentos com cabeçalhos, texto, tabelas, listas, fotografias, etc..
- Ligar informação via links de hipertexto, com um simples "click" num botão.
- Criar formulários, capazes de conduzir transacções em serviços remotos, para os mais diversos fins: pesquisas, comércio, etc..
- A capacidade de embeber, no documento HTML, os mais diversos tipos de documentos: Folhas de cálculo, vídeo, áudio, etc..

Os documentos HTML deverão ser perfeitamente visualizados em qualquer navegador e em qualquer plataforma, pois os produtores de conteúdos não deverão necessitar de desenvolver mais do que uma única versão de cada documento. Cada uma das versões do HTML, atrás mencionadas, surgiu com o objectivo de obter o consenso dos vários fabricantes e

[1] Aplicação *Java* a correr do lado cliente

[2] A lista dessas alterações estão disponíveis em http://www.w3.org/TR/html4/appendix/changes

salvaguardar os investimentos já efectuados por parte dos produtores de conteúdos. Ao tentar manter sempre uma compatibilidade com as anteriores versões, assegurava-se a validade dos documentos em termos de compatibilidade com os navegadores *Web*. Se não for feito um esforço deste tipo corre-se o risco de caminhar para um conjunto diverso de soluções particulares, de formatos incompatíveis, conduzindo à redução do potencial comercial da *Web* para todos os participantes.

Este princípio de universalidade não se resume à compatibilidade inter-versões. O HTML tem sido desenvolvido com base no princípio de que todos os tipos de dispositivos terminais deverão ser capazes de usar a informação disponível na *World Wide Web*: PC's com ecrãs das mais variadas resoluções, telefones celulares[1], em sistemas com acessos de grande ou pequena largura de banda, para dispositivos de entrada ou saída, etc., ou seja, uma variedade extremamente vasta de equipamentos terminais.

Estrutura de um documento HTML

Um documento HTML inicia-se sempre com uma declaração <!DOCTYPE>, seguida pelo elemento HTML, o qual contém ainda dois outros elementos: HEAD e BODY.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
<TITLE>A minha página pessoal</TITLE>
... outros elementos do cabeçalho
</HEAD>
<BODY>
... corpo do documento ...
</BODY>
</HTML>
```

O elemento HEAD, pode conter o elemento TITLE, que define o título do documento, elementos META, os quais permitem a inserção de meta dados como autor, aplicação geradora, algumas palavras-chave, etc.. No elemento HEAD, podem ainda ser colocados *scripts* (executados logo na abertura do documento) ou definição de funções e variáveis, as quais podem ser usadas no resto do documento HTML.

Quanto ao elemento BODY, a estrutura e os elementos que a compõem, é muito diversa. Nele podem coexistir elementos de imagens, tabelas, segmentos de texto com propriedades especificas (parágrafos, cabeçalhos), pequenos *scripts*, objectos *ActiveX*, etc.

[1] Para os telefones celulares já existe uma linguagem específica, o *Wireless Markup Language* - WML . Esta é uma aplicação XML cujo DTD está disponível a partir do endereço http://www.wapforum.org//DTD//wml_1.1.xml

Na prática, as marcas de início e de fim dos elementos HTML, HEAD e BODY podem ser omitidas, pois podem ser inferidas pelo analisador em conformidade com o DTD.

Qualquer documento em conformidade com a especificação do HTML deve iniciar-se com a declaração <!DOCTYPE>, que no exemplo a seguir apresentado, permite ao processador obter a informação de que o conteúdo do documento está em HTML, versão 3.2. É de salientar que esta indicação da versão do HTML usado, define qual o DTD a usar. De igual forma, o elemento TITLE também deverá ser incluído. Assim sendo, o mínimo necessário para se ter um documento HTML é o seguinte:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<TITLE>A minha página pessoal</TITLE>
```

Document Type Definition – DTD de um documento HTML

O DTD de um documento, dita a sua estrutura, os seus elementos e atributos. Sendo assim, é natural que cada versão de HTML tenha um determinado conjunto de DTD's específicos.

Por exemplo, o HTML 4.0 especifica três DTD's, sendo assim os autores destes documentos têm de incluir uma das seguintes declarações nos seus documentos, dependendo da variedade de elementos que eles suportam.

- O DTD *Strict* inclui todos os elementos e atributos que não foram descontinuados ou não aparecem em documentos com *frames*. Nesses documentos deverá ser incluída a declaração:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
"http://www.w3.org/TR/REC-html40/strict.dtd">
```

- O DTD *Transitional*, inclui todos os elementos do DTD *Strict* mais os elementos descontinuados. Nesses documentos deverá ser incluída a declaração:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/REC-html40/loose.dtd">
```

- O DTD *Frameset*, inclui todos os elementos do DTD *Transitional*, assim como as *frames*. Nesses documentos deverá ser incluída a declaração:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"
"http://www.w3.org/TR/REC-html40/frameset.dtd">
```

O caso desta versão de HTML não é única pois a próxima geração de HTML, o XHTML também tem os mesmos três DTD's.

A RFC 1866 [TBDC1995], refere também para o HTML 2.0 um DTD com três variantes, são eles: DTD *Strict*, DTD *Level 1* e o DTD *Strict Level 1*.

Pontos fracos do HTML

Como já foi referido, o HTML sofre de limitações na sua capacidade de posicionamento dos objectos que o constituem. Esta é justificada pela necessidade de independência quanto à plataforma que obviamente dificulta a criação de mecanismos precisos para o posicionamento. A capacidade de variar a forma como os objectos são apresentados, também é limitada. Todas as capacidades do HTML são intrínsecas à respectiva versão ou por outras palavras, dependente dos elementos definidos na DTD. A carência de extensibilidade do HTML, leva a que os utilizadores do HTML tenham de esperar por uma nova versão onde talvez, venham a ser especificados novos elementos, capazes de dar resposta às suas necessidades.

Outra carência do HTML é a inadequada capacidade de dar estrutura e significado à informação. As folhas de estilo *Cascading Style Sheets* – CSS [HLBB1996] (ver em 3.1.3), são a forma como essa limitação é contornada, permitindo que seja feita alguma separação do conteúdo e da sua apresentação. Havendo uma separação da informação e da descrição como esta é apresentada, torna-se muito mais eficiente a sua manipulação. Uma nova especificação, o XML 1.0, e*Xtensible Markup Language* [TBJPCS1998], mais próxima do SGML, graças à sua extensibilidade e total capacidade de separação dos dois tipos de informação, permite uma gestão da informação contida nos seus documentos muito mais eficiente e poderosa .

3.1.1.3 XML – *eXtensible Markup Language*

O XML [TBJPCS1998] é um formato portável e extensível de descrição de documentos, que dada a sua crescente popularidade poderá tornar-se num formato universal de dados, para a partilha de informação.

Pode ser considerado um subconjunto, ou como uma forma mais restrita do SGML, cujo objectivo é permitir que documentos genéricos, SGML, possam estar disponíveis para envio, recepção e processamento, via *Web*, do mesmo modo que hoje em dia é possível com os documentos HTML.

Foi concebido para permitir uma fácil implementação e interoperabilidade com o SGML e o HTML. Nesta linguagem, ao contrário do SGML, do qual o XML é um caso particular, muitos aspectos do XML são fixos, já que o SGML é extremamente amplo e genérico. Um destes aspectos o qual pode referir-se a título de exemplo é o seguinte: o XML especifica a escolha da sintaxe de caracteres, de tal modo que qualquer utilizador XML usa a mesma sintaxe de caracteres. Todas as marcas XML começam por '<' (sinal menor) e terminam com o carácter '>' (sinal maior).

A extensibilidade do XML

A especificação XML, como um caso particular da norma ISO 8879, a SGML, permite dar aos seus utilizadores total autonomia no que toca à criação de novas marcas, o que não acontece no HTML. É neste aspecto que o XML é inovador. Enquanto que outras linguagens, como é o caso do HTML, definem apenas um conjunto fixo de elementos e se não existe um dado elemento então é necessário esperar por uma nova versão, o XML, como uma meta

linguagem de definição de linguagens de marcadores, dá aos seus utilizadores a capacidade de definir novos elementos, desde que os elementos e as suas marcas, estejam organizados de acordo com um conjunto de princípios [TBJPCS1998]. Por exemplo, se é necessário representar uma família e toda a sua informação genealógica, certamente é necessário representar pessoas, datas de nascimentos, falecimentos, casamentos e divórcios, etc.. Então, basta criar o DTD que especifique os elementos necessários, as respectivas marcas e a estrutura hierárquica. Assim, deixa-se de forçar a informação a encaixar-se em parágrafos, listas de itens, tipos de fontes, etc., como acontece ao usar o HTML.

Aplicações

Esta linguagem foi desenvolvida pelo *SGML Editorial board*, formado sob os hospícios do *World Wide Web Consortium* (W3C) em 1996. Depois do lançamento da especificação 1.0 em Fevereiro de 1998, a linguagem XML já entrou nos meios científicos e nos domínios das ciências, do grafismo, do reconhecimento de voz, do comércio electrónico. Uma prova disso são as inúmeras aplicações do XML, utilizadas na definição de linguagens para áreas específicas como: a matemática, com a *Mathematical Markup Language* - MathML[1], a química, o *Chemical Markup Language* – CML[2], nas comunicações moveis, a *Wireless Markup Language*, anteriormente conhecida como *Handheld Devices Markup Language* - HDML, a música *Music Markup Language* – MusicML e em aplicações de voz VoxML[3]. As folhas de estilo *eXtensible Style Language* - XSL, são também uma aplicação XML.

Algumas das grandes companhias adoptaram estratégias viradas para o XML. São alguns exemplos dessa aposta no XML é o trabalho levado a cabo pelas empresas:

- A IBM, por exemplo, considera o XML tão determinante como o *Java* para fazer progredir o comércio electrónico com base em diferentes plataformas informáticas. A IBM propõe ferramentas evoluídas, processadores (*parsers*) desta linguagem e começa a introduzir o XML nos seus produtos, nomeadamente no DB2 e no barramento interaplicacional MQSeries .

- A *Microsoft* baseia igualmente no XML a próxima geração da sua oferta neste domínio. Concretamente, o *framework BizTalk*, anunciado em Março de 1999 pelo próprio *Bill Gates*. Completa a arquitectura integradora da *Microsoft – Windows* DNA - *Distributed Network Architecture*, para comércio electrónico.

- A SAP, está a abrir a sua interface BAPI – *Business Application Programing Interface* à linguagem XML, a fim de optimizar a interoperabilidade das aplicações com a sua solução R/3.

- Por sua vez, a *Sun Microsystems* criou um grupo para o desenvolvimento de uma extensão XML para a sua plataforma *Java*. Além disso, a *Sun* utilizará o XML para armazenar as propriedades dos seus componentes de servidor *Enterprise JavaBeans* de forma normalizada,

[1] Mais informações em http://www.w3.org/TR/2000/WD-MathML2-20000328

[2] Mais informações em http://www.xml-cml.org

[3] Desenvolvida pela Motorola, mais informação disponível em http://www.voxml.com

independente do servidor de aplicações. Este facto melhorará a portabilidade dos *Enterprise JavaBeans*.

Estrutura de um documento XML

Sendo o XML uma especificação baseada na norma SGML, é composto da mesma forma por: um cabeçalho, facultativo, uma árvore de elementos, a instância do documento, e finalmente instruções de processamento, igualmente facultativas e destinadas às aplicações externas. A definição de tipo de documento e os aspectos relacionados com a apresentação são habitualmente tratados à parte, em DTDs e folhas de estilo. No entanto, um documento XML poderá ser autónomo, e possuir uma indicação standalone='yes' no cabeçalho. Neste caso o DTD e folhas de estilo constituem parte integrante do documento XML. Um exemplo deste caso é o seguinte documento XML, estando o DTD descrito na marca <!DOCTYPE >.

```
<?xml version='1.0' standalone='yes'?>
<!DOCTYPE MEU_DOCUMENTO [
    <!ELEMENT MEU_DOCUMENTO ANY>
    <!ELEMENT TITULO (#PCDATA)>
    <!ELEMENT TEXTO  (#PCDATA)>
    <!ENTITY  ola  "Ol&#225;">
]>
    <MEU_DOCUMENTO>
    <TITULO>
    XML
    </TITULO>
    <TEXTO>
    &ola; XML !!!
    </TEXTO>
</MEU_DOCUMENTO>
```

Abrindo este documento com o *Internet Explorer 5.0* o resultado é o seguinte:

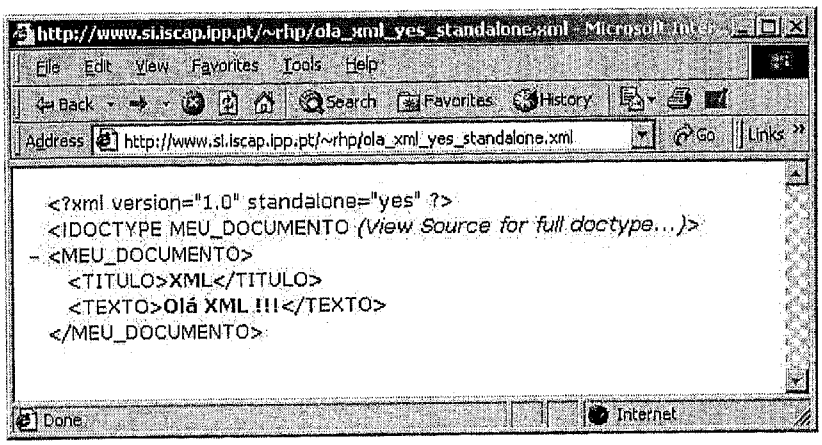


Figura 16 - Exemplo de um documento XML autónomo

A DTD embutido no documento XML define um tipo de documentos MEU_DOCUMENTO, constituído por elementos do tipo MEU_DOCUMENTO. Este elemento, por sua vez é composto por outros dois elementos: TITULO e TEXTO e a entidade `ola`. O elemento MEU_DOCUMENTO está declarado como ANY, o que significa que pode conter qualquer tipo de dados, neste caso dois elementos. Os elementos TITULO e TEXTO são do tipo `#PCDATA`, o que significa que o elemento não pode conter mais nenhum elemento dentro dele, apenas informação. A entidade `ola` está definida para representar a letra 'O', seguida de 'l' e o carácter 225 ('á'), formando a palavra "Olá". É graças a esta definição, no DTD, que o processador XML reconhece a entidade ola e a substitui durante o processamento do documento XML.

Existe ainda possibilidade da DTD estar definida fora do documento XML, nesse caso na declaração XML, a indicação `standalone='no'` indica ao processador que existe dependência de outros ficheiros. O DTD, cuja referência é feita através da expressão `<!DOCTYPE MEU_DOCUMENTO SYSTEM 'ola_xml.dtd'>`, a qual indica ao processador que a DTD "MEU_DOCUMENTO" está no ficheiro (ou URL) "ola_xml.dtd". Nesse caso, o documento XML tomaria a seguinte forma:

```
<?xml version='1.0' standalone='no'?>
<!DOCTYPE MEU_DOCUMENTO SYSTEM 'ola_xml.dtd'>
<MEU_DOCUMENTO>
   <TITULO>
      XML
   </TITULO>
   <TEXTO>
      &ola; XML !!!
   </TEXTO>
</MEU_DOCUMENTO>
```

O DTD terá o seguinte código:

```
<!ELEMENT MEU_DOCUMENTO ANY>
<!ELEMENT TITULO (#PCDATA)>
<!ELEMENT TEXTO  (#PCDATA)>
<!ENTITY  ola    "Ol&#225;">
```

E o resultado seria o seguinte:

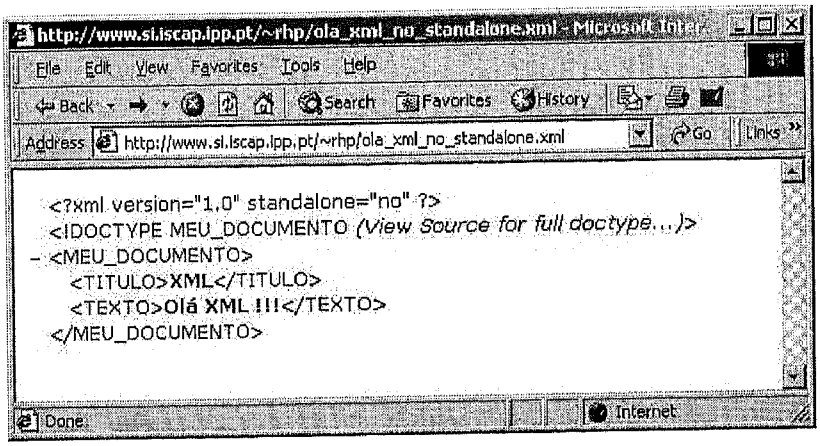


Figura 17 - Exemplo de um documento XML não autónomo

De facto, a única diferença visível é o código XML. É o código XML que o navegador *Internet Explorer 5.0* exibe, isso acontece porque o seu processador não sabe qual o aspecto visual dos elementos, definidos no DTD e posteriormente utilizados no documento XML. Num documento XML é definida a estrutura e o sentido da informação, os aspectos ligados à apresentação são deixados ao cuidado das folhas de estilo. Mais à frente, quando forem abordadas as folhas de estilo *Cascading Style Sheets* - CSSs e as folhas *eXtensible Stylesheet Language* - XSL, este exemplo será completado (ver os exemplos nas páginas 81 e 83).

3.1.2 Linguagens de descrição de página

3.1.2.1 PostScript

A linguagem *Postscript* [ADOBE1985] é uma linguagem de programação interpretativa simples com capacidades gráficas potentes. A sua primordial aplicação consiste em descrever a aparência de texto, formas gráficas e imagens, em páginas impressas. Nesta linguagem, um programa pode ser usado para comunicar a descrição de um documento imprimível oriundo de um sistema de composição para um sistema de impressão. A descrição é de alto nível e independente do periférico.

A capacidade desta linguagem de descrição de página, inclui as seguintes facetas, as quais poderão ser utilizadas em qualquer combinação:

- Formas arbitrárias construídas a partir de linhas rectas, arcos e curvas cúbicas; essas formas podem intersectar-se a si mesmas, ou entre si, e contêm secções e buracos desconexos.

- Figuras primitivas cuja forma pode ser desenhada com linhas de qualquer espessura, preenchidas com qualquer cor, ou usada como fronteira tendo em vista cortar quaisquer outros gráficos.
- Texto muito integrado com gráficos. No modelo de gráficos do *Postscript*, os caracteres de texto (tanto nas fontes normalizadas como nas definidas pelo utilizador) são tratados como formas gráficas, que podem ser operadas por quaisquer operadores de gráficos *Postscript*.
- Imagens derivadas de fontes naturais (ex.: fotografias) ou geradas sinteticamente. O modelo gráfico *Postscript* permite obter imagens de qualquer resolução e com limites dinâmicos. Proporciona, ainda, inúmeras funcionalidades para controlar o processo de *Rendering* das imagens num dispositivo de saída.
- Uma funcionalidade de sistema que suporta todas as combinações de transformações lineares, incluindo translação, escalamento, rotação, reflexão e inclinações. Estas transformações podem ser aplicadas uniformemente a todos os elementos da descrição de uma página, incluindo texto, formas gráficas e imagens.

A operação de *Rendering* da descrição de uma página *Postscript,* pode ser obtida numa impressora ou noutro dispositivo de saída, quando esta é entregue a um interpretador *Postscript* que a controla. À medida que o interpretador executa os comandos *Postscript,* para desenhar os caracteres, figuras, imagens, etc., estas são convertidas de uma descrição *Postscript* de alto nível, para um formato de dados de baixo nível, específico do dispositivo de saída.

Normalmente, as descrições da página *Postscript* são automaticamente geradas por programas de composição, tais como: processadores de texto, ilustradores, sistemas CAD, entre outros. Geralmente os programadores escrevem programas *Postscript* apenas quando criam novas aplicações. Todavia, em situações especiais, um programador escreverá programas *Postscript* para tirar partido das capacidades do *Postscript* não acessíveis através de um programa aplicacional particular.

As extensas capacidades descritivas de página *Postscript* são embutidas numa linguagem de programação genérica. Esta linguagem inclui um conjunto de tipos de dados tais como: números, vectores (*arrays*) e sequências de texto (*strings*). Define estruturas de controlo primitivas, tais como condicionais e ciclos. Procedimentos e algumas facetas pouco usuais, tais como dicionários (tabelas associativas), também são definidos. Tudo isto, permite aos programadores de aplicações definir operações de alto nível, que são exactamente adequadas às necessidades da aplicação, gerando então descrições de página que invocam essas operações de alto nível. Tal descrição de página é mais compacta e fácil de gerar que outra escrita inteiramente em termos de um conjunto fixo de operações básicas.

Os programas em *Postscript* são criados, transmitidos e interpretados na forma de código fonte, não existe uma forma compilada ou codificada desta linguagem. A linguagem é totalmente definida em termos de caracteres, susceptíveis de serem impressos, como se de um texto se tratasse. Esta representação é conveniente aos programadores para criar, manipular, e entender o código *Postscript*. O conjunto limitado de caracteres facilita o armazenamento e transmissão de ficheiros *Postscript* sob os diversos sistemas, fomentado a independência relativamente à plataforma.

Estrutura de um documento *Postscript*

Um documento *PostScript* geralmente consiste em duas partes: um prólogo seguido de um *script*. O prólogo contem definições específicas, que são usadas no *script*.

Para melhor se compreender a estrutura de um programa *Postscript*, o seguinte exemplo tenta demonstrar a necessidade e a funcionalidade do prólogo, face ao *script*. O *PostScript* não suporta a primitiva que desenha um rectângulo. Desta forma, um programa *Postscript* para construir um rectângulo, tem de definir um caminho rectangular, especificando quatro segmentos de recta, e então, desenhar as linhas, ou preencher a área com uma cor qualquer. Se o rectângulo é frequentemente utilizado, na descrição de uma página, será de todo o interesse definir um procedimento parametrizável para o desenhar, e sempre que for necessário este pode ser invocado, com os devidos parâmetros. A definição desse procedimento é feita no prólogo.

A parte referente ao *script*, consiste numa sequência de páginas separadas. Cada página pode ser totalmente autónoma das restantes, isto é, não existe qualquer interdependência entre elas. Cada uma delas depende apenas das definições presentes no prólogo. A linguagem dispõe de funcionalidades que permitem garantir essa independência. Esta independência entre as páginas traz enormes vantagens para as operações de "Abrir" e "Guardar", e também na gestão dos recursos de memória.

Nesta linguagem não existe nada que formalmente distinga o prólogo do *script*, ou que exija a independência do *script* relativa à página. Tal estrutura é uma mera convenção, a qual será abordada mais à frente, contudo para muitas aplicações é recomendável que assim o seja.

O prólogo é geralmente criado manualmente, enquanto o *script* é gerado automaticamente, utilizando as definições presentes no prólogo.

De seguida é apresentado um outro exemplo, agora de um documento *PostScript*, ou mais precisamente, o esqueleto de um documento *Postscript*. Facilmente se deduz do exemplo, que as propriedades são definidas à custa de linhas iniciadas por %%, seguidas por uma palavra chave e o respectivo valor. Cada uma dessas definições é designada por comentário. Estes os comentários *Postscript*, estão divididos em três classes: cabeçalho, corpo e cauda.

```
%!PS-Adobe-1.0
%%Creator: Rui Pereira
%%Title: Dissertação
%%CreationDate: Fri Mar 17 12:33:00 2000
%%Pages: (atend)
%%DocumentFonts:(atend)
%%BoundingBox: 0 0 612 792
%%EndComments
... o prólogo do documento é inserido aqui, se houver ...
%%EndProlog
%%Page: 0 1
```

```
... a primeira página ...
%%Page: 1 2
... a segunda página ...
%%Page: 2 3
... a terceira página ...
%%Trailer
... a cauda do documento, se houver ...
%%DocumentFonts: Times-Roman Times Italic Times-bold
%%Pages: 3
```

Interpretador

Para facilitar a interpretação do ficheiro *Postscript*, existe um conjunto de convenções de estrutura. Um documento (programa) *Postscript* que obedece a essas convenções de estrutura, é designado por documento em conformidade. A convenção da estrutura não tem qualquer efeito sobre a execução do programa por um interpretador *Postscript*. No entanto, muitas aplicações só aceitam programas em conformidade com essas convenções.

Um requisito das convenções de estrutura é que deverá ser capaz de obter toda a informação de estrutura da descrição de uma página sem ter de interpretar ou executar todo o programa.

As convenções de estrutura, fazem uso dos comentários *Postscript* para representar esta informação.

3.1.2.2 PDF

O *Portable Document Format* – PDF [ADOBE1999] é um formato de ficheiros para guardar documentos, graficamente e tipograficamente complexos. O formato PDF garante que o *layout* seja preservado tanto no ecrã como na impressão. Todas as definições relativas ao *layout* de um documento são componentes fixos, isto é, não é permitida nenhuma variação na forma como estas são interpretadas. Assim a aparência de um documento PDF é completamente fixa.

O formato PDF tem as suas origens num velho e poderoso predecessor, o *PostScript*, uma linguagem de descrição de página.

O PDF usa os elementos de descrição de página do *PostScript*, mas por questões de simplicidade não usa estruturas de controlo, tais como condições e ciclos. Adicionalmente, o formato PDF usa muitas funções de hipertexto e elementos iterativos, para enriquecer o documento.

A estrutura e conteúdo de um ficheiro PDF é difícil de distinguir: existem objectos directos e indirectos, tabelas com posições em ficheiros, compressão e encriptação. Para ler e entender um ficheiro PDF é necessário ter imensa experiência em programação, o que não é necessário para ler e interpretar um ficheiro HTML ou XML. Criar um ficheiro PDF menos trivial à mão,

num editor de texto, constitui um exercício que derrota o programador mais experiente. Na realidade, para criar um ficheiro PDF é necessário *software* apropriado. Tal *software* é disponibilizado pela empresa *Adobe Systems*[1], mas cada vez mais surgem outros fabricantes. A *Adobe* é a criadora tanto do *PostScript* como do PDF, e é também a guardiã de ambas as normas. No caso do PDF, este termo é merecido, pois não existem variantes ou dialectos específicos de fabricante. Desde que a *Adobe* introduziu o *Acrobat* em 1993, com o PDF 1.0, surgiram posteriormente apenas três extensões do formato, onde se destacam duas delas: o PDF 1.2, base do *Acrobat 3.0,* e mais recentemente, o PDF 1.3 [ADOBE1999], a base do *Acrobat 4.0.*

Apesar de geralmente se encarar formatos de ficheiros mantidos por um único fabricante de uma forma bastante céptica, esta situação teve as suas vantagens. Com o *Acrobat*, a *Adobe* conseguiu mostrar que é possível desenvolver e colocar no mercado, com sucesso, um formato poderoso, face à existência de outros formatos. A ampla distribuição e aceitação das aplicações *Adobe*, na área gráfica, tornaram mais fácil a aceitação do novo formato, por parte deste importante grupo de utilizadores. A distribuição livre do programa para a visualização, o *Acrobat Reader*, via *Internet,* foi também muito importante para o seu sucesso. A adopção do *Acrobat* pela industria de impressão e a sua integração com a pré impressão, com avultados investimentos, foi também um factor na estabilidade do formato PDF.

Estrutura de um documento PDF

Ao contrário do *Postscript* e do HTML, um ficheiro PDF é auto-suficiente, isto é, contém todas as fontes, imagens e outros componentes de que necessita. Um único ficheiro PDF pode conter um documento, qualquer que seja o seu tamanho, isto é, sem necessitar de nenhum outro ficheiro contendo objectos nele inseridos.

Um documento PDF é totalmente orientado para a apresentação, e não existe nenhuma preocupação com a estrutura do texto e do documento. A unidade básica de um documento (ficheiro) PDF é a página. O texto não é guardado como um conjunto de objectos estruturados, como parágrafos ou cabeçalhos mas, tal como no *Postscript*, como elementos de *layout*: caracteres, palavras ou linhas. Naturalmente isto torna mais difícil a posterior modificação do texto, pois o processo de recriar a informação de estrutura não é muito fácil.

Um ficheiro PDF tem uma sintaxe extremamente complexa, que muitos dos programadores, mesmo os mais experientes, não conseguem dominar. No entanto, no anexo B é exibida uma listagem relativa a um pequeno exemplo, tentando-se que seja o mais simples possível, o qual poderá ajudar na compreensão deste formato. O resultado deste ficheiro, também em anexo, é simplesmente uma frase. Dois aspectos a salientar relativamente a este exemplo são: o local onde a frase é colocada, e as coordenadas da sua localização.

[1] Página da empresa na *Internet* http://www.adobe.com

3.1.3 Apresentação dos documentos

HTML *versus* PDF

Pode ser verificado pelas descrições precedentes, de ambos os formatos, que o HTML e o PDF têm fortes propriedades complementares. Os predecessores totalmente diferentes do HTML e PDF, SGML e *PostScript*, respectivamente, com as suas estritas ênfases na estrutura ou apresentação, dão ao HTML e ao PDF os seus pontos fortes e os fracos.

Como no mundo real ambos são significativos e não tem sentido haver disputa entre conteúdo estrutural de um e a aparência agradável de outro. Não é de surpreender que ambos os formatos se estejam a aproximar, mais do que nunca estes estão perto um do outro. A comunidade do HTML está permanentemente a incluir novas marcas, com o intuito de adicionar novos controlos de *layout*. Por outro lado, o PDF já contém o mesmo hipertexto e funções de formulários, tal como já acontece há muito com o HTML, mas continua incapaz de exportar uma tabela oriunda de um documento sem criar uma terrível confusão.

A tabela a seguir apresentada, faz uma comparação do HTML com o PDF. Compara aspectos relativos à apresentação e à estrutura, entre outros, de cada um dos tipos de formato de documentos.

Característica	HTML	PDF
Responsabilidade pelo formato	Consórcio W3 e fabricantes	Adobe
Standard	Muitas variantes	Standard uniforme
Visualizadores para diferentes sistemas operativos	Navegadores livres e comerciais	*Adobe Acrobat Reader* (distribuído gratuitamente)
Software para criar documentos	Ferramentas livres e comerciais	Ferramentas comerciais (maior parte); algumas ferramentas livres
Editando/alterando documentos	Simples, usando editores de texto ou HTML	*Acrobat* (limitado) ou programas adicionais
Informação sobre um documento	Meta marcas	Campos de informação do documento
Criação de novos documentos (sem usar rotinas de conversão)	Com programas de autoria HTML	Dificilmente possível
Estrutura do documento	Muito boa	Pobre

Facilidade de pesquisa	Boa	Boa
Ferramentas de pesquisa	Excelente	Excelente
Funções de hipertexto	Excelente	Excelente
Qualidade da apresentação dos documentos	Pobre	muito alta
Quem determina a aparência do documento?	Utilizador	Autor
O *layout* adapta-se às condições de visualização	Sim	Não
O documento está contido em um ficheiro	Não (os gráficos são em separado)	Sim (com a excepção de ficheiros de vídeo)
Aptidão de visualização	Muito elevada	Alta, se correctamente formatada
Aptidão para imprimir	Baixa	Alta, se correctamente formatada
Layout e tipografia	Pobre	Muito boa
Formato de ficheiro	Marcas simples	Complicada
Geração dinâmica no servidor	Fácil	Difícil
Tamanho do ficheiro	Pequeno	Médio a grande

Tabela 1 – Comparação HTML/PDF

HTML *versus* XML

Inicialmente o XML foi apresentado como um substituto potencial do HTML. Na realidade, este último sofre de muitas limitações, ligadas à mistura do conteúdo com a informação relativa à apresentação. No fundo o HTML é uma linguagem que permite apenas apresentar informação. Recordando um exemplo já utilizado, uma página HTML poderá comportar a indicação: "<B> Livro de matemática = 100 páginas A4 </B>". Isto significa apenas que o texto entre as marcas deverá ser apresentado em negrito. Em contrapartida, a indicação não revela nada sobre a natureza dessa informação (número e formato das páginas do documento). Quem consulta a página HTML lerá a expressão: "**Livro de matemática = 100 páginas A4**" apresentada num tipo de fonte carregada, e provavelmente, deduzirá que o documento em causa tem 100 páginas e que o formato dessas páginas é o formato A4(210 x 297 mm). Mas,

apesar desta dedução ser possível para um ser humano, um computador não tem a mesma capacidade. A linguagem XML permita dar resposta a necessidades deste género, dado que é uma linguagem que marca o tipo de dados, dando-lhes significado, por forma a torná-los compreensíveis, mesmo por diferentes computadores, em qualquer plataforma.

Assim se concluí que existem duas grandes diferenças entre o HTML e o XML. A primeira é a capacidade do XML para separar o conteúdo da apresentação. Ou seja, um documento XML não contém mais do que os dados e os marcadores sobre os dados (que indicam o sentido dos mesmos). A segunda grande diferença entre estas duas linguagens é a extensibilidade da linguagem XML. Na realidade, é possível definir-se novos marcadores, de acordo com as necessidades do utilizador. Esta linguagem é de facto, uma meta linguagem, ou seja, uma linguagem de descrição de linguagens, estando muito mais perto do seu antecessor, o SGML. Sendo assim, cada área cientifica, área industrial, ou até cada empresa, pode criar um derivado do XML. O que já é um facto, o caso das linguagens: MathML, VoxML, CML, entre outras.

Folhas de Estilo CSS para o HTML

O HTML tem as suas raízes na SGML, a qual sempre se distinguiu por ser uma linguagem de especificação de informação através de marcadores.

Com o decorrer da evolução do HTML, cada vez mais os seus elementos e atributos virados para a apresentação, deram lugar a outros mecanismos, em particular às folhas de estilo. A experiência mostrou que separando a estrutura do documento da sua apresentação, se reduz o custo de o disponibilizar numa ampla gama de plataformas, tipos de informação, etc..

O W3C tem promovido activamente o uso de folhas de estilo na WWW, desde a sua fundação em 1994. Esta actividade resultou em duas recomendações do W3C, *as Cascading Style Sheets, level 1 e level 2*, CSS1 e CSS2 respectivamente, implementadas na generalidade dos browsers, embora não da forma mais consistente.. A *Cascading Style Sheets, level 1*, surgiu como recomendação W3C [HLBB1996] em Dezembro de 1996. Descreve a linguagem CSS como um modelo simples de formação visual. A *Cascading Style Sheets, level 2*, surgiu como recomendação W3C [BHCI1998] em Maio de 1998, assenta na CSS1 e acrescenta o suporte específico de folhas de estilo para impressoras e outros dispositivos. Acrescenta também à CSS original a capacidade do uso de fontes susceptíveis de serem descarregadas por download, e ainda elementos de posicionamento e de tabelas. Actualmente a terceira recomendação das *Cascading Style Sheets - level 3* está já em desenvolvimento.

As folhas de estilo têm como objectivo descrever como deverá visualmente ser apresentada a informação dos documentos, em ecrãs, impressoras, ou, ainda, como deverá ser pronunciada.

As *Cascading Style Sheets* - CSS constituem um mecanismo simples para adicionar os estilos de fonte, cor, espaçamento, etc. Ao agregar folhas de estilo a documentos estruturados (HTML por exemplo) na WWW, torna-se possível influenciar a forma de apresentar esses documentos, sem ser necessário sacrificar a independência relativamente ao dispositivo de apresentação, ou de criar-se novos marcadores.

Folhas de Estilo XSL para o XML

A actividade do W3C no campo das folhas de estilo, não se restringe às CSS, vocacionadas para documentos HTML. Esta actividade estende-se dando origem a um grupo de trabalho incumbido de delinear uma nova linguagem de estilos, cujo principal alvo seria o XML. As folhas de estilo *eXtensible Stylesheet Language* – XSL foram propostas ao W3C em Agosto de 1997. A esse primeiro documento seguiram-se três designados de *Working Drafts*. Em Abril de 1999, a par com o terceiro documento de *Working Draft*, surge separadamente um documento que especifica uma linguagem de transformação, o XSLT [W3C1999a]. Em Julho de 1999 surge uma linguagem de expressão, a Xpath [W3C1999b]. Estas duas linguagens, uma de transformação e outra de expressão, tornam-se em recomendação em Novembro de 1999.

Em Janeiro de 2000, a Microsoft lança a versão 1.0 de um conversor XSL para XSLT. Trata-se de uma folha de estilo que actualiza a folha de estilo XSL do seu navegador, o *Internet Explorer 5.0,* para a folha de estilo compatível com a recente recomendação do XSLT.

Nos objectivos de uma folha de estilo como o XSL, constata-se que esta tem uma dupla faceta. Se por um lado as folhas de estilo XSL têm a capacidade de extrair a informação contida num documento XML, comportando-se este como uma base de dados, por outro lado, estas folhas de estilo têm também a capacidade de transformar essa mesma informação. Da evolução dos acontecimentos acima relatados torna-se visível a separação dessas duas facetas do XSL em duas recomendações separadas: A Xpath uma linguagem de expressão de caminhos, capaz de descrever o caminho até um elemento, numa estrutura arborescente como a do XML, e o XSLT, uma linguagem de transformação capaz de transformar a informação de um documento, dando origem a outro documento, XML ou outro.

Em Setembro de 1998, foi proposta a criação de uma linguagem de interrogação de documentos XML [W3C1998a]. Nesse mesmo ano, em Novembro, num documento designado por "*The Query Language Position Paper of XSL Working Group*" [W3C1998b], o *XSL Working Group* manifestava a sua opinião relativamente à possibilidade de ser criada uma linguagem, extensão do XSL, específica para interrogar documentos XML. A posição do grupo de trabalho relativamente à criação da nova linguagem de interrogação era a de não haver essa necessidade, isto porque, já tinham sido desenvolvidas as tecnologias relevantes a uma linguagem de interrogação a documentos XML. Essas tecnologias seriam os *XSL patterns* e as *XSL templates*, para a extracção da informação e a materialização dos resultados da extracção, respectivamente. Nesse documento, o grupo de trabalho apontava um conjunto de características das folhas de estilo XSL, que já as colocavam como um mecanismo de interrogação a documentos XML. Acreditava-se que uma linguagem de interrogação não ia melhorar a eficiência das folhas de estilo XSL. As recomendações das linguagens Xpath e XSLT são o resultado do trabalho do grupo nesta área.

Actualmente já existe um grupo de trabalho no W3C, dedicado ao estudo dos requerimentos para a interrogação de documentos XML. O último *draft* deste grupo de trabalho[1] publicado em 15 de Agosto do presente ano, aponta como objectivos: a criação de um modelo de dados para documentos XML, um conjunto de operadores de interrogação para esse modelo, e uma linguagem de interrogação, baseada nesses operadores. Nesse mesmo documento, é

[1] Disponível em http://www.w3.org/TR/xmlquery-req.html

estabelecida a mesma relação, desta, com as linguagens XSLT, Xpath e Xpointer. Refere também que este grupo de trabalho deve tomar em consideração as capacidades de expressão e pesquisa do Xpath, quando for definida a álgebra e sintaxe de interrogação dessa linguagem.

Seguidamente as *patterns* e os *templates*, das folhas de estilo XSL, serão abordados. Nessa abordagem tornar-se-ão claros os objectivos e as funcionalidades destas folhas de estilo. Mais à frente, no ponto relativo aos aspectos ligados à gestão da informação (ver em 3.1.4) serão abordados os pontos de intercepção entre as folhas de estilo XSL com as linguagens de interrogação a bases de dados.

Patterns

As folhas de estilo do XML, as XSL, usam uma linguagem de padrões para extracção e identificação de elementos do documento XML. Estes padrões existem sob duas formas: *match patterns* e *select patterns*.

- As *select patterns* extraem todos os nós que correspondem a um critério especificado pela *select pattern*. Este processo é conhecido como "*selection*".

- As *match patterns*, por sua vez permitem saber se um dado nó do documento XML é identificado pela *pattern*. Este processo é conhecido como "*matching*".

A linguagem de *patterns*, tem uma sintaxe baseada no modelo utilizado nos sistemas de ficheiros para descrever o nome e caminho completo até um ficheiro qualquer. A razão pela qual isto acontece é perfeitamente compreensível. Um documento XML também pode ser encarado como um sistema de ficheiros, onde cada elemento do documento pode ser visto como um directório, e os elementos descendentes como subdirectórios. As barras separadoras, tal como num sistema de ficheiros, separam os diversos níveis hierárquicos dos elementos, da mesma forma que num sistema de ficheiros. A barra "/" significa a raiz do sistema e ".." significa o directório da hierarquia imediatamente a baixo da actual, numa *pattern* representam respectivamente o elemento raiz do documento XML e o nó do nível hierárquico imediatamente a baixo do actual..

As XSL estabelecem uma notação com qualificadores. Estes qualificadores atribuem ao nome do elemento no caminho, as restrições tanto ao nível do conteúdo como de atributos, dos elementos a serem seleccionados.

Templates

A folha de estilo XSL, examina o documento e aplica-lhe os *templates* aos elementos para criar um novo documento de resultado. As folhas de estilo XSL usam elementos xsl:template para definir como o documento resultado deverá ser construído. O conteúdo de cada um dos elementos da folha de estilo XSL é um *template* para o conteúdo a ser inserido no documento resultado.

A seguir são apresentados dois exemplos de *templates* XSL, nos quais são aplicados os dois tipos de *patterns* (*select* e *match*) anteriormente abordadas.

- *Template* que utiliza uma *pattern* do tipo *match*

```
<xsl:template match="TITULO">
     Título: <xsl:value-of/>
</xsl:template>
```

Para cada instância do elemento TITULO é construída uma linha de texto, com a palavra Título seguida de ":", seguida do valor textual do elemento TITULO.

- *Template* que utiliza três *patterns,* uma *match* e duas *selection*

```
<xsl:template match="DOCUMENTOS/CABECALHO">
     <xsl:value-of select="./TITULO"> <xsl:value-of select="./AUTOR"/>
</xsl:template>
```

Esta folha de estilo, permite obter um conjunto de linhas, que em conjunto formam uma espécie de tabela. A primeira coluna com o título de cada um dos documentos e a segunda com o autor. A *match pattern* selecciona todos os elementos CABECALHO dos documentos (cada documento também um elemento). A *select pattern*, usa o carácter "." para referir-se a cada um dos cabeçalhos, seleccionando assim o título e o autor. Neste exemplo, o carácter "." tem exactamente o mesmo papel quando num sistema de ficheiros é utilizado na seguinte sequência de comandos, considerando os elementos titulo e autor dois ficheiros de texto:

```
cd documentos/cabecalho
cat ./titulo
......
cat ./autor
......
```

Estes exemplos demonstram o uso das *patterns* e dos *templates*, para a extracção de informação de um documento e para construir um documento resultado. Recorrendo a folhas de estilo torna-se possível manipular a informação existente num documento XML.

Dois tipos de folhas de estilo

Uma vez que o XML permite separação do conteúdo da sua apresentação, o W3C introduziu as folhas de estilo XSL. O HTML não permite a mesma separação, porém o uso das CSS em documentos HTML permite que haja alguma separação da informação de conteúdo da informação relativa à forma como esse conteúdo é apresentado. O uso de folhas de estilo em documentos HTML permite aproximar o HTML do XML.

À volta destas duas folhas de estilo gera-se alguma confusão. A tabela seguinte, faz uma comparação dos principais aspectos ligados aos dois tipos de folhas de estilo, pretendendo assim fazer um ponto da situação do que foi dito até aqui.

	CSS	**XSL**
Podem ser usadas com o HTML ?	*Sim*	*Não*
Podem ser usadas com o XML ?	*Sim*	*Sim*
São uma linguagem de transformação?	*Não*	*Sim*
Sintaxe	*CSS*	*XML*

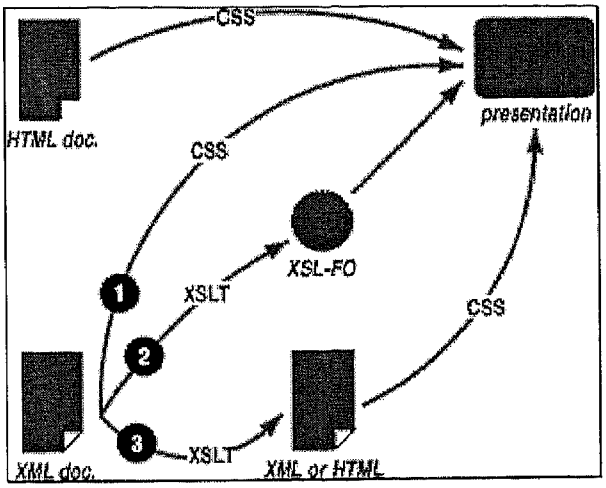
Tabela 2 - Comparação CSS/XSL

As CSS permitem apenas adicionar estilos, que podem ser utilizados para descrever como um dado elemento é apresentado. Enquanto isso, as XSL permitem a manipulação da informação e transformar os documentos. É claro que sendo possível transformar o documento, então é possível acrescentar a informação necessária à sua apresentação, por exemplo, a transformação de um documento XML num documento HTML, contendo todos os elementos relativos à apresentação.

Ao mesmo documento XML, é possível aplicar diferentes folhas de estilo e assim obter publicações adaptadas a diversos meios e dispositivos de saída. Desta forma, o XSL reforça assim a importância do XML e oferece a possibilidade de evitar o recurso a uma linguagem de programação para efectuar essas transformações.

As folhas de estilo CSS, são bastante mais simples de usar e entender. Para as favorecer ainda mais, actualmente existem muito mais editores WYSIWYG [1]para CSS do que para XSL.

[1] WYSIWYG- Acrónimo da expressão What I See Is What I Get



Este diagrama ilustra três forma de apresentar um documento XML. O primeiro caso, não prevê a transformação do documento, enquanto nos outros dois casos, o documento XML é transformado. No segundo, em objectos XSL-FO para posteriormente ser apresentado. No terceiro caso, o documento XML, é inicialmente transformado num documento HTML, que mais facilmente é visualizado.

De seguida serão apresentados dois exemplos que ilustram o primeiro e terceiro caso.

Figura 18 - Apresentação[1] de um documento XML

O documento XML utilizado para estes dois exemplos é o mesmo que já foi utilizado para a exemplificação das DTDs, que sem qualquer folha de estilo não permitia a visualização correcta dos elementos constituintes do documento XML.

Primeiro caso, uso da folhas de estilo CSS.

- Documento XML

```
<?xml version='1.0' standalone='no'?>
<?xml-stylesheet type="text/css" href="ola.css"?>
<!DOCTYPE MEU_DOCUMENTO SYSTEM 'ola_xml.dtd'>
<MEU_DOCUMENTO>
    <TITULO>
          XML
    </TITULO>
    <TEXTO>
          &ola; XML !!!
    </TEXTO>
</MEU_DOCUMENTO>
```

- Folha de estilo CSS (ficheiro ola.css)

```
TITULO
{
      COLOR: blue;
      DISPLAY: block;
      FONT-FAMILY: courier;
      FONT-SIZE: 10pt;
```

[1] Figura obtida a partir do endereço http://www.w3.org

```
        FONT-WEIGHT: bold;
}
TEXTO
{
        COLOR: blue;
        DISPLAY: block;
        FONT-FAMILY: arial;
        FONT-SIZE: 25pt;
        FONT-WEIGHT: bold;
}
```

- Resultado

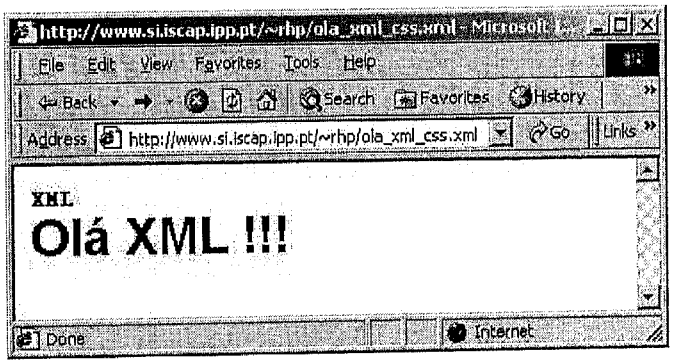


Figura 19 - Apresentação do documento XML recorrendo a CSS

Para que a folha de estilo fosse tomada em consideração na apresentação do documento foi necessário fazer uma alteração ao código inicial, fazendo-lhe referência no documento XML, na segunda linha do documento XML.

Neste exemplo, o papel da folha de estilo resume-se apenas a dar a indicação ao processador XML, que os elementos do tipo TITULO e TEXTO deverão ser visualizados com uma determinada fonte, tamanho, etc.. Relativamente ao mesmo exemplo, apresentado para a exemplificação dos DTD's (ver na página 67), nota-se que a folha de estilo permite a visualização do conteúdo do documento devidamente formatado, em vez de se ver o código XML. Mas toda a informação do documento XML é exibida sem qualquer transformação, ou efeito mais complexo.

As folhas de estilo XSL, permitem ir bastante mais longe do que isto, dado que permitem com mais alguma complexidade, a aplicação de efeitos mais elaborados. O exemplo que se segue usa um *template* XSL, para a transformação do documento XML, num outro documento, em HTML. Para que neste exemplo haja também uma definição das propriedades do tipo de letra usada na apresentação do conteúdo dos elementos do documento XML, foi utilizada uma folha de estilo CSS, aplicado ao resultado da transformação (em HTML). Sendo assim, como se pode constatar do *template* XSL, o conteúdo do elemento TITULO, é transposto para o

elemento TITLE do HTML, e TEXTO para o elemento BODY. Desta forma a folha de estilo tem de ser aplicada a estes elementos do HTML.

Terceiro caso, uso da folhas de estilo XSL.

- Documento XML

```
<?xml version='1.0' standalone='no'?>
<?xml-stylesheet type="text/xsl" href="ola.xsl"?>
<!DOCTYPE MEU_DOCUMENTO SYSTEM 'ola_xml.dtd'>
<MEU_DOCUMENTO>
   <TITULO>
      XML
   </TITULO>
   <TEXTO>
      &ola; XML !!!
   </TEXTO>
</MEU_DOCUMENTO>
```

- Folha de estilo XSL

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  <xsl:template match="/">
     <HTML>
        <HEAD>
          <TITLE>
            <xsl:for-each select="MEU_DOCUMENTO">
               <xsl:value-of select="TITULO"/>
            </xsl:for-each>
          </TITLE>
          <STYLE TYPE="text/css">
          BODY
          {
               COLOR: blue;
               DISPLAY: block;
               FONT-FAMILY: arial;
               FONT-SIZE: 25pt;
               FONT-WEIGHT:
               bold;
          }
          </STYLE>
        </HEAD>
        <BODY>
          <xsl:for-each select="MEU_DOCUMENTO">
```

```
            <xsl:value-of select="TEXTO"/>
          </xsl:for-each>
        </BODY>
      </HTML>
    </xsl:template>
  </xsl:stylesheet>
```

- Resultado

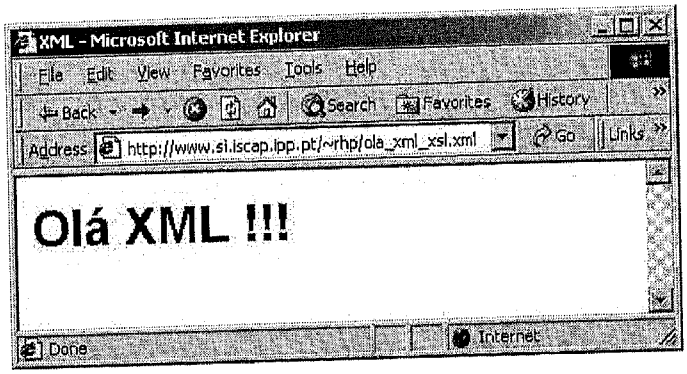


Figura 20 - Apresentação do documento XML recorrendo a XSL

A Figura 20 exibe o resultado da aplicação da folha de estilo XSL, ao documento XML. Como é possível constatar, a partir da figura, o conteúdo do elemento TITULO, já não aparece no mesmo local que o exemplo anterior, em vez disso, surge como título do documento visualizado no navegador. Este facto não é surpresa, pois se for feita uma análise do *template* presente na folha de estilo XSL, é possível verificar uma transposição do conteúdo do elemento TITULO, do documento XML para o elemento TITLE do novo documento HTML, o mesmo acontece com o elemento TEXTO, relativamente ao elemento BODY. Como esta folha de estilo XSL é usada apenas como *template*, para a transposição do conteúdos, foi usada outra folha de estilo, inserida no documento HTML, para a formatação dos estilos do texto. Como a frase "Olá XML!!!" faz agora parte do elemento BODY, do HTML, os elementos de estilo CSS definem os estilos para a apresentação desse elemento BODY.

Do HTML até ao XML

A base instalada do HTML, como é do conhecimento geral, é imensa, e isso, naturalmente tem de ser considerado. Sendo assim, torna-se óbvio que esta linguagem tem ainda muito tempo de vida pela frente. Quanto ao XML, a curto prazo, não terá impacto sobre os postos clientes. No entanto, e tal como acontece com o *Java*[1], este impacto é ainda ao nível do servidor.

[1] É o caso dos *Servlets*

Enquanto o HTML concentra todos os esforços na forma como apresentar a informação, o XML apenas comporta a informação, deixando ao cuidado da XSL os aspectos relativos à apresentação. No entanto, é possível do lado cliente, transformar automaticamente um ficheiro XML de modo a que possa ser apresentado por um cliente HTML. É isso que permite, por exemplo, a função *Data Binding* do DHTML[1], que pode ser utilizado pelo *Internet Explorer da Microsoft*, a partir da sua versão 4.0. A partir da versão 5.0 do Internet Explorer já não são necessários os *scripts* complexos do DHTML para a ligação dos dados à forma, o navegador já processa os documentos XML. Foi no *Internet Explorer* 5.0 que os dois exemplos, atrás apresentados, foram visualizados.

Mas é do lado servidor que o XML actualmente tem mais protagonismo, graças às folhas de estilo XSL a informação contida num documento XML facilmente é convertida para um documento HTML e enviada como se tratasse de um vulgar documento HTML, com as vantagens dai inerentes: a facilidade com que se torna possível alterar a apresentação da informação, jogando com a folha de estilo XSL, e algo ainda mais poderoso, a forma como podem ser feitas as ligações a bases dados que contêm a informação.

Os métodos clássicos de ligação de repositórios de documentos (estáticos ou armazenados em base de dados), a um servidor *Web* podem ser facilmente superados por esquemas de partilha de informação assente no XML. O HTML tem apenas a capacidade de exibir a informação, o XML, vai mais longe, funciona como um mecanismo de extracção e transformação da informação, e finalmente é capaz de a apresentar.

XHTML

A *Extensible HyperText Markup Language* - XHTML [W3C2000] surge como um passo na evolução do HTML em direcção ao XML, dando ao HTML a extensibilidade que lhe falta. O XHTML consiste numa família de tipos de documentos e módulos, que reproduzem, subaproveitam ou estendem o HTML 4. A família de tipos de documentos XHTML são baseados no XML e estão preparados para funcionarem em conjunção com os agentes XML.

A especificação XHTML 1.0 descreve o primeiro tipo de documento na família XHTML. Esta consiste numa reformulação dos três tipos de documentos do HTML 4, como aplicações do XML 1.0. Pretende-se que seja usado como uma linguagem para conteúdo, que esteja em conformidade com o XML e ao mesmo tempo, se determinados cuidados [W3C2000] forem tidos em conta, manter-se-á a compatibilidade com o HTML.

O XHTML é uma aplicação do XML, compatível com os agentes do HTML. A migração do HTML para o XHTML, traz vários benefícios.

- Os documentos XHTML são compatíveis com o XML e sendo assim podem ser manuseados por ferramentas XML.
- Os documentos XHTML podem ser escritos para serem tão bem, ou melhor, manuseados como eram antes pelos existentes agentes HTML 4, assim como nos novos agentes XHTML 1.0.

[1] O XML DSO (*Data Source Object*), disponibiliza estas funções que permitem a ligação do XML ao mundo do DHTML.

- Os documentos XHTML podem usar aplicações (ex: *scripts* e *applets*) que se baseiam tanto no *Document Object Model* - DOM [W3C1998c] do HTML como no DOM do XML.

Segundo os autores da especificação XHTML 1.0, a família XHTML é o próximo passo na evolução da *Internet*. Ao migrar para o XHTML, os produtores de conteúdos podem entrar no mundo do XML, com todos os seus benefícios, conseguindo manter a compatibilidade dos seus conteúdos, tanto para o passado, como para o futuro.

3.1.4 Manipulação da informação contida nos documentos

O XML e os SGDB

O XML está a tornar-se numa norma universal para a partilha de informação e se já é reconhecido como um mecanismo de partilha, também é um facto de que é reconhecido como um promissor agente de armazenamento de informação. O XML é capaz de não só funcionar como um interface de partilha de informação entre o SGBD e as aplicações, mas também de guardar informação, recorrendo à sua estrutura de dados arborescente. Dada esta dupla faceta do XML, colocam-se dois cenários possíveis:

- A criação de módulos de interface XML com os actuais SGBD's.

  Esta solução é a proposta dos fabricantes de SGBDR's, tais como a IBM, a *Oracle* ou a *SyBase*. A solução proposta por estas empresas consiste em acrescentar um subnível XML nos seus SGBDR's, permitindo assim a translação entre os dados arborescentes XML e as tabelas dos SGBDR's. Estas empresas explicam a sua opção com o argumento de que desta forma os seus clientes podem preservar os investimentos já realizados e beneficiar ao mesmo tempo das vantagens do XML.

- O desenvolvimento de bases de dados para o armazenamento nativo dos dados XML.

  Estas soluções são preconizadas por empresas como a *Object Design* e a *Software AG*, através das suas ofertas de servidores de dados XML, respectivamente o *eXcelon*[1] e o *Tamino*[2]. Estas empresas explicam a sua opção com o argumento de que tal tipo de servidor aumenta os desempenhos para armazenar, procurar, editar e partilhar os dados XML. Este tipo de servidores constituem estruturas em árvore, que se projectam com toda a naturalidade em bases de dados de objectos.

Projectar um modelo relacional num modelo arborescente.

[1] Mais informações em http://www.odi.com

[2] Mais informações em http://www.softwareag.com

A estrutura em árvore do XML é representada muito facilmente num modelo de objectos. Uma relação hierárquica entre dois objectos é feita por apontadores, situados entre cada um deles. Contrariamente, para projectar os documentos XML numa base de dados relacional, é necessário reparti-los por várias tabelas do SGBDR aquando de qualquer introdução de informação, bem como recompor os dados XML a partir das diferentes tabelas do SGBDR quando é efectuada qualquer leitura ou pesquisa da informação.

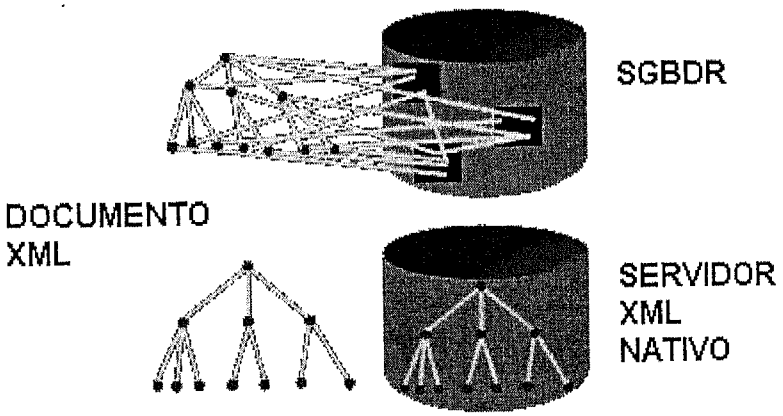


Figura 21 - Representação do documento XML num modelo relacional

Para melhor entender a forma como é feita a transposição dos dados estruturados segundo um modelo relacional para um modelo de objectos como o utilizado em documentos XML, considere-se o seguinte exemplo: três tabelas de uma base de dados relacional contêm a informação relativa a um repositório de documentos. A tabela ANOTACAO contém dois registos, os quais, por sua vez, contêm duas anotações relativas a uma dada HIPERLIGACAO pertencente a um DOCUMENTO.

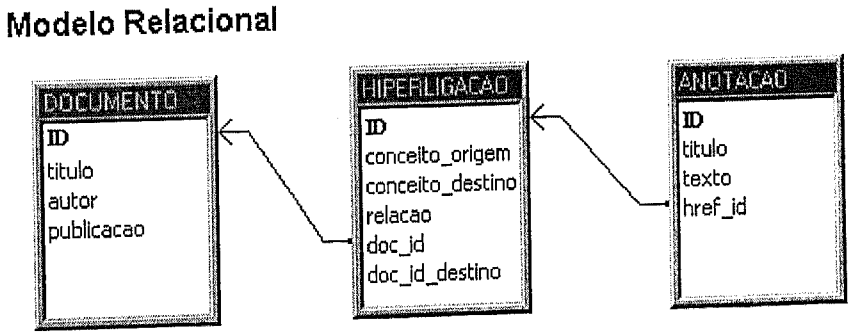


Figura 22 – Modelo relacional

O documento XML capaz de conter a informação dessas três tabelas, teria a seguinte forma:

Documento XML com duas anotações

```
<DOCUMENTO>
   <TITULO> ..... </TITULO>
   <AUTOR> ..... </AUTOR>
   <PUBLICACAO> </PUBLICACAO>
   <HIPERLIGACAO>
      <ANOTACAO>
         <TITULO> anotação 1 </TITULO>
         <TEXTO> ..... </TEXTO>
      </ANOTACAO>
      <ANOTACAO>
         <TITULO> anotação 2 </TITULO>
         <TEXTO> ..... </TEXTO>
      </ANOTACAO>
      <DOC_ID_DESTINO> ..... </DOC_ID_DESTINO>
      <CONCEITO_ORIGEM> ..... </CONCEITO_ORIGEM>
      <CONCEITO_DESTINO> ..... </CONCEITO_DESTINO>
      <RELACAO> ..... </RELACAO>
   </HIPERLIGACAO>
</DOCUMENTO>
```

Figura 23 - Documento XML com as duas anotações

Desta forma torna-se perfeitamente claro que uma base de dados de objectos está totalmente adaptada ao XML e com uma capacidade de integração muito maior, sem ser necessária a distribuição das várias partes do documento pelas tabelas.

XQL - A linguagem de interrogação das bases de dados XML

Conforme já referido, em Setembro de 1998 foi submetida uma proposta ao W3C para a especificação de uma nova linguagem de interrogação, para bases de dados XML, designada por *XML Query Language* – XQL [W3C1998a]. Pretendia-se que fosse uma extensão do XSL e que permitisse descrever critérios de pesquisa. Consequentemente, deveria permitir criar interrogações para procurar e filtrar elementos XML. Deveria proporcionar uma notação compreensível e concisa para apontar para elementos específicos e encontrar nós (elementos de uma estrutura arborescente XML) com caractirísticas particulares. Como resultado de interrogações XQL, seria obtida uma estrutura arborescente de dados XML.

A proposta da XML-QL define uma cláusula WHERE para obtenção da informação, e uma cláusula CONSTRUCT para construções de novas estruturas de dados. A especificação do XSL divide essas facilidades como *patterns* e *templates*, respectivamente.

O XQL é, para os dados XML, aquilo que o SQL é para a interrogação a bases de dados relacionais..

As folhas de estilo XSL e a linguagem de interrogação XQL

Para além da função para a qual as folhas de estilo XSL foram concebidas, estas têm também a capacidade de servir como um mecanismo básico de interrogações a documentos XML. As

linguagens de folhas de estilo XSL e as linguagens convencionais de interrogação a bases de dados têm dois pontos onde essas funcionalidades se sobrepõem: o primeiro é o facto de que ambas obtêm informação a partir de estruturas de dados extraídas da base de dados. O segundo ponto é que ambas constroem novas estruturas de informação a partir da informação obtida da base de dados. A seguinte tabela compara essas facilidades para o SQL e o XSL:

	SQL	**XSL**
Extracção de informação	SELECT - identifica os valores a extrair da base de dados	*Select patterns* – identificam os nós a processar
	WHERE – define os critérios	Os qualificadores nos padrões definem os critérios.
Construção	As interrogações permitem obter uma tabela	As folhas de estilo criam novos documentos
	ORDER BY – ordena os tuplos da tabela resultado de uma interrogação	xsl:sort - Ordena os elementos criados

Tabela 3 - Comparação SQL/XSL

Naturalmente que as funcionalidades de uma linguagem de interrogação e as folhas de estilo XSL se sobrepõem em muitos aspectos. Porém, as diferenças existem entre uma folha de estilo e uma linguagem de interrogação.

Por exemplo, uma linguagem de interrogação a bases de dados dá mais ênfase à:

- Possível complexidade das interrogações.
- Referenciar e reutilizar os resultados das interrogações intermédias.
- Capacidade de efectuar junções de resultados.
- Capacidade de obter informação a partir de múltiplos documentos.

Enquanto uma linguagem de interrogação dá ênfase a este conjunto de aspectos, as folhas de estilo normalmente dão mais importância a aspectos como:

- A capacidade de lidar com a heterogeneidade e recursividade da informação.
- O controlo dos espaços em branco.
- Marcas, numerações e outras questões relacionadas como o posicionamento.
- A capacidade de suportar múltiplos documentos de saída.

Adicionalmente a estas exigências da capacidade de transformação, uma linguagem para folhas de estilo também define um conjunto de aspectos ligados à formatação.

3.2 Publicar na Web

Desde há muito que se tem procurado obter modelos e tecnologias que sejam capazes de armazenar, tratar e disponibilizar grandes quantidades de informação, das mais diversas formas, ao maior número de utilizadores.

Sistemas centralizados de grande porte, com grandes quantidades de informação, muitas vezes com Sistemas de Gestão de Bases de Dados (SGBD) proprietários, foram o ponto de partida da evolução que se assistiu até hoje. Geralmente estes sistemas têm necessidade de transacções em tempo real (OLTP – *on-line transaction processing*), o que amplia a complexidade do problema.

A evolução destes sistemas conduziu ao aparecimento de novas arquitecturas que assentam na existência de dois tipos de intervenientes: aqueles que prestam serviços e aqueles que necessitam de serviços. Esta arquitectura, designada por Cliente/Servidor, desde logo evoluiu, para outras dela derivadas. A inclusão de outros elementos nesta arquitectura, para além do cliente e do servidor, regra geral melhora o desempenho e a robustez dos sistemas. Estas arquitecturas *n-tier*, assentam nas tecnologias distribuídas.

A explosão das capacidades de processamento e transporte de informação, associada à generalização do uso da tecnologia de objectos, leva à proliferação de arquitecturas distribuídas, tais como as tecnologias *CORBA, COM/DCOM.*

O *World Wide Web* – WWW tem também a sua responsabilidade nesta evolução, pois é este o fenómeno que torna possível que uma rede, que já há muito tinha extrapolado as fronteiras do seu pais natal e já então com uma cobertura a nível mundial, de facto, seja uma rede global e amplamente utilizada por todos. O WWW tornou-se num meio universal de comunicação, tanto ao nível da cobertura como das formas de comunicar que possibilita, pois os meios que comporta são todos aos quais um processo de digitalização possa ser aplicado.

Nesta linha de evolução surge o *Java*. Sendo o *Java* uma linguagem de extrema portabilidade e orientada ao objecto, acelera esta evolução no sentido dos objectos distribuídos. De facto a tecnologia já está em todo o lado, desde os electrodomésticos, nos sistemas de informação, do lado servidor na forma de *servlets*, e do lado cliente na forma de *applet*, e ainda como componentes de *software*

3.2.1 Protocolo HTTP para o transporte dos documentos pela *Internet*

O *Hypertext Transfer Protocol* – HTTP [TBRFHF1997], é um protocolo de nível aplicacional, sem estado, não orientado à conexão, para sistemas de informação hipermédia, distribuídos e colaborativos. O HTTP tem sido usado no *World Wide Web* desde 1990. A primeira versão do HTTP, HTTP/0.9, era um protocolo muito simples para transferência de dados em formato RAW, através da *Internet.*

Algumas melhorias foram introduzidas neste protocolo na sua versão 1.0 (HTTP/1.0) definido na RFC 1945 [TBRFHF1995]. Essas melhorias, possibilitaram o envio de mensagens num formato similar ao *Multipurpose Internet Mail Extensions* - MIME [NFNB1996], contendo meta informação sobre os dados transferidos e modificadores na semântica pergunta/resposta.

No entanto, esta versão não tomava suficientemente em consideração os efeitos da hierarquia de *proxies*, *caching*, a necessidade de conexões persistentes e computadores virtuais.

A proliferação de aplicações que implementavam este segundo protocolo de uma forma incompleta e se autodenominavam "HTTP/1.0", colocava problemas quanto às verdadeiras capacidades da aplicação.

Por esta e outras razões, surge uma nova especificação deste protocolo, o HTTP/1.1 definido na RFC 2068 [TBRFHF1997].

Mensagem HTTP

É a unidade básica de uma comunicação HTTP, consiste numa sequência estruturada de bytes enviada através de uma conexão HTTP, cujo formato está definido na RFC 822 [DC1982]. Estas mensagens contêm métodos, cabeçalhos, códigos de resultados do pedido e um corpo contendo o resultado do pedido.

Métodos

Consistem num conjunto de instruções que os clientes têm à disposição, susceptíveis de serem invocadas no servidor. Cada um destes métodos permite a execução de uma dada operação aplicada a um dado *Universal Resource Identifier* –URI [TB1994], ou informação a ser transferida (transferência de entidades). Os métodos previstos no HTTP/1.1 são os seguintes: `"OPTIONS"`, `"GET"`, `"HEAD"`, `"POST"`, `"PUT"`, `"DELETE"` e o `"TRACE"`. É ainda possível estender este conjunto de métodos, desde que o servidor em questão seja capaz de interpretar o respectivo *token*. Se o servidor não reconhecer algum método, envia como resposta o erro (*status code*) número 501 - *Not Implemented*.

Cabeçalhos

Consistem em campos que permitem a transferência de informações adicionais. Estes campos ajudam à resolução de problemas de configuração. Permitem, por exemplo, graças ao *Host Header*, que vários sítios *Web* possam estar no mesmo servidor HTTP, sem que seja necessário associar um endereço IP a cada um deles, o envio de *cookies*, mecanismos de autenticação, entre muitas outras funcionalidades.

Código de resultado

Naturalmente a execução dos métodos pelo servidor é efectuada com ou sem sucesso. A informação do resultado da execução do pedido feito pelo cliente, é enviada num cabeçalho sob a forma de códigos de resultado. Estes códigos subdividem-se em classes. Cada uma dessas classes descreve um conjunto de resultados categorizados. Desta forma, os códigos, 1xx –

Informativos; 2xx – Sucesso; 3xx – Redireccionamento; 4xx – Destinados a erros do cliente, nomeadamente erros de sintaxe ou por não poderem ser executados; 5xx – Destinados a erros do servidor, por qualquer impossibilidade de serem executados. Essa lista de códigos é extensível, dado que as aplicações HTTP não necessitam de entender todos os códigos.

Descrição do funcionamento

O protocolo HTTP é um protocolo baseado numa arquitectura cliente/servidor. Sendo assim um cliente envia um pedido ao servidor na forma de uma mensagem HTTP, contendo um método de pedido, um URI, a indicação da versão de protocolo, seguida de uma mensagem tipo MIME, contendo informação de cliente, e um possível conteúdo. O servidor interpreta esse pedido, executa-o e envia a resposta sob a forma de uma mensagem HTTP de resposta com a indicação do estado da linha, onde se inclui também a versão do protocolo da mensagem e o resultado de execução do pedido. Seguindo-se uma mensagem do tipo MIME, contendo informação do servidor, meta informação e possivelmente informação a transferir resultado da execução do pedido pelo servidor.

As características mais marcantes do protocolo HTTP são as de não ter estado nem manter conexão. O cliente faz um pedido, é executado no servidor e o resultado enviado ao cliente. Estas características colocam algumas limitações do protocolo. O desempenho é uma delas. Para transferir uma página HTML com referências a outros recursos, como imagens por exemplo, seria necessário o estabelecimento de diversas conexões, uma por cada recurso. Outra limitação é causada pela não existência de estado impossibilitando o relacionamento, ou a memória, entre os diversos pedidos, por exemplo, um pedido de autenticação, seguido de outro, relativo a um qualquer recurso. As conexões persistentes e os *cookies* são uma forma de contornar estas duas limitações do protocolo.

Conexões persistentes

Estas permitem que ligações entre clientes e servidores possam ser reutilizadas. Desta forma, reduz-se o tempo de negociação inicial e contribui-se simultaneamente para o não congestionamento da *Internet*. Pois não sendo necessário fechar e abrir de novo a ligação, reduz-se imenso o número de pacotes de fecho e abertura de ligações TCP. Isto seria ainda agravado pelo facto de que estes pacotes não estão sujeitos aos mecanismos de controlo de fluxo. Outra vantagem deste tipo de conexão, é o facto de pedidos poderem ser executados sem ter de esperar pela respectiva resposta.

Cookies

Um *Cookie* [DKLM1997] consiste num mecanismo que permite ao servidor HTTP ter "memória" sobre os diversos pedidos efectuados por um dado utilizador sobre um recurso. Este é usado para guardar informações específicas do utilizador, que mais tarde, numa conexão posterior, o permitem identificar. Desta forma, o protocolo HTTP obtém a capacidade de "recordar" conexões anteriores.

Este mecanismo é particularmente importante em aplicações *Web* com alguma complexidade, onde existem longas sequências de perguntas e respostas entre o utilizador e o serviço.

A instrução `Set-Cookie` é utilizada no cabeçalho enviado pelo servidor HTTP para o cliente, com a intenção de que este último o guarde associando-o ao recurso acedido. Mais tarde, quando o cliente voltar a estabelecer uma conexão a esse recurso, envia o *cookie* juntamente como o pedido.

Transferência de documentos

Um cliente que pretende receber um dado documento, envia uma mensagem de *request* ao servidor. A listagem que se segue exemplifica um pedido de um documento HTML, feito pelo browser *Internet Explorer 5.0*, a um servidor *Apache 1.3.3*.

```
Pedido de navegador ao servidor HTTP
------------------------------------------------------------
GET /~rhp/index.html HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.5 [en] (Win98; I)
Host: mickey.iscap.ipp.pt:8888
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
------------------------------------------------------------
```

O servidor responde, e essa resposta cai num de dois casos possíveis: sucesso ou insucesso. As duas listagens que se seguem exemplificam um caso de sucesso e outro de insucesso, respectivamente.

Quando há sucesso, o servidor envia uma série de informações, tal como já descrito, e após a indicação do tipo de conteúdo, recorrendo-se do `entity-header` e o `Content-Type`, envia o documento pretendido.

Se pelo contrário algo corre mal, já não é um código da classe 2xx, que é enviado, mas outro da classe 4xx ou classe 5xx. Neste caso, à semelhança do caso de sucesso também é enviado após o `entity-header`, o `Content-Type` e um documento HTML, contendo uma mensagem de erro. Neste caso, o documento não foi encontrado, mas outras razões poderiam ser o motivo do insucesso.

Este parâmetro, `Content-Type`, pretende informar o navegador do tipo de conteúdo enviado. Desta forma, o navegador saberá qual o tratamento apropriado a dar aos *bytes* que recebe. É só depois de receber esta informação que saberá se se trata de texto, de uma imagem ou de algum objecto que requer algum *plug-in*.

Pedido com sucesso

```
------------------------------------------------------------
HTTP/1.1 200 OK
Date: Mon, 07 Jun 1999 15:30:56 GMT
```

```
Server: Apache/1.3.3 (Unix)
Last-Modified: Wed, 28 Apr 1999 10:43:06 GMT
ETag: "9e008-204-3726e63a"
Accept-Ranges: bytes
Content-Length: 516
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML><HEAD> ...documento... </HTML>
-----------------------------------------------------------
```

Pedido com erro

```
-----------------------------------------------------------
HTTP/1.1 404 Not Found
Date: Mon, 07 Jun 1999 15:45:12 GMT
Server: Apache/1.3.3 (Unix)
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">

<HTML><HEAD><TITLE>404 Not Found</TITLE> ...mensagem de erro...</HTML>
-----------------------------------------------------------
```

O caso atrás descrito, refere-se à transferência de um documento inteiro. Neste diálogo, entre o servidor e o cliente, será de destacar o parâmetro `Accept-Ranges: bytes`. Este visa a indicação por parte do servidor de que é capaz de aceitar pedidos de *Byte Ranges* [TBRFHF1997]. Esta indicação é importante, pois permite ao navegador ter a certeza de que o servidor é capaz de aceitar tais pedidos, apesar de previamente já ter sido informado da versão do protocolo HTTP. Este procedimento permite contornar situações de implementações incompletas do protocolo HTTP.

Para a situação de transferência de apenas parte ou partes de um documento, o mecanismo de *Byte Ranges* terá de ser usado. Para corresponder a este requisito, o HTTP recorre mais uma vez às mensagens MIME.

Mas se é necessário que o corpo da mensagem contenha não apenas um documento, como atrás exemplificado, mas vários ou ainda várias partes de um documento, o HTTP prevê os `Multipart-types` e `Range requests` [TBRFHF1997].

O MIME disponibiliza um conjunto de tipos `multipart` (*multipart-types*) que encapsulam uma ou mais partes do documento a transferir num único corpo de mensagem. Trata-se de um processo idêntico ao aplicado no corpo de uma mensagem de correio electrónico composto por mais do que uma parte, por exemplo, um anexo. Todos os tipos de `multipart` têm a mesma sintaxe, definida no MIME.

Este mecanismo é muito importante, pois é ele que permite, em aplicações como o *Acrobat Reader* (ver em 3.2.2), aceder directamente a partes específicas de um documento sem ter de o transferir integralmente. O exemplo a seguir apresentado, mostra como o *Acrobat Reader*, socorrendo-se deste mecanismo do protocolo HTTP 1.1 é capaz de transferir três partes distintas de um documento PDF.

```
-----------------------------------------------------------
GET /~rhp/teste2/fullindx.pdf HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.5 [en] (Win98; I)
Range: bytes=3583842-3585035,3585036-3585583,3585584-3588234
Request-Range: bytes=3583842-3585035,3585036-3585583,3585584-3588234
Host: mickey.iscap.ipp.pt
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
-----------------------------------------------------------
```

O método GET, neste exemplo, pede o documento `fullindx.pdf`, que se encontra no directório *home* do utilizador rhp. Só que, neste caso, o cliente acrescenta ao seu pedido a indicação das faixas de *bytes* a serem transferidas, usando os cabeçalhos `Range` e `Request-Range`.

Como resposta a um pedido deste tipo, o servidor deverá enviar uma resposta do tipo:

```
-----------------------------------------------------------
HTTP/1.1 206 Partial content
Date: Wed, 16 Jun 1999 16:25:24 GMT
Last-modified: Wed, 16 Jun 1999 14:58:08 GMT
Content-type: multipart/byteranges; boundary=THIS_STRING_SEPARATES

--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 3583842-3585035/8000
...o primeiro bloco...
--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 3585036-3585583/8000
...o segundo bloco...
--THIS_STRING_SEPARATES
Content-type: application/pdf
Content-range: bytes 3585584-3588234/8000
...o terceiro bloco
--THIS_STRING_SEPARATES--
```

Esta resposta, tal como já acontecia nos casos anteriores, inicia-se com a indicação do código de resultado. No entanto, e como é natural, o código é diferente, trata-se do código 206 Partial content. Desta forma, o navegador passa a saber que se trata de uma parte de um documento. Toda a sintaxe MIME que se segue descreve as várias partes, em dimensões e tipo de conteúdo, ficando o parâmetro `boundary`, responsável pela delimitação dos vários blocos.

O `Content-type` do resultado é do tipo `multipart/byteranges`, enquanto as partes têm um `Content-type` do tipo `application/pdf`, como seria de esperar. A indicação `Content-range` informa o cliente a qual das faixas esses *bytes* dizem respeito.

Esta abordagem da transferência de documentos focou apenas pedidos de um cliente (*download*), de documentos alojados no servidor, porém, o HTTP prevê que o cliente possa depositar (*upload*) documentos no servidor.

3.2.2 Byteserving

O que é Byteserving ?

Byteserving significa enviar apenas os *bytes* pedidos, isto é, enviar apenas os *bytes* que dizem respeito à fracção necessária do documento. Com o *Byteserving* aplicado por exemplo a documentos *Adobe* PDF, torna-se possível transferir uma página do documento de cada vez, sem ter de o transferir na sua totalidade. Então, sendo possível transferir apenas a parte pretendida do documento, deixa de ser necessário esperar até que se complete a transferência do documento na sua totalidade. Como é óbvio, trata-se sem qualquer sombra de dúvida de uma funcionalidade muito útil e com enormes benefícios. Sendo estes, ao nível de conforto para o leitor, que não tem de esperar até à conclusão da transferência para poder iniciar a leitura. Dado que a transferência pode ser feita em *background*, o leitor pode ter um primeiro contacto com o texto antes da conclusão da transferência e pode interrompê-la caso não lhe interesse. Tudo isto vai permitir uma maior rentabilidade da utilização dos meios de telecomunicações utilizados, porque evita-se a transferência de informação sem interesse (a totalidade do documento ou algumas páginas), assim como permite que o leitor aproveite o tempo necessário à transferência do documento.

O *Byteserving* é também, conhecido como *page-at-a-time download* ou ainda por *byte-range download.*

Descrição do funcionamento (aplicado no *Acrobat Reader*)

Funciona de uma forma totalmente transparente, sem qualquer intervenção do utilizador. Na verdade este mecanismo assenta numa das inovações que o HTTP/1.1 nos trouxe, o *Byte-Ranges.*

O *byteserving* tem um universo de aplicações muito grande, uma delas permite optimizar a transferência de documentos PDF, através da *Internet.* A *Adobe* disponibilizou um

visualizador, o *Acrobat Reader*. Esta aplicação permite a visualização destes documentos, num modo local, isto é, abertos a partir de um ficheiro em disco, para além deste modo, na versão 3 do *Acrobat Reader*, tornou-se possível visualizar também estes documentos na janela de um navegador, como se tratasse de um documento HTML.

Lado cliente

Esta plataforma cliente WWW, consiste no recurso a um *plug-in*, baseado na tecnologia *ActiveX*. Este *plug-in* será o responsável pela interpretação dos *bytes* enviados pelo servidor e recebidos pelo navegador, tornando-se no responsável pelo diálogo cliente/servidor. Ao mesmo tempo, também é responsável pelo diálogo cliente/utilizador. Este dispositivo que entende o formato PDF, é capaz de o descodificar e exibir na janela do navegador. Ao mesmo tempo, o objecto *ActiveX* coloca uma barra de ferramentas dentro da janela do navegador para permitir o controlo do documento pelo utilizador. O diagrama que se segue ilustra este modo de funcionamento.

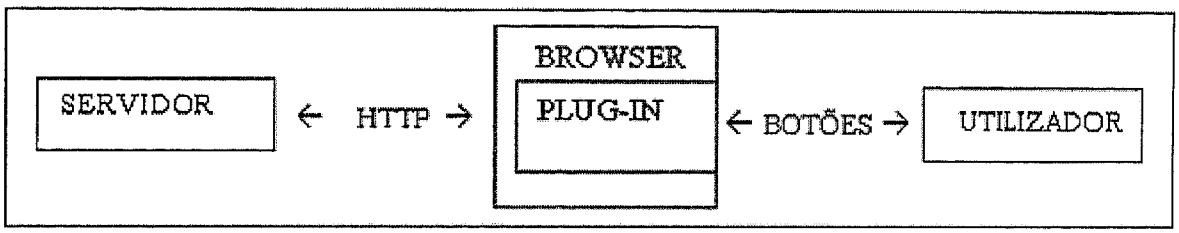


Figura 24 - Interacção do *plugin*

Lado servidor

O servidor tem de suportar o protocolo extensão do HTTP, designado por *"Byte-range Retrieval Extension to HTTP"*, documentado no *draft* de Janeiro de 1997 do HTTP/1.1, a RFC2068 [TBRFHF1997].

São exemplo de *Web Servers* que já suportam nativamente este protocolo: O *Apache*, a partir da versão 1.2, o *Netscape Enterprise Server 2.0*, o *Netscape FastTrack 2.0* e o *Internet Information Server* da Microsoft.

Nativamente, porque se o servidor não suportar este protocolo e não for possível fazer a evolução para uma versão mais recente que suporta este mecanismo, existe a possibilidade de instalar um CGI (ver em 3.2.3) que lhe confere essa funcionalidade. Existem várias versões deste *script*[1], sendo talvez a mais famosa uma versão escrita em *Perl*. Este *script* terá de estar definido como a aplicação do lado servidor que atende e interpreta os pedidos cliente, de

[1] Esse *script* está disponível em http://www.pdfzone.com

documentos específicos, por exemplo, documentos em formato PDF. A sua função é fazer aquilo que o servidor não sabe fazer. Deverá interpretar os cabeçalhos dos pedidos HTTP e extrair os *bytes* do ficheiro, e, finalmente, enviá-los para o cliente dentro de uma mensagem `multipart`.

Preparação dos documentos PDF

Mas, para que haja condições para que o ficheiro PDF possa ser servido no modo de *byteserving*, ele tem de ser previamente preparado. Essa preparação consiste numa operação designada como optimização. A optimização é feita recorrendo a outra aplicação vocacionada para o formato PDF, o *Acrobat Exchange*. Esta aplicação é capaz de editar estes documentos, e assim permite optimizar estes ficheiros de uma forma individual. O processo consiste em editar o ficheiro e fazer a operação de "Salvar Como" com a opção "Optimizar" activada. A mesma aplicação permite ainda efectuar esta operação em *batch*, dispondo para isso de uma função capaz de processar de uma só vez uma directoria contendo estes ficheiros.

Adobe Acrobat Reader

O pacote de *software Adobe Acrobat*, para lidar com documentos PDF, conta actualmente com a versão 4.0. Esta recente versão, assim como a anterior, disponibilizam ferramentas: para conversão do *PostScript* em PDF (a aplicação *Distiller*), uma aplicação de edição (*Acrobat Exchange*) e um visualizador (*Acrobat Reader*), em que este último surge em dois modos: como aplicação autónoma e como um componente *ActiveX*, passível de ser embebido numa qualquer aplicação ou navegador *Web*, utilizando o mecanismo cuja descrição de funcionamento acabou de ser efectuada.

Por outro lado, a *Adobe* fornece um conjunto de mecanismos que permitem o desenvolvimento de aplicações compatíveis com o formato PDF.

Se a versão anterior desta aplicação trouxe o *byteserving*, a versão 4.0 deste visualizador de documentos PDF introduziu um conjunto de inovações das quais a que mais se destaca é um conjunto de métodos que permitem a manipulação do documento PDF com *VBscript*, acedido através do componente *ActiveX*.

3.2.3 Extensões aos servidores *Web*

Os serviços *Internet* são baseados numa arquitectura cliente/servidor e o serviço *Web* não foge à regra. Dos agentes que intervêm num serviço como a *Web*, navegador, servidor e eventualmente um *proxy*, aquele que suporta a maior parte da complexidade do serviço é o servidor, já que o *proxy* limita-se a servir de intermediário e o cliente é um mero visualizador do que o servidor processa e envia. Aliás, é esse o princípio subjacente ao serviço *Web*, ter clientes magros.

O agente servidor consiste num dispositivo que de base tem apenas a função de servir ficheiros e execução de programas, os CGI's. Estes são aplicações geralmente implementadas

em *software*, desenvolvidas de modo a permitirem essas funcionalidades básicas, e que outras lhe possam ser adicionadas, através de extensões.

Extensões baseadas em *Common Gateway Interface* - CGI

Os programas CGI consistem no tipo de extensões mais básicas. São programas autónomos, executados ao nível do sistema operativo. A ordem de execução é dada pelo servidor *Web*, que lhe passa um conjunto de parâmetros (via método GET ou POST, ver em 3.2.1). Esses parâmetros são a "matéria prima" do CGI. O CGI envia o resultado da sua execução, para o seu dispositivo de saída (o *standard output*). O servidor recolhe esse resultado, o documento HTML ou outro tipo qualquer, que de imediato o envia para o cliente, via protocolo HTTP.

Esta foi a primeira técnica utilizada no desenvolvimento de extensões para servidores *Web* e é ainda muito utilizada. São programas escritos numa qualquer linguagem de programação.

A execução de um programa num sistema operativo dá origem a um processo. Em situações de sobrecarga, com um elevado número de pedidos a serem submetidos ao servidor, o número de processos naturalmente torna-se elevado, com consequências graves em termos de esgotamento dos recursos de memória e CPU.

O tempo de criação do processo é outra limitação desta técnica, que em situações de congestionamento aumenta drasticamente, pelo facto dos recursos de memória e CPU, estarem perto do limiar com comutações de páginas de memória virtual frequentes.

Por outro lado, sendo cada um dos processos um mecanismo autónomo, a partilha de informação e recursos é efectuada recorrendo a mecanismos do próprio sistema operativo, os *Interprocess Communications* – IPC[1]. Estes são mecanismos dependentes do sistema operativo o que limita bastante a portabilidade das aplicações.

A evolução conduziu ao aparecimento de novas técnicas, ambientes e linguagens, vocacionadas para este tipo de aplicações. Essas linguagens fornecem bibliotecas de funções e procedimentos em ambientes integrados, com grandes vantagens no que toca à reutilização de código. Desta forma, é possível a partilha de um único processo (ou um número limitado) do sistema operativo. Caso sejam ambientes orientados aos objectos, quando em funcionamento, muito provavelmente esses objectos já estão instanciados, além do tempo da criação de um novo processo, o tempo que demora a instanciar esse objecto pode ser evitado. O PHP[2], *scripts* em *Perl*[3], *Active Server Pages* – ASP[4], *Java Server Pages* – JSP e *Servlets* são alguns exemplos destas técnicas.

Extensões baseadas em *Servlets*

Uma aplicação desenvolvida em *Java* para a sua execução necessita de um ambiente especial, uma máquina virtual [TLFY1999]. Esta consiste num programa executado sobre o sistema operativo, que interpreta o código binário *Java*, o *bytecode* [TLFY1999].

---

[1] Mecanismos de semáforos, memória partilhada, mensagens, sinais, etc..

[2] Mais informação em http://www.php.net

[3] Mais informação em http://www.perl.com

[4] Mais informação em http://www.microsoft.com

A linguagem de programação *Java*, é orientada por objectos e faz uma abstracção total do *hardware*, o que permite a total independência dos programas desenvolvidos nesta linguagem relativamente ao sistema operativo. Por outro lado, o acesso directo à memória não é possível (caso dos apontadores do C e C++). Assim, elimina-se a possibilidade de corrupção das estruturas de dados, permitindo que as aplicações desenvolvidas nesta linguagem sejam mais isentas de problemas e consequentemente mais robustas. Outra facilidade que o ambiente de execução virtual proporciona é a *Garbage Collection* [TLFY1999]. Esta faz uma gestão automática da memória, libertando o programador das preocupações relativas à destruição dos objectos que já não são necessários e ocupam espaço em memória. Porém, existe já desenvolvido um interface, o *Java Native Interface* – JNI [TLFY1999], que permite embeber, na aplicação *Java*, código nativo para a plataforma onde a aplicação é executada. Além deste interface, existem inúmeras API's que facilitam extraordinariamente o desenvolvimento de aplicações. São alguns exemplos, os mais relevantes para a presente dissertação, as seguintes API's e produtos: o JDBC [GHRCMF1997] para a interacção com bases de dados; JAXP [JDD2000] uma API para manipular estruturas de dados XML; *Coccon*[1] um produto livre desenvolvido pelo grupo *Apache* para o desenvolvimento de aplicações *Java* com suporte XML; o *JavaMail* [SUN1998], para a interacção com agentes de correio electrónico; e muitas outras, cuja referência seria exaustiva.

Os *servlets* são aplicações desenvolvidas em *Java*. O seu desenvolvimento é baseado numa API, o *Java Servlet* [DRC1999]. Esta API para ser utilizada requer o kit de desenvolvimento *Java Servlet Development Kit* – JSDK. Uma aplicação *servlet* (`MeuServlet` na Figura 25) é uma extensão à classe `HttpServlet`, a qual já tem os métodos HTTP pré definidos (exemplo: *doGet*() e *doPost*()), que devem ser redefinidos no sentido da atribuir o comportamento desejado ao *servlet*, para cada um desses métodos. A classe `HttpServlet` é uma extensão à classe `GenericServlet`, que a especializa para o protocolo HTTP, pois um *servlet* pode utilizar outros protocolos, como o FTP [JPJR1985], SMTP [JP1982], etc.. A Figura 25 ilustra a relação entre estas classes.

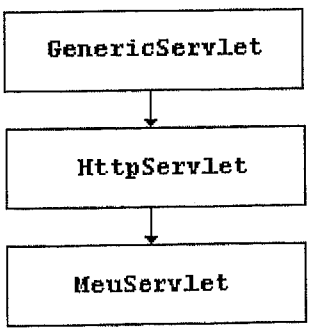


Figura 25 – Relação de especialização

Um *servlet* é um programa que depois de escrito pelo programador tem de ser compilado, o mesmo não acontece com as *JavaServer Pages* - JSP. Estas são páginas *Web*, com código *Java* embebido, compiladas na primeira vez que são pedidas ao servidor. Baseiam-se na API

[1] Mais informação em http://xml.apache.org

JSP [EPLC1999], uma extensão aos *Servlets*, e requerem para o seu desenvolvimento o *JavaServer Web Development Kit* – JSWDK.

Um *servlet* só é instanciado quando é necessário, portanto até lá não ocupa recursos. Quando é necessário, o método o `init()`, pré definido mas que naturalmente pode ser alterado, permite a inicialização de todas a variáveis do programa, e até o estabelecimento das ligações a bases de dados, se for o caso. O *servlet* é destruído pelo servidor quando não for necessário. Enquanto estiver instanciado, os seus métodos podem ser invocados inúmeras vezes.

Breve comparação de tecnologias

Qualquer uma das soluções atrás apresentadas, baseadas em *Java*, permite o desenvolvimento de aplicações independentes da plataforma de execução. Isto constitui uma grande vantagem comparativamente com outras soluções, como por exemplo os ASP's, que estão limitados ao ambiente *MS-Windows*.

Relativamente aos CGI's, os *servlets* apresentam duas vantagens muito importantes. A primeira reside no facto de que um *servlet* é um objecto que, depois de instanciado, os seus métodos podem ser executados inúmeras vezes, com as vantagens já referidas. A segunda advém do facto de existirem inúmeras API's e produtos, que facilitam extraordinariamente o desenvolvimento.

Um *servlet* é uma aplicação *Java*, cujo código binário não é nativo da plataforma de execução. Sendo isso uma vantagem, poderá no entanto constituir uma desvantagem relativamente às aplicações nativas da plataforma, os CGI's por exemplo. Se o tradutor que faz a tradução do pseudo código para o código nativo, o compilador JIT ("*just-in-time*"), não for suficientemente eficaz, as perdas de performance durante a execução das aplicações poderão ser muito significativas. Sendo necessária a compilação, torna a execução do *servlet* bastante mais eficiente que as linguagens *script*, como por exemplo o PHP e o *Perl*, que são interpretados no momento.

3.2.4 Segurança dos documentos

Uma abordagem da segurança dos documentos e da informação que comportam, tem de ser efectuada tendo em conta várias perspectivas: A segurança de informação, isto é, privacidade e corrupção; A autenticidade da origem do documento (assinaturas digitais); E quais os riscos que os utilizadores correm ao abrir um documento. Relativamente à segurança da informação, esta pode ainda ser colocada de três formas: segurança do formato, a segurança das transacções e a autenticação no acesso aos documentos.

Segurança do formato

A segurança do formato tem implicações ao nível da privacidade e corrupção da informação contida no documento. Medidas deste género devem limitar o acesso à informação, para leitura e edição.

As linguagens abordadas baseadas em marcadores, o HTML e o XML, não tem nenhum mecanismo que nativamente permita a protecção da informação. Estas são normas totalmente abertas, que em caso de ser necessário proteger a informação que contêm terão de ser aplicadas técnicas de encriptação externas ao formato.

Relativamente à garantia de não corrupção da informação, existe um grupo de trabalho[1] no W3C, que tem como missão criar para o XML uma sintaxe que possa ser usada na representação de assinaturas de recursos *Web*, protocolos de transferências de mensagens e procedimentos, para processar e verificar essas assinaturas [JR2000].

O formato de descrição de página, PDF, desde o aparecimento do *Acrobat 2.x*, já permite proteger a informação, por criptografia. Esta criptografia faz parte do formato. Para o utilizador este processo é bastante simples. Na aplicação *Exchange*[2] é possível definir se o ficheiro é guardado, com ou sem encriptação, e se o acesso deverá ficar limitado através de uma palavra chave. Esta aplicação permite ainda a definição de outras directrizes de segurança, relativas a determinadas acções que podem ou não ser efectuadas:

- Impressão – Um documento pode ser visualizado, mas não permitida a impressão
- Alterações ao documento – O *Exchange* é uma ferramenta de edição e activando a inibição de edição, o documento PDF só pode ser aberto para leitura. Os campos dos formulários podem também ser inibidos de ser preenchidos.
- Selecção de texto e gráficos – Uma forma de inibir a exportação de informação do documento, por "*Copy & Paste*".
- Adição e alteração de anotações – As anotações de um documento podem ser protegidas.

Mas o uso da palavra chave não é obrigatório, isto é, o criador de um documento pode encriptar um documento e não defini-la. Neste caso, o documento pode ser aberto, apenas as restrições ao documento são protegidas por uma segunda palavra chave.

A encriptação do PDF é baseada no algoritmo RC4, desenvolvido por Ron Rivest. Para além deste algoritmo, o MD5 [RR1992] também é utilizado. Ambos são utilizados em muitas outras aplicações. Dadas as políticas restritivas dos produtos de criptografia nos Estados Unidos, o algoritmo RC4 pode apenas ser exportado com chaves de comprimento até 40 bits. Por esta razão, o *Acrobat* e muitas outras aplicações como o *Netscape Navigator*, usam chaves com 40 bits.

A segurança oferecida pela encriptação de um documento PDF não é a melhor. Num concurso da empresa RSA, onde se pretendia testar o grau de protecção de algoritmos de encriptação, um algoritmo RC5 com uma chave de 40 bits foi quebrado em 3,5 horas. Para uma chave de 48 bits já foram necessários 313 horas. Os algoritmos RC4 e RC5 não são directamente comparáveis, contudo, este facto não pode ser ignorado. A protecção do *Netscape Navigator* com uma chave de 40 bits também foi quebrada. Depois destas duas experiências, não se pode garantir uma segurança total para um documento PDF.

[1] Mais informação a partir de http://www.w3c.org/Signature.

[2] O Exchange permite a edição de documentos PDF.

Relativamente aos documentos PDF encriptados, existem duas observações a fazer. A primeira é o facto de que a aplicação *Distiller*[1] não permitir a encriptação dos documentos PDF, essa encriptação tem de ser feita a posteriori, recorrendo ao *Exchange*. Outra observação importante, são as implicações no processo de indexação dos documentos. Se um documento PDF estiver encriptado, o *Catalog*[2] não é capaz de o indexar, por outro lado se a encriptação for aplicada posteriormente à catalogação os mecanismos de pesquisa não funcionam correctamente, pois o documento foi alterado. Este ciclo vicioso pode ser quebrado, se os documentos forem optimizados, aplicando-lhes o processo de optimização[3]. Esta operação trás ainda outros benefícios e naturalmente continua a ser necessária a palavra chave para abrir os documentos.

Segurança das transacções

Se o documento for auto suficiente em termos de segurança do formato, a segurança da sua transferência entre dois sistemas através de uma rede insegura é pouco crítica, pois, mesmo que o documento seja interceptado[4], este dificilmente será descodificado. Agora se não existem mecanismos intrínsecos ao documento para protecção dos dados, como acontece no HTML, é necessária a implementação de mecanismos externos, que protejam a informação transaccionada através da rede de computadores.

Na *Internet*, em particular na *Web*, a técnica mais habitual consiste na utilização do protocolo SSL - *Secure Sockets Layer* [AFPKPK1996]. Este foi desenvolvido pela *Netscape* para o estabelecimento de conexões seguras. A utilização desta técnica, consiste em acrescentar uma camada protocolar entre a pilha TCP/IP e o protocolo aplicacional, por exemplo, o protocolo HTTP. A conjugação do HTTP com o SSL, designada por HTTPS, permite que a informação transaccionada entre o servidor e o cliente (o navegador) possa estar protegida.

O protocolo SSL permite a mútua autenticação do servidor e navegador e usa assinaturas digitais para salvaguardar a integridade da informação e a encriptação para a privacidade.

Este protocolo suporta a escolha de um conjunto de algoritmos de criptografia, autenticação e assinaturas. Esta possibilidade de escolha, do algoritmo, permite tirar proveito de novos algoritmos e que a escolha possa ser efectuada mediante as regras de utilização da criptografia do país. Esta e outras definições são negociadas no início da ligação, entre o servidor e o cliente.

No início de uma sessão SSL existe uma fase de negociação. Esta fase recorre a três protocolos: SSL *Handshake Protocol*, para o estabelecimento da sessão entre o servidor e cliente; SSL *Change Cipher Spec Protocol,* para a especificação do algoritmo de criptografia a utilizar; e o SSL *Alert Protocol*, para o transporte das mensagens de erro entre o servidor e o cliente. Estes protocolos, assim como a informação aplicacional, são encapsulados num outro protocolo, o SSL *Record Protocol.* A Figura 26, esquematiza a interacção entre o SSL e os restantes protocolos da pilha protocolar.

[1] O *Distiller* permite a transformação de documentos *PostScript* em PDF.

[2] O *Catalog* permite a catalogação de documentos PDF.

[3] Processo descrito em 3.2.2

[4] São perfeitamente conhecidos os riscos corridos quando, numa rede local com topologia em barramento, alguém coloca um terminal, com o interface de rede em modo promíscuo.

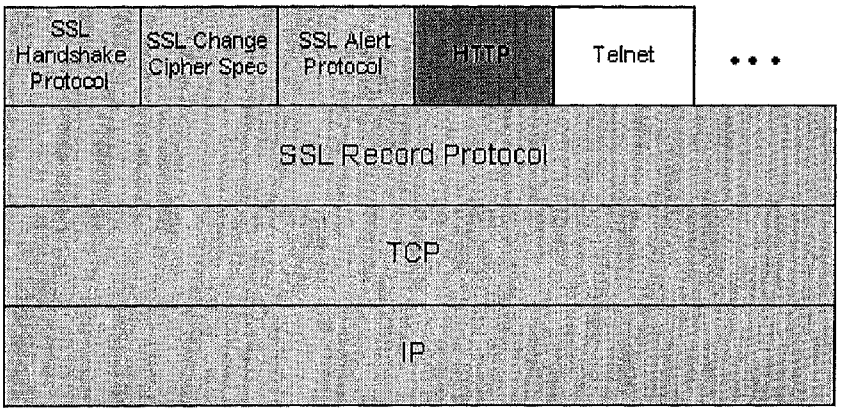


Figura 26 - Pilha protocolar TCP/IP com SSL

Para os utilizadores o uso desta camada adicional de *software*, tem poucas implicações. Os endereços http://máquina.domínio passam a ter indicação deste protocolo, https://máquina.domínio. Outra consequência, esta com implicações nulas para os utilizadores, é utilização de outra porta TCP, a porta 443, quando o HTTP habitualmente usa a porta 80.

Este protocolo actualmente conta com as seguintes versões: O SSL 2.0 [KEB1995], a primeira versão implementada, o SSL 3.0 [AFPKPK1996] e actualmente em desenvolvimento pela *Internet Engineering Task Force* – IETF, o *Transport Layer Security* - TLS [TDCA1997], baseado no SSL versão 3.0. Os servidores *Web*, como o *Apache*[1] e MS - *Internet Information Server*,[2] utilizam este mecanismo de segurança das transacções. O mesmo acontece com a generalidade dos navegadores *Web*.

Um protocolo extensão do HTTP, o S-HTTP (Secure HTTP), pode também ser utilizado na transmissão de informação de uma forma segura, através de uma rede de computadores. Enquanto que o protocolo SSL estabelece uma conexão segura entre o servidor e o cliente, na qual, qualquer quantidade de informação pode ser transferida, o S-HTTP baseia-se na transferência segura de mensagens individuais. O S-HTTP é bastante menos popular que o SSL e não é suportado pelo mesmo número de navegadores. Este protocolo também foi submetido ao IETF para a aprovação como uma norma de *Internet*.

Autenticidade e integridade dos documentos

Para o envio de uma mensagem através de uma rede, mesmo que esteja encriptada, é necessário que no destino, quem a recepciona, tenha certeza da autenticidade do conteúdo e que de facto a origem é correcta. Como foi anteriormente descrito o protocolo SSL já implementa esse tipo de segurança com assinaturas digitais, no entanto, as aplicações podem implementar as suas próprias medidas de segurança, que podem ser as mais diversas. Uma

[1] http://www.apache.org
[2] http://www.microsoft.com

delas consiste na encriptação da informação existente no documento com a obrigatoriedade de uma palavra chave para a desencriptação.

Autenticação no acesso aos documentos

Outra medida com vista à protecção da informação contida em documentos, consiste no controlo do acesso, baseada em mecanismos de autenticação. O protocolo SSL também já implementa mecanismos deste género, os quais permitem a autenticação mútua dos sistemas servidor e cliente. Mas a autenticação do utilizador é outro nível de autenticação, extremamente importante para garantir da privacidade da informação dentro de um grupo de utilizadores de dado sistema.

A autenticação do utilizador é geralmente efectuada ao nível aplicacional. O protocolo HTTP prevê mecanismos de autenticação de utilizadores, implementados na generalidade dos servidores *Web*. O uso deste meio de autenticação é bastante eficaz e simples, para recursos estáticos, documentos HTML, PDF, etc., dado que a protecção é efectuada pelo próprio servidor HTTP, ao nível do directório. O funcionamento deste método de autenticação, implementado no HTTP, pode basear-se num de dois mecanismos designados por: *Basic Authentication* [TBRFHF1997] e *Digest Authentication* [JFPHJH1997].

O primeiro, usa uma base de dados de utilizadores, contendo para cada um deles um par de valores *username/password* associado a um *Basic realm* [TBRFHF1997], que identifica o recurso. Este mecanismo está amplamente implementado na generalidade dos servidores e clientes.

O segundo mecanismo é mais elaborado, baseia-se no MD5 [RR1992], uma função *hash* criptográfica aplicada à palavra chave. Esta técnica tem o inconveniente de existirem várias palavras chave que conduzem ao mesmo resultado da função *hash*. Embora sendo difícil conseguir duas palavras chave que tenham o mesmo resultado, no cálculo da função, são acrescentadas outras informações ao argumento da função, de modo a reduzir a probabilidade de surgir uma situação desse tipo. Essa função tem o seguinte aspecto:

```
MD5(MD5(<palavra chave>)+":"+<nº inventado na ocasião>+":"+MD5(<método>+":"+<uri>))
```

É o resultado desta função *hash* que é enviado através da rede, o que constitui uma medida adicional de segurança ao mecanismo *Basic*, visto que não obriga o envio da palavra chave. No outro extremo a mesma função é aplicada à palavra chave e são esses dois resultados que são comparados. São muito poucos os agentes que aplicam este mecanismo de autenticação, um deles é o servidor *Apache*.

Para recursos dinâmicos, aplicações extensão do servidor HTTP: CGI's, *Servlets* ou ASP's, poderão ser as próprias aplicações a implementar este tipo de mecanismos. Esta técnica tem a vantagem de permitir uma maior facilidade de gestão dos utilizadores e categorias de acesso, visto que não obriga a alterações de configuração do servidor HTTP. Por outro lado, obtém-se uma política de acessos bastante mais integrada e flexível com a aplicação.

Riscos que os utilizadores correm ao abrir os documentos

As possibilidades de ataques maliciosos através da *Internet* a um computador, de um utilizador *Web*, têm de ser estudadas conjuntamente com as técnicas de tornar a *Web* mais dinâmica. Porque proteger demasiado os sistemas implica perdas de dinamismo.

Abrir uma página HTML não implica qualquer risco para o computador. O mesmo não se pode dizer para um documento *PostScript*, PDF, ou ainda se essa página contiver qualquer código *VBscript*. Os objectos neles embebidos, e programas por eles executados, esses sim podem implicar graves riscos. Uma página HTML pode conter diversos objectos, e desse conjuntos de possíveis objectos serão abordados os *Applets* e o *ActiveX*.

Quanto aos documentos PDF, estes permitem a execução de programas. Desta forma, carregar um documento PDF desconhecido e abri-lo, representa um risco de este documento colocar em execução um programa externo com intenções maliciosas. Atenta a este problema a *Adobe*, na sua versão 3.01 do *Acrobat*, implementou uma caixa de diálogo que avisa o utilizador assim que um programa qualquer tem de ser executado. Esta caixa de dialogo pergunta ao utilizador se o programa deverá ou não ser executado.

Contudo, este risco não se compara aos riscos colocados pelo *ActiveX*. Um programa VBscript embebido numa simples página HTML, pode dar ao atacante completo acesso ao computador, via objectos *ActiveX*.

O *ActiveX* é suportado por sistemas *Windows*, até ao momento não existem outros sistemas que o suportem, embora isso já esteja previsto. Este facto torna os sistemas *Windows* os únicos vulneráveis a este tipo de ataque.

A tecnologia *ActiveX* é baseada em técnicas *Windows* OLE (*Object Linking and Embedding*) e COM (*Component Object Model*). Está muito bem integrada com o sistema operativo *Windows*, o que naturalmente trás inúmeras vantagens. O *ActiveX* não é uma linguagem de programação, mas uma especificação para módulos de *software* que podem ser escritos em qualquer linguagem como o C/C++, *Visual Basic* ou *Java*. Contrariamente aos *Applets*, os controlos *ActiveX* (módulos OCX), têm controlo total sobre o sistema, com todas as vantagens obvias. Mas por outro lado, esta grande capacidade de controlo sobre o sistema do utilizador pode ser utilizada de uma forma maliciosa. Um componente de *software ActiveX*, pode apagar ficheiros, ter acesso a informações confidenciais, etc.. O *ActiveX* suporta um mecanismo de certificados de segurança digitais que o podem certificar. Porém, se os fabricantes podem certificar um *ActiveX*, os atacantes também o podem fazer.

Uma das propriedades do *ActiveX* que o torna flexível, mas por outro lado reduz ainda mais a segurança, é o caso de numa página HTML existir uma referência a um objecto *ActiveX* que não está instalado no computador. Neste caso o navegador pode carregá-lo através da *Internet* sem a intervenção do utilizador. Para prevenir este tipo de acção, a única forma efectiva de garantir a protecção, que nenhum módulo é instalado, consiste em desactivar o carregamento de módulos OCX e a sua execução. É claro que desta forma prejudica-se o potencial de certas aplicações, se não mesmo a sua utilidade.

Os *applets* em questões de segurança já oferecem mais garantias para os utilizadores da *Web*. Um *applet* permite a execução de um amplo conjunto de acções. Mas, no que toca a acções que necessitem do acesso ao disco local, estas não são permitidas. A máquina virtual, o ambiente de execução do *applet*, não permite esse, assim como outros tipos de acções que comprometam a segurança do computador que o executa.

Capítulo 4 - Desenvolvimento de um protótipo do serviço proposto

Neste capítulo apresenta-se um protótipo que implementa as ideias essenciais do modelo de um serviço de revisão bibliográfica colaborativa, descrito no segundo capítulo. No seu nível mais básico constitui um interface de acesso a um repositório de documentos em formato electrónico. Assentes neste nível existem dois outros, vocacionados para o registo de informação acrescentada pelos utilizadores do serviço.

4.1 Objectivos do serviço

Com este protótipo, pretende-se fornecer aos seus utilizadores uma ferramenta de trabalho que os ajude a minimizar alguns dos desafios (ver na página 42) que actualmente encontram ao manipular toda a informação que têm à disposição. Pretende-se acima de tudo que os utilizadores tenham à sua disposição não só a informação primária (primeiro nível do modelo: os documentos e a respectiva catalogação), mas também ligações entre a informação, meta dados e o apoio da comunidade de interesses, graças ao espírito de colaboração mútua entre utilizadores.

O serviço constitui uma ferramenta de apoio ao trabalho de investigação, tanto individual como colaborativo. Os utilizadores do serviço, no seu trabalho de investigação, podem utilizar o serviço como ferramenta de anotação pessoal que, de uma forma natural, no decorrer do trabalho, vai gerando a informação que é acrescentada aos documentos que constituem a base do sistema, informação dos segundo e terceiro níveis. Toda a informação produzida por um utilizador no seu trabalho de revisão pode ser disponibilizada à comunidade de utilizadores do serviço, caso este o autorize, ou ser utilizada apenas para fins pessoais.

4.2 Descrição da arquitectura do serviço

Para a implementação do serviço foi utilizada como base uma arquitectura capaz de corresponder às necessidades do modelo proposto e simultaneamente também capaz de adaptar-se a qualquer meio aplicacional envolvente, em particular o sistema de catalogação e o sistema de autenticação externo. Esta é uma arquitectura conceptual que esquematiza o relacionamento entre os diversos elementos lógicos que a compõem. Segmenta o sistema em três subsistemas, cada um deles com a responsabilidade de dar resposta a um conjunto específico de funcionalidades necessárias à manipulação dos três níveis de informação referidos no modelo proposto. Não existe qualquer relacionamento directo entre os três níveis de informação com os três subsistemas. Estes são responsáveis por funcionalidades que interceptam ortogonalmente os três níveis de informação, tendo esta divisão em vista apenas questões operacionais.

Para que não existam ambiguidades na descrição efectuada a partir daqui, fica desde já definido o seguinte: sistema é o conjunto dos três subsistemas; subsistema é um dos três subsistemas da arquitectura conceptual; sistema de catalogação é um dos elementos (físicos ou lógico dependendo do contexto) do subsistema servidor; protótipo é o mecanismo que foi desenvolvido de modo a permitir a articulação entre os três subsistemas, faz parte do subsistema servidor.

A Figura 27 esquematiza a referida arquitectura.

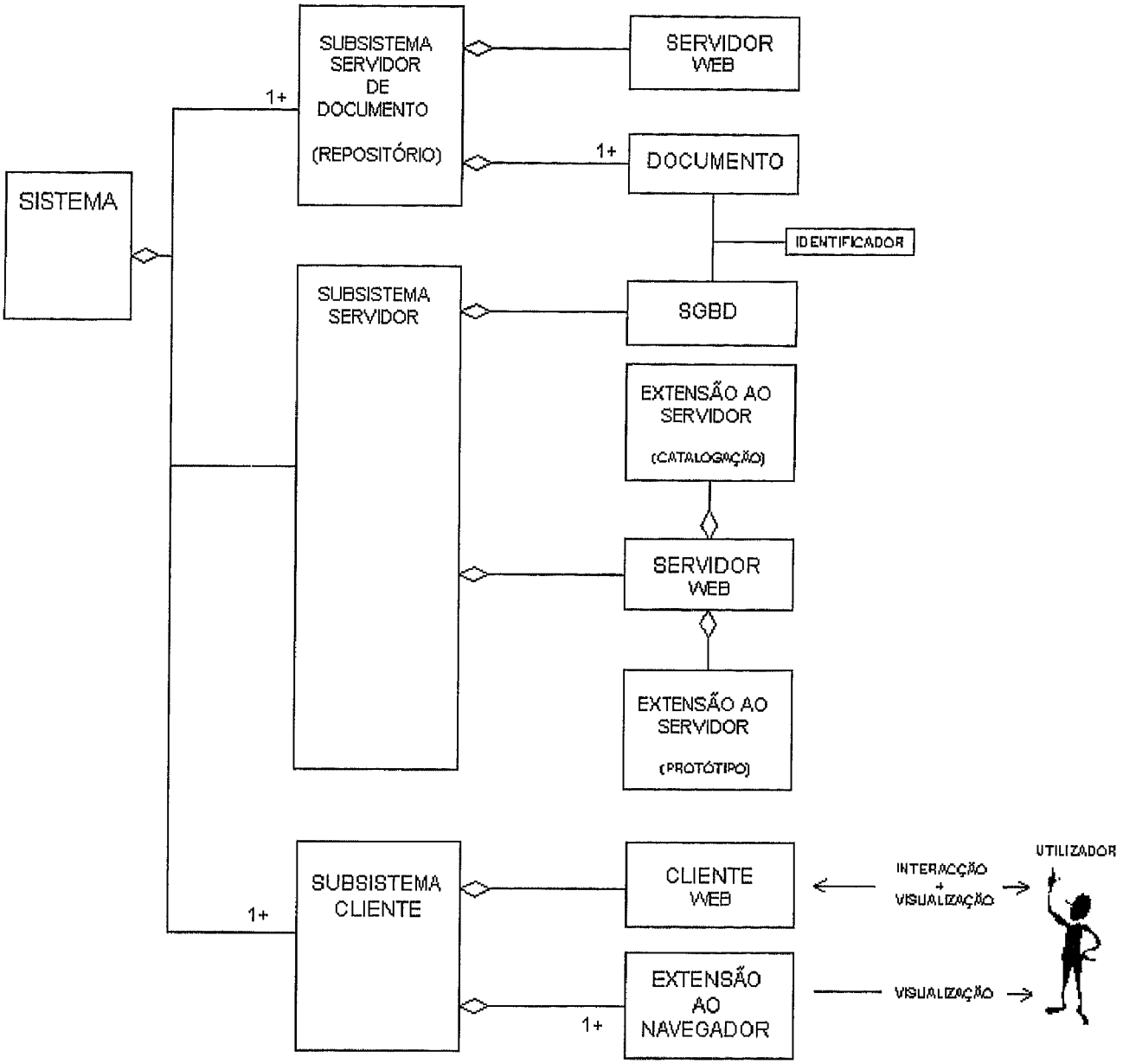


Figura 27 - Diagrama conceptual do sistema

4.2.1 Subsistema repositório de documentos

O subsistema repositório é o elemento do sistema que guarda e fornece os documentos. Este agrega os documentos e um dispositivo que os envia directamente ao cliente. Para esse envio deverão ser utilizados protocolos de rede, como por exemplo o HTTP e FTP.

Existe um relacionamento lógico, entre cada um dos documentos e um registo bibliográfico no SGBD do subsistema servidor. Este relacionamento é caracterizado por um identificador, o qual permite identificar o documento univocamente, esteja ele onde estiver, porque este subsistema pode estar distribuído.

O cliente quando pretende um documento já deve ter o conhecimento da sua URL. Assim, o acesso aos documentos é efectuado de uma forma totalmente transparente para o cliente.

4.2.2 Subsistema servidor

O subsistema servidor deverá guardar, manipular e fornecer toda a informação associada aos documentos, sendo a localização destes a informação mais relevante. No SGBD, estão os meta dados de catalogação dos documentos espalhados pelos diversos repositórios. Além destes meta dados, o SGBD contém as hiperligações, os meta dados das hiperligações (anotações objectivas) e as anotações (subjectivas).

O elemento servidor deste subsistema é aquele que funciona como interface entre as suas extensões (catalogação e o protótipo desenvolvido) e os clientes (o navegador).

4.2.3 Subsistema cliente

O subsistema cliente tem apenas a função de interface entre o utilizador e o resto do sistema. Este subsistema interage com o subsistema servidor, para acesso à informação contida no SGBD e com o subsistema repositório, para o acesso aos documentos.

Caso seja necessário, este deverá ter uma extensão por cada formato de documentos que o navegador nativamente não suporte.

4.3 Descrição do serviço implementado

4.3.1 Tipo de documentos

O modelo que serviu de base a este serviço deixa em aberto o tipo de documentos existentes no repositório. Contudo, foi estruturado com vista ao desenvolvimento de serviços de apoio à investigação, mais concretamente de revisão bibliográfica colaborativa. Por isso, pressupõe que a maioria dos documentos do repositório são artigos científicos.

Os formatos mais usuais em artigos científicos são o *LaTeX*, *PostScript*, PDF e *MS-Word*. O uso do formato PDF neste tipo de documentos tem vindo a crescer [SH1998a]. O facto da maior parte do material disponível para construir o repositório estar no formato PDF, torna a compatibilidade do protótipo com este formato obrigatória. Para efeitos de teste do modelo, considerou-se suficiente o suporte deste formato.

4.3.2 Arquitectura funcional do sistema implementado

O diagrama conceptual da Figura 27 mostra como os diversos componentes lógicos se relacionam. Cada um dos objectos desse diagrama tem uma função muito específica. De modo a ajustar essas funções ao cenário onde o sistema foi implementado, procedeu-se a uma reorganização dessas funções. A integração com um sistema de catalogação fez com que elementos como o SGBD e o servidor *Web*, do subsistema servidor, fossem separados.

O diagrama da Figura 28 ilustra o relacionamento entre os componentes físicos, o fluxo de informação e protocolos utilizados.

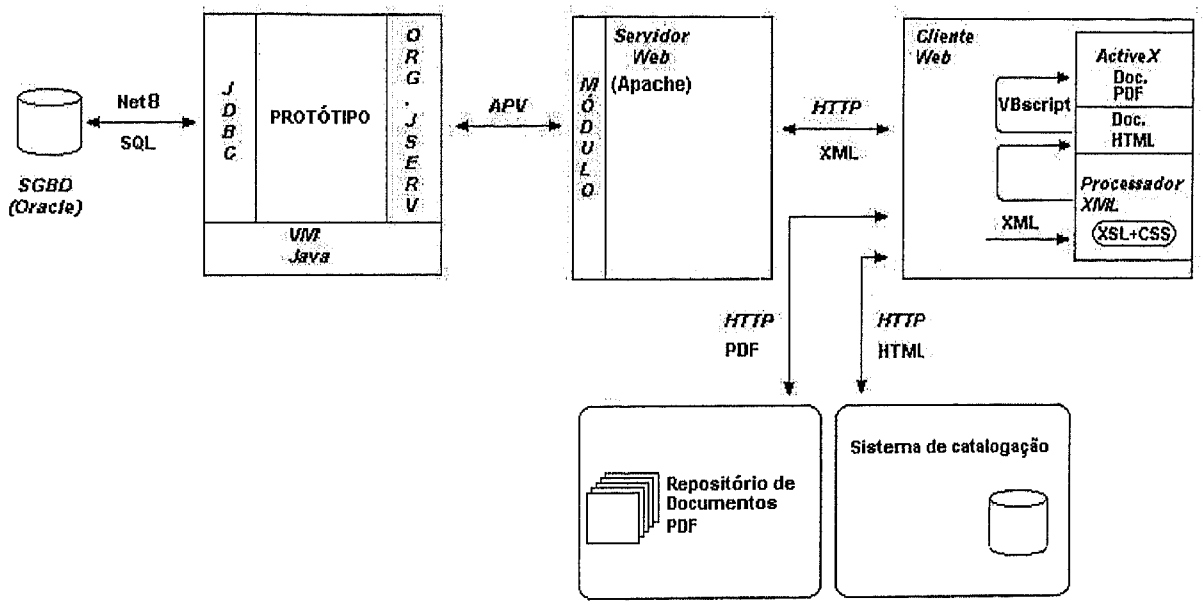


Figura 28 - Diagrama funcional do sistema implementado

O bloco "Repositório de Documentos PDF" implementa integralmente o subsistema repositório. Usa um sistema de ficheiros para alojar os ficheiros PDF e um servidor HTTP para o envio desses documentos através da *Internet*. A Figura 28 ilustra apenas um repositório, mas podem ser vários.

É o bloco "Sistema de catalogação" o elemento responsável pela agregação lógica dos diversos repositórios que constituem o subsistema repositório. Este bloco é totalmente implementado à custa de dois serviços fornecidos pela biblioteca, o sistema de catalogação

Aleph[1] e o sistema de indexação para pesquisa em texto integral[2]. Esta integração será descrita em pormenor mais à frente (ver em 4.3.4).

O bloco central desta implementação é o protótipo desenvolvido. Consiste num conjunto de classes *Java*, que implementam o segundo e terceiro níveis do modelo. Este elemento é uma extensão ao servidor HTTP. A sua arquitectura interna, funcionamento e as interfaces com o SGBD e com o servidor HTTP, serão descritos na próxima secção.

O bloco "Servidor *Web*" é constituído por um servidor HTTP, o *Apache* (versão *1.3.9*), mais um módulo, o *Apache-Jserv* (versão 1.1), este também uma extensão ao servidor *Apache*. Este módulo tem como função servir de interface entre o servidor HTTP e a máquina virtual *Java*, o ambiente do protótipo.

O SGBD é outro elemento do sistema. Pertence ao subsistema servidor e presta serviços ao sistema de catalogação, assim como ao protótipo, ambos, extensões ao servidor HTTP. Na implementação efectuada, este SGBD é constituído por dois SGBD's distintos. O primeiro é responsável pelo arquivo de toda a informação de catalogação. A sua estrutura está intimamente ligada ao sistema de catalogação *Aleph*. O segundo SGBD guarda toda a informação que diz respeito ao protótipo desenvolvido em *Java*. O *Microsoft Index Server*, responsável pela indexação e pesquisa em texto integral dos documentos, guarda a sua informação em ficheiros específicos, os quais não poderão ser considerados como um SGBD.

4.3.3 Protótipo da extensão ao servidor HTTP

O protótipo desenvolvido consiste num conjunto de classes *Java*. Cada uma destas classes tem o objectivo de corresponder a um conjunto de necessidades.

Estando o diálogo entre os subsistemas cliente e servidor assente num protocolo como o HTTP, sem estado nem sessão, torna-se imperativo que algum destes dois intervenientes guarde o estado do sistema. A opção tomada foi guardar essa informação de estado no servidor, visto que fazê-lo no cliente aumentaria a sua complexidade e tornaria a segurança do sistema mais débil. Obviamente, esta opção representa carga para o servidor. Porém, para a plataforma utilizada, a máquina virtual *Java*, essa informação de estado não é significativa visto que são necessários apenas alguns *bytes* para representar o estado de cada cliente. Para esse efeito, foi construída uma classe especialmente adaptada para guardar e manipular as variáveis de estado. Deu-se o nome `Sessao` a essa classe. Como esta tem de ser persistente, é agregada a uma classe `Session` definida na API *Java Servlet* (ver em 3.2.3). Esta classe `Session` é instanciada sempre que surge um novo cliente, passando-lhe um *cookie* (ver em 3.2.1) que guarda a referência que identifica a classe `Session` no servidor. A classe `Session`, fornece dois métodos: `putValue()` e `getValue()`, estes servem para colocar e obter objectos (propriedades) dentro do espaço do objecto instanciado, e persistente, a classe `Session`.

[1] Mais informações em http://www.aleph.com.il

[2] Constituído por um servidor *Web*, o *MS Internet Information Server*, um mecanismo de indexação e pesquisa em texto integral ,o *MS Index* Server, e uma extensão fornecida pela *Adobe*, o *Ifilter* PDF. Este último confere ao *Index Server* a capacidade de lidar com o formato PDF.

Sendo assim, esta classe já fornece todos os mecanismos necessários à gestão de uma sessão. Porém, a técnica utilizada consiste em atribuir à classe `Sessao` um estatuto de propriedade da classe `Session`. O valor acrescentado que a classe `Sessao` oferece consiste no facto de que esta já tem métodos específicos para a manipulação de cada uma das propriedades do estado do sistema, com todos os mecanismos de controlo (ex. id de um documento é maior do que zero) dessas propriedades. Assim sendo, a propriedade `Sessao` é lida no inicio da execução do *Servlet* e pode ser manipulada ao longo da execução deste, através dos métodos previamente construídos.

A classe `Motor` é a única com a capacidade de alterar os valores das variáveis de estado guardadas na classe `Sessao`. Desta forma, todos os pedidos são submetidos a esta classe, uma especialização da classe `HttpServlet` (ver em 3.2.3). Como esta classe herda os métodos da classe `HttpServlet` podem ser-lhe submetidos pedidos HTTP GET, ou POST (ver em 3.2.1), vindos do exterior (do cliente por exemplo). Depois de alterar o estado do sistema, esta classe envia uma notificação a cada uma das classes *servlet* responsáveis pela gestão do interface com o utilizador, que necessitarem de actualização. Esses classes poderão ser qualquer uma das seguintes: `GesDoc`, `GesAnot`, `GesMenu` e `GesMenuLink`. Todas estas classes são *servlets* e têm respectivamente as seguintes funções: manipulação do documento, visualização das anotações, gestão do menu de navegação e gestão do menu de hiperligações (planos de assunto).

Outra classe desenvolvida foi a `ConnSQL`, com a qual se pretendeu simplificar a comunicação com o SGBD. Usa classes da API JDBC como a `Connection`, entre outras, e tornou-se bastante útil, porque implementa funções específicas para a manipulação, dentro do SGBD, de entidades como: anotações, hiperligações, meta dados, utilizadores, etc..

As quatro figuras que se seguem ilustram respectivamente: o diagrama estático com o relacionamento das classes atrás mencionadas, dois diagramas de estado e uma descrição de caso de uso através de um diagrama de interacção. Os diagramas de estado descrevem apenas o estado geral do sistema na `Figura 30` e o super estado "visualização" na `Figura 31`. No entanto, são suficientes para se compreender a interacção entre os diversos *servlets*. A `Figura 31` descreve o comportamento do sistema no estado de "visualização". Neste estado, as variáveis `doc_id` e `pag` (ambas propriedades da classe `Sessao`) guardam respectivamente o identificador e a página do documento em visualização. Os eventos possíveis neste estado estão descritos no diagrama. Estes correspondem todos a eventos de *click* que por sua vez invocam o método HTTP GET ao *servlet* `Motor`, com os respectivos parâmetros. Sempre que um destes eventos ocorre, o *servlet* `Motor` altera os valores das propriedades da classe `Sessao` e invoca o método HTTP GET em classes como: `GesDoc`, `GesAnot`, etc.. O diagrama da `Figura 32` descreve o ciclo de comutação de plano de assunto. Neste caso, o utilizador depois de visualizar os planos de assunto (ver a `Figura 37`), num documento gerado pelo *servlet* `GesDoc`, que interceptam o documento activo pode optar por comutar para um desses planos, ou então, continuar no plano em que está cancelando a comutação.

Os eventos associados aos controlos são: `BT_PESQ_DOC`, `BT_PESQ_PLANO`, `BT_DOCUMENTO`, `BT_COMUTAR_PLANO`, `BT_DDOC`, `BT_DPAG`, `BT_PDOC`, `BT_PPAG`, `BT_ANOTAR`, `BT_PREV`, `CT_GOTO` e `BT_NEXT`. Estão representados na parte inferior (na barra de ferramentas com a mesma ordem da esquerda para a direita) da `Figura 33`. O botão BT_PLANO corresponde à hiperligação que dá acesso ao plano seleccionado, tem como argumento o identificador do plano.

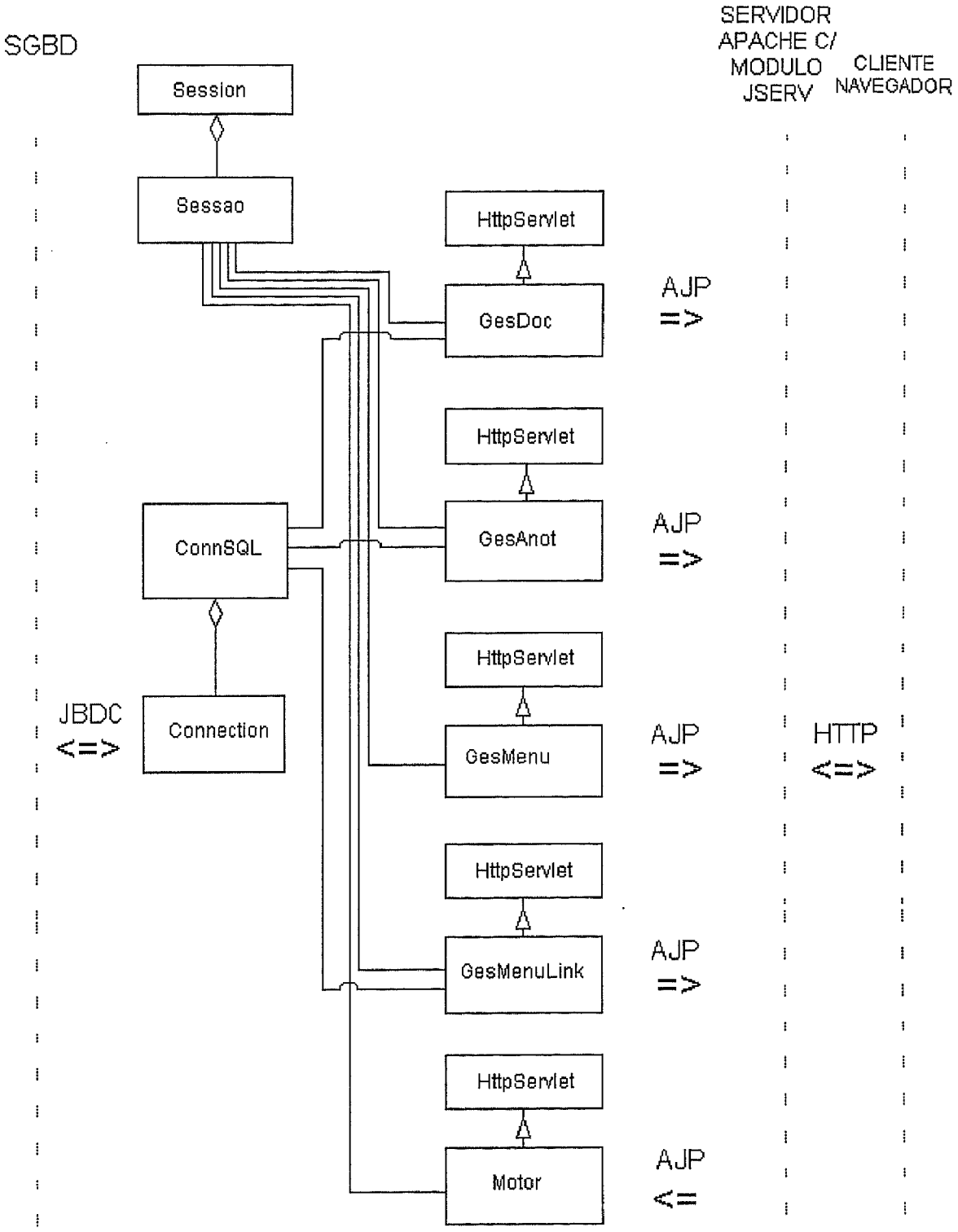


Figura 29 - Diagrama de classes do protótipo

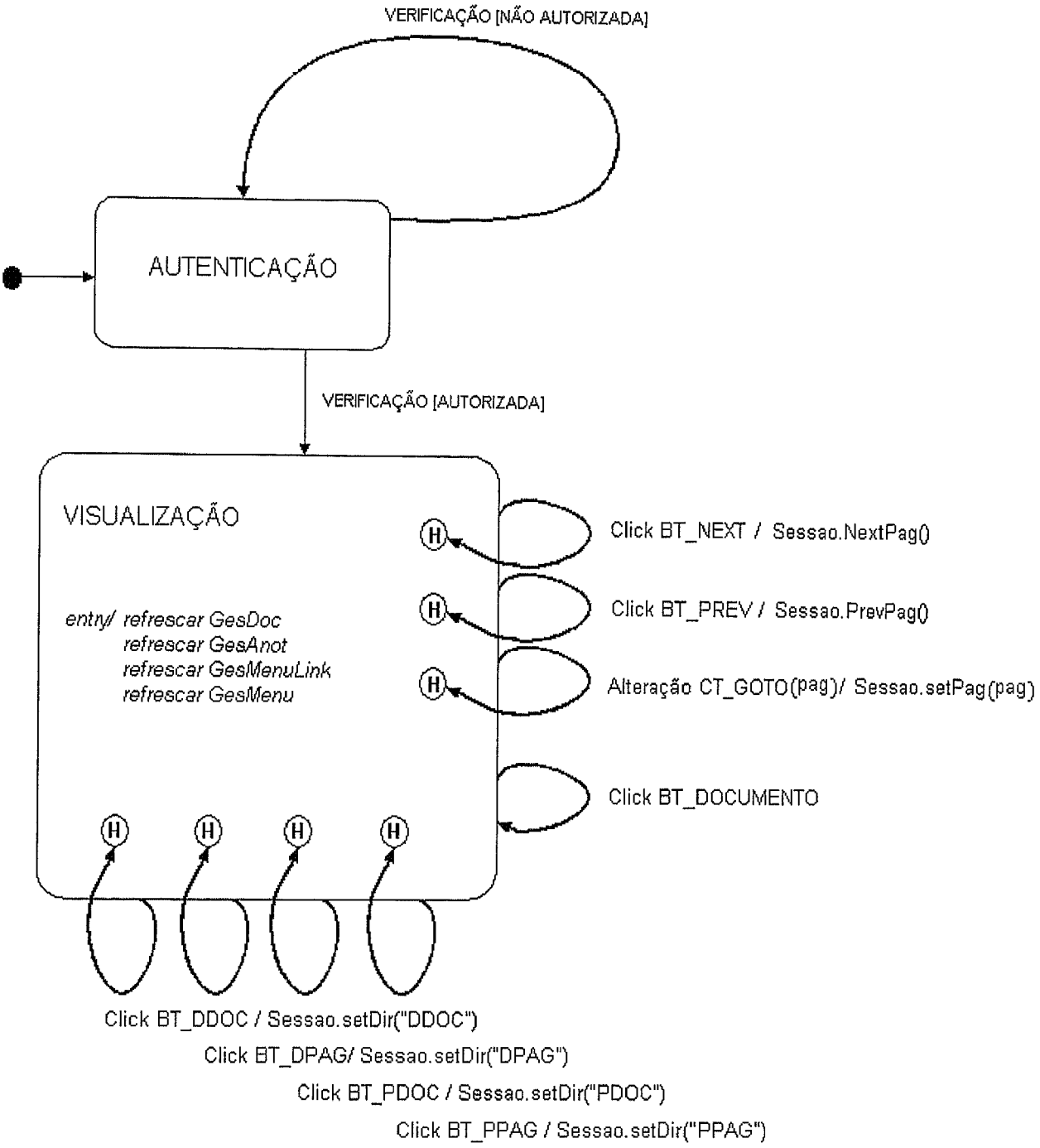


Figura 30 - Diagrama geral de estados

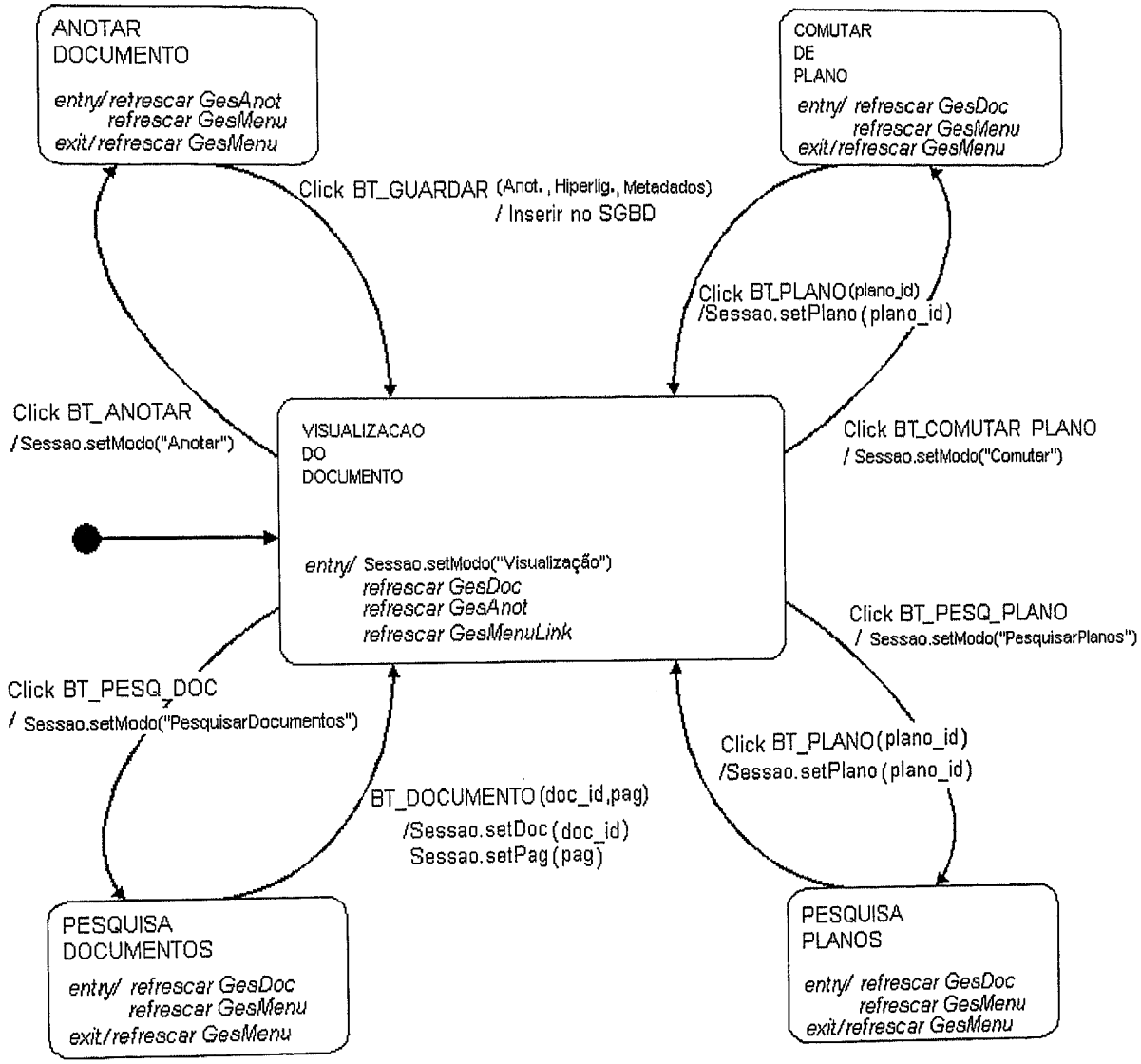


Figura 31 - Diagrama do super estado Visualização

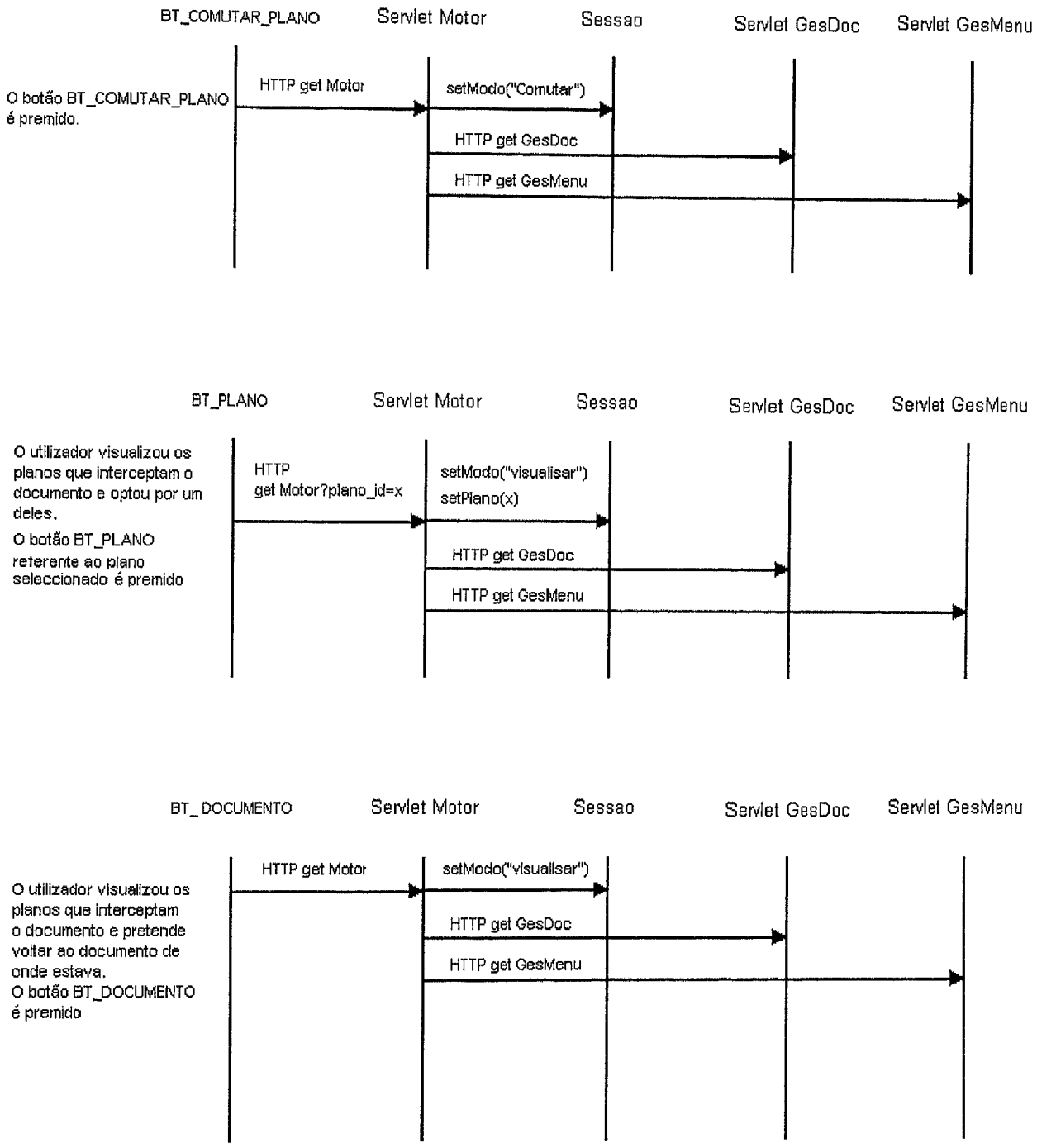


Figura 32 - Caso de uso de uma comutação de plano

Interfaces do protótipo com o sistema

Como já foi referido o mecanismo do interface da comunicação entre o protótipo desenvolvido e o SGBD, um servidor *Oracle 8i*, é suportado pela API *Java*, o JDBC. A *Oracle*[1] disponibiliza dois tipos de ligação JDBC às suas bases de dados: a OCI – *Oracle Call Interface*, que permite o acesso efectuado com chamadas directas à bases de dados recorrendo a linguagens de programação externas. E para além do JDBC OCI, a *Oracle* disponibiliza também o *JDBC Thin*. Este consiste num *driver* capaz de aceder ao SGBD via uma rede TCP/IP. A opção tomada foi o *JDBC Thin*, precisamente pela razão de permitir o acesso através de uma rede TCP/IP e o seu uso no lado cliente JDBC (o protótipo é cliente do SGBD) ser bastante mais simples

Também é através de uma rede TCP/IP que se estabelece a comunicação entre o protótipo e o servidor *Web*, um *Apache*. Neste diálogo é utilizado um protocolo designado por *Apache Jserv Protocol* - AJP[2], implementado do lado do protótipo por uma classe `Jserv` e do lado do servidor *Web* pelo módulo *Apache-JServ*. A utilização desta classe *Jserv* é totalmente transparente para os *servlets* do protótipo.

Dado que os protocolos aplicacionais usados nas duas ligações do protótipo assentam no topo da pilha protocolar TCP/IP, então o protótipo assim como esses dois elementos podem estar alojados em computadores totalmente distintos.

4.3.4 Integração com outros sistemas

Sistema de catalogação da biblioteca

Esta ferramenta de apoio ao trabalho de pesquisa, pressupõe a integração com um sistema bibliográfico de catalogação. O conjunto de meta dados da informação de catalogação, deve permitir a pesquisa pelo autor, título, assunto, data de publicação, etc.. No caso do protótipo desenvolvido fez-se a ligação a um sistema de gestão de bibliotecas *Aleph*.

Por outro lado, os artigos em formato PDF foram também submetidos a um processo de indexação, o qual torna possível a pesquisa eficiente, em texto integral. Desta forma, o referido sistema de catalogação da biblioteca e o mecanismo de indexação e pesquisa em texto integral, em conjunto com o repositório, implementam o primeiro nível do modelo proposto. Com isto, consegue-se um dos objectivos do serviço proposto (ver em 2.1), o aproveitamento e integração de recursos já existentes, e que todas as questões ligadas à catalogação e à criação de mecanismos de pesquisa já estejam resolvidas.

Localização dos documentos

É da responsabilidade do sistema de catalogação fornecer a localização do documento. O processo é bastante simples. O sistema implementado não conhece a localização dos

[1] O SGBD *Oracle 8i* é desenvolvido por uma empresa do mesmo nome.

[2] Mais informação em http://java.apache.org/jserv/protocol/AJPv11.html

documentos, isto é, a sua URL. Em sua substituição, usa um identificador, o registo bibliográfico do artigo científico, no sistema de catalogação. Sendo assim, sempre que é necessário obter a URL de um documento qualquer, o protótipo interroga o serviço de catalogação, dando-lhe o identificador do documento com parâmetro. Este processo de interrogação é descrito a seguir.

Conversão do identificador do documento na sua URL

Para se conseguir a transparência geográfica quanto à localização dos documentos, optou-se por utilizar um identificador do documento em vez da sua URL. Desta forma, foi necessário criar um mecanismo que permitisse a descodificação da URL. Então, foi implementada uma função (ver em 2.4) no lado do serviço de catalogação, que consiste num CGI (ver em 3.2.3) que aceita como parâmetro o identificador do documento PDF. Como resultado, é produzida uma página HTML que se limita a fazer um redireccionamento para uma URL. Nesta URL, existe um *servlet* (a classe *servlet* `Motor` do protótipo desenvolvido), que invoca um procedimento interno o qual, actualiza a variável interna do sistema que guarda a URL do documento PDF.

Este modo de passar a informação da URL, do serviço de catalogação, para o interior de um objecto, instanciado na máquina virtual *Java*, consistiu no aproveitamento do mecanismo utilizado para passar outros parâmetros vindos do interface com o utilizador. Para além desta vantagem, consegui-se com isto minimizar a intervenção do lado do serviço de catalogação. Mas sem dúvida alguma a maior vantagem é a total independência do protótipo implementado relativamente ao serviço de catalogação que usar.

Base de dados de catalogação distribuída

O repositório de documentos como já foi demonstrado poderá estar geograficamente distribuído. Sendo assim, os diversos documentos que constituem este arquivo distribuído certamente estarão catalogados nos respectivos sistemas. Desta forma, é extremamente importante que essa informação, também distribuída, possa ser reutilizada assim como acontece com os documentos. Uma solução consiste em recorrer a um meta motor de pesquisa que seja capaz de utilizar as bases de dados de catalogação desses repositórios. Para o sistema desenvolvido o uso deste meta motor é totalmente transparente pois este não passa de um motor de pesquisa que lhe fornece o identificador do documento e a respectiva URL. Uma solução destas exige apenas que não haja conflito entre os identificadores de documentos. É perfeitamente natural que em duas bibliotecas hajam dois identificadores iguais para documentos distintos. Sendo assim, em caso de não haver garantia de não colisão de identificadores, adicionalmente a este deverá existir um identificador da base de dados que lhe é concatenado, integrando-se no próprio identificador.

Autenticação

Justificada a necessidade de um processo de autenticação (ver em 2.3), procedeu-se à implementação dos mecanismos necessários. Para que este mecanismo possa validar os utilizadores, obrigatoriamente tem de possuir a informação necessária. Para gerir esta

informação de utilizadores, seria necessário também desenvolver ferramentas de gestão dos utilizadores.

Para evitar o desenvolvimento destas ferramentas, optou-se por estruturar o mecanismo de autenticação de modo a poder utilizar o sistema de informação[1] da faculdade. Com esta opção pretende-se que os utilizadores com acesso ao sistema de informação passem também a ter acesso a este novo serviço. Por outro lado, tendo já os utilizadores do Si-FEUP, um *username* e *password* atribuído, não têm decorar mais uma *password* e com isto simplifica-se também o processo de gestão de contas de acesso, do novo serviço. Este fica totalmente sustentado pelo sistema de informação, no que toca à gestão das contas de acesso.

Cada um dos documentos existentes no repositório, pode ser acedido por dois endereços: através de uma URL HTTP, e outra, que sendo embora também uma URL HTTP, é na prática um endereço de uma aplicação *servlet* com um parâmetro, o identificador do documento PDF ao qual se pretende aceder. A grande diferença entre estes dois métodos, é que, o primeiro existe para aqueles utilizadores que não têm acesso ao sistema e fornece apenas os documentos. O segundo, exige a autenticação de um utilizador do sistema e naturalmente fornece outros níveis de informação. O processo de autenticação pretende proteger as hiperligações e anotações dos utilizadores do serviço e não os documentos, pois estes estão sempre acessíveis através de servidor HTTP do respectivo repositório.

4.3.5 Controlo do documento PDF, embebido no navegador Web

Dado que a totalidade do material que constitui o repositório é constituído por documentos no formato PDF, foi necessário acrescentar a uma extensão ao subsistema cliente, de modo que fosse capaz de reconhecer o formato PDF. Essa extensão recorre à tecnologia *ActiveX* e consiste numa OCX fornecida pela empresa que desenvolveu o formato, a *Adobe*.

O navegador dotado com esta extensão passa a considerar o documento PDF como um objecto. À parte deste, o navegador contem ainda outros objectos, estruturados segundo um modelo de objectos do documento, o DOM. Através de *scripts, JavaScript* e *VBscript*, é efectuado o controlo da navegação no documento PDF, embebido no navegador *Web*.

É graças a uma das recentes inovações disponível apenas a partir da versão 4.0 do *Acrobat Reader*, que se tornou possível o avanço e retrocesso, na visualização da página do documento PDF. Esse controlo é sustentado num dos novos métodos do objecto *ActiveX*. Nenhuma das técnicas existentes até ao aparecimento desta nova versão da OCX que contém o método `SetCurrentPage()`, do objecto PDF, era capaz de uma forma tão simples de resolver o aspecto da navegação dentro do documento PDF. Em alternativa a esta solução para controlar o documento, seria necessário implementar um mecanismo capaz de reconhecer o formato PDF. O desenvolvimento desse mecanismo teria de recorrer a tecnologias OLE, DDE ou *Interaplication Comunication* – IAC[2]. Estas ferramentas estão disponíveis em bibliotecas de funções, acessíveis através de linguagens de programação como

[1] Serviço Si-FEUP, disponível através de http://www.fe.up.pt

[2] Mais informação em http://www.adobe.com

o C/C++. Obviamente, uma solução deste género teria um período de desenvolvimento muito mais demorado.

Ainda como alternativa à solução baseada em *ActiveX*, poderiam ser inseridas no documento, não em formato PDF, mas sim em *PostScript*, marcas, designadas por *Named Destinations*. Estas marcas permitem a abertura de um documento na página referida. Para isso, é necessária a edição do documento em formato *PostScript* e inserir-lhe após a linha "EndSetup" [ADOBE1985] a marca que define o destino. Esta marca tem a seguinte sintaxe:

/ DEST / nome / PAGE nr_pag / VIEW modo / DEST pdfmark

nome - nome pelo qual o ponto do documento será referenciado

nr_pag - página pretendida no documento

modo - modo correspondente à forma como o documento será visualizado

A hiperligação que permite o acesso a esta marca tem a seguinte forma:

http:// computador / [caminho completo do documento PDF] # nome

O último passo consiste na conversão do documento *PostScript* para o formato PDF.

De facto, a forma como isto permite chegar a uma página específica do documento PDF é bastante prático. No entanto, o mesmo não se pode dizer acerca do modo como essa ligação é estabelecida. Se não houver a possibilidade de edição dos documentos, esta solução, para além de não ser prática, torna-se impossível. Agravada pelo facto de ser um processo difícil de concretizar para uma quantidade de documentos como aquela que se prevê que exista no repositório.

Outras soluções, também pouco eficientes e práticas, foram encontradas, que nem merecem referência.

4.3.6 Vantagens do uso do XML nas transacções servidor / navegador

Com o intuito de tirar proveito das vantagens do XML, adoptou-se por separar a informação a ser exibida no interface, dos aspectos ligados à forma como essa informação é apresentada. Os estilos e a aparência do interface ficam totalmente sob controlo de folhas de estilo XSL e CSS. Desta forma, muita da complexidade dos *servlets* foi reduzida, dado que é bastante mais simples construir um documento XML, do que um documento HTML, com todos os pormenores de apresentação. A informação produzida pelo *servlet*, é enviada em XML para o processador XML do navegador, no lado cliente. Recorrendo ao processo descrito em 3.1.3 utilizando uma XSL *template*, o processador XML constrói o documento HTML.

Depois de tomada a opção foi necessário desenvolver uma definição do tipo de documento, uma DTD XML. Este tem de suportar a definição dos diversos elementos que constituem os documentos XML transaccionados entre os subsistemas servidor e cliente.

4.3.7 Interface com o utilizador

O interface com o utilizador, do protótipo desenvolvido, recorre a um navegador *Web*. Desta forma, os utilizadores poderão utilizar o sistema através da *Internet*, sem terem de efectuar qualquer parametrização para o seu uso. Por outro lado, a sua utilização fica limitada apenas à cobertura da própria *Internet* e se no ponto de utilização existe instalado um simples navegador.

O interface com o utilizador tem um ambiente estruturado em áreas funcionais específicas. Estas áreas são três: a primeira que de certa forma funciona como uma janela principal, permite ao utilizador a visualização do documento PDF. Esta área permite também funcionar como ecrã de pesquisa de documentos e planos de assunto e ainda para a comutação de plano. A pesquisa de documentos que pode ser efectuada, tanto num normal processo de procura de novos documentos, como num processo de estabelecimento de ligação a outros.

Existe uma segunda área utilizada como consola de navegação e barra de ferramentas. Esta parte do ambiente permite ao utilizador navegar no documento e nas respectivas anotações, e acesso às ferramentas do serviço, tais como: selecção do modo de visualização das ligações, de e para outros documentos, pesquisa e comutação de plano, pesquisa de documentos e anotar.

A terceira área está reservada à visualização, inserção, edição e remoção de anotações. A Figura 33, a seguir apresentada, ilustra a situação de visualização de um documento. No lado direito do ecrã são visíveis as anotações referentes a um dado plano de assunto. A Figura 34 ilustra um processo de anotação de um documento. Na parte direita da figura é visível o formulário de introdução da anotação.

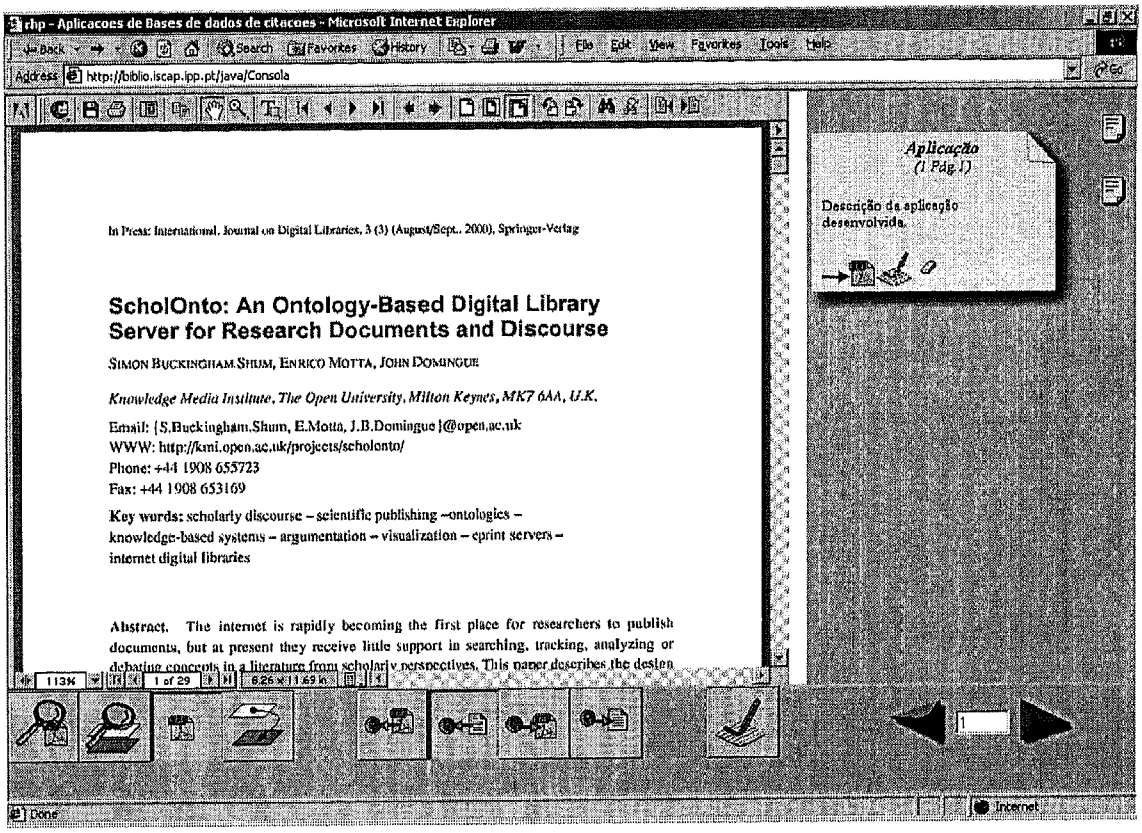


Figura 33 – Modo de Visualização

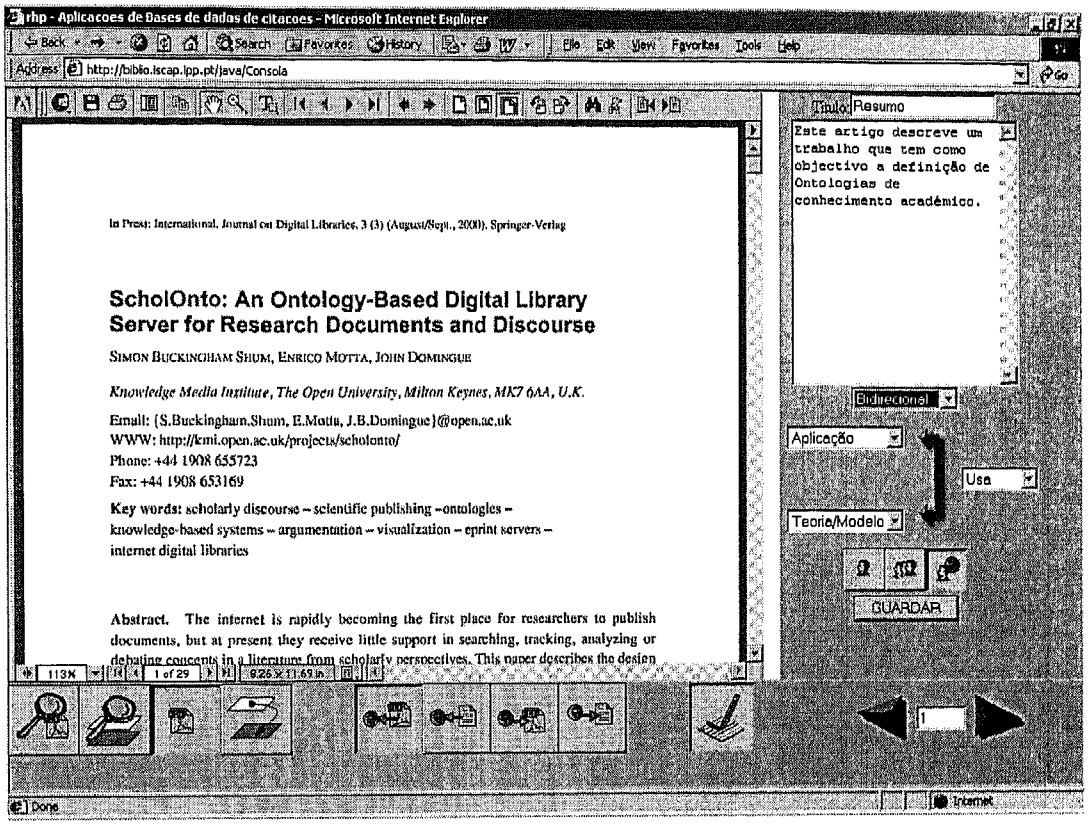


Figura 34 - Anotar um documento

O utilizador pode introduzir as suas anotações. Estas poderão ter uma hiperligação que pode ainda ser bidireccional Os meta dados que especificam a relação dessa hiperligação também são opcionais. Por defeito qualquer definição de conceitos é definida como "não definido". Obrigatoriamente uma anotação tem um título. Ao inserir a anotação, o autor, deverá ainda definir se essa é privada, para seu uso pessoal, ou pública (por defeito é pública).

A edição das anotações é outra facilidade do serviço. Cada autor pode alterar qualquer conteúdos das suas anotações e hiperligação, ou até remover a anotação.

Será de referir que um utilizador qualquer pode inserir anotações num plano que não é seu, porém, em futuras evoluções do serviço o dono do plano poderá ter a possibilidade de definir essa e outros tipos de permissões sobre um plano que tenha criado.

Como foi referido, a inserção de uma anotação pode ter uma hiperligação associada, que obviamente a relaciona com um dado plano de assunto. Essa hiperligação liga essa página do documento a outra página, desse, ou muito provavelmente de outro documento qualquer. Para a especificação do destino da hiperligação, o utilizador tem à sua disposição um conjunto de controlos de navegação na página. Mas se pretender que essa hiperligação tenha como destino outro documento, o utilizador acciona o mecanismo de pesquisa, o mesmo usado para as pesquisas, que lhe fornece o documento ao qual deverá querer estabelecer a ligação. A Figura 35 ilustra o ecrã de pesquisa, que pode ser utilizado nas duas situações referidas.

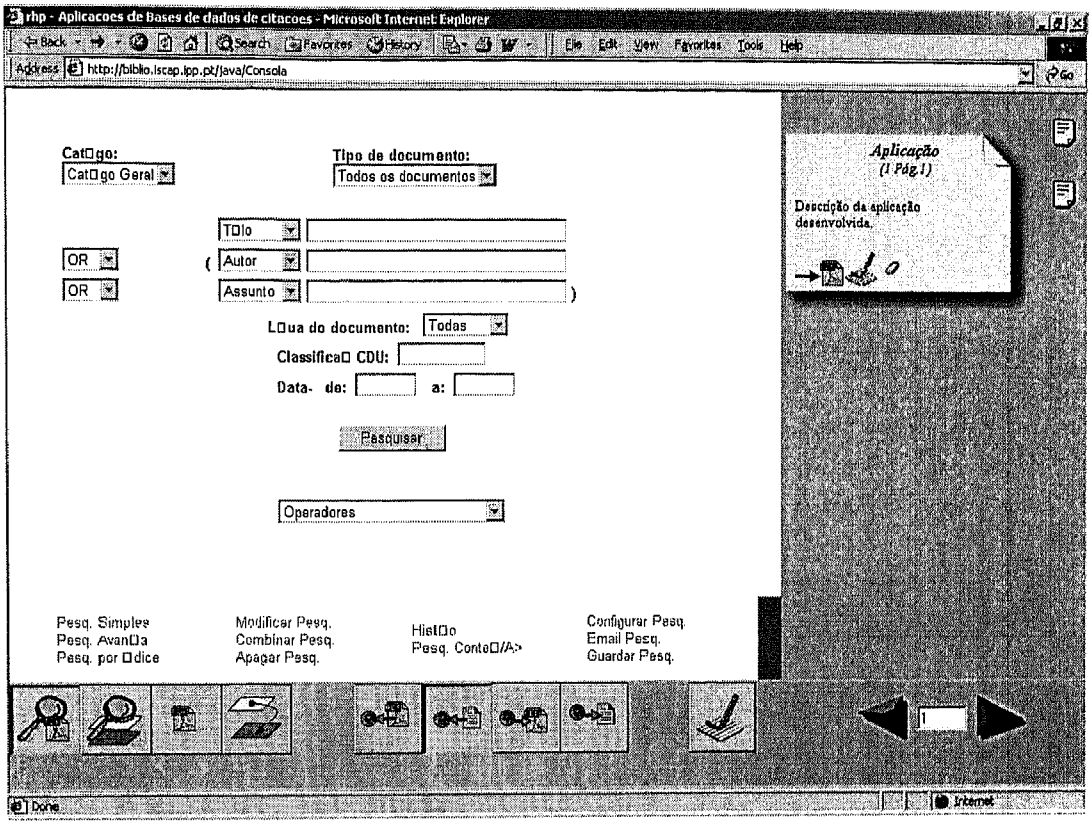


Figura 35 - Pesquisa e selecção de documentos

A comutação de um plano de assunto para outro, pode ser feita a qualquer momento, desde que no ponto de plano actual haja uma intercepção com aquele que é pretendido. A Figura 37 ilustra como são apresentados os planos. Porém o utilizador pode em qualquer momento fazer uma pesquisa sobre todos os planos de assunto. Desta forma, o utilizador pode comutar para outro plano, sem estar sujeito à existência de uma intercepção entre esses dois planos, o actual e o pretendido, a Figura 36 ilustra essa forma de comutar de plano.

Dada a possibilidade de associação do plano de assunto a uma página específica do documento, o utilizador tem quatro opções de visualização das anotações e hiperligações referentes a esse plano. Estas são as quatro combinações possíveis, resultado da opção: hiperligações para essa página, ou dessa página para outros documentos. Mais duas opções similares mas relativas à totalidade das hiperligações, de e para, o documento. O utilizador ao seguir um dado plano de assunto, em cada nó (um documento) tem estas quatro opções, lhe permitem obter os documentos e planos, que referenciam e são referidos, no documento e plano actual.

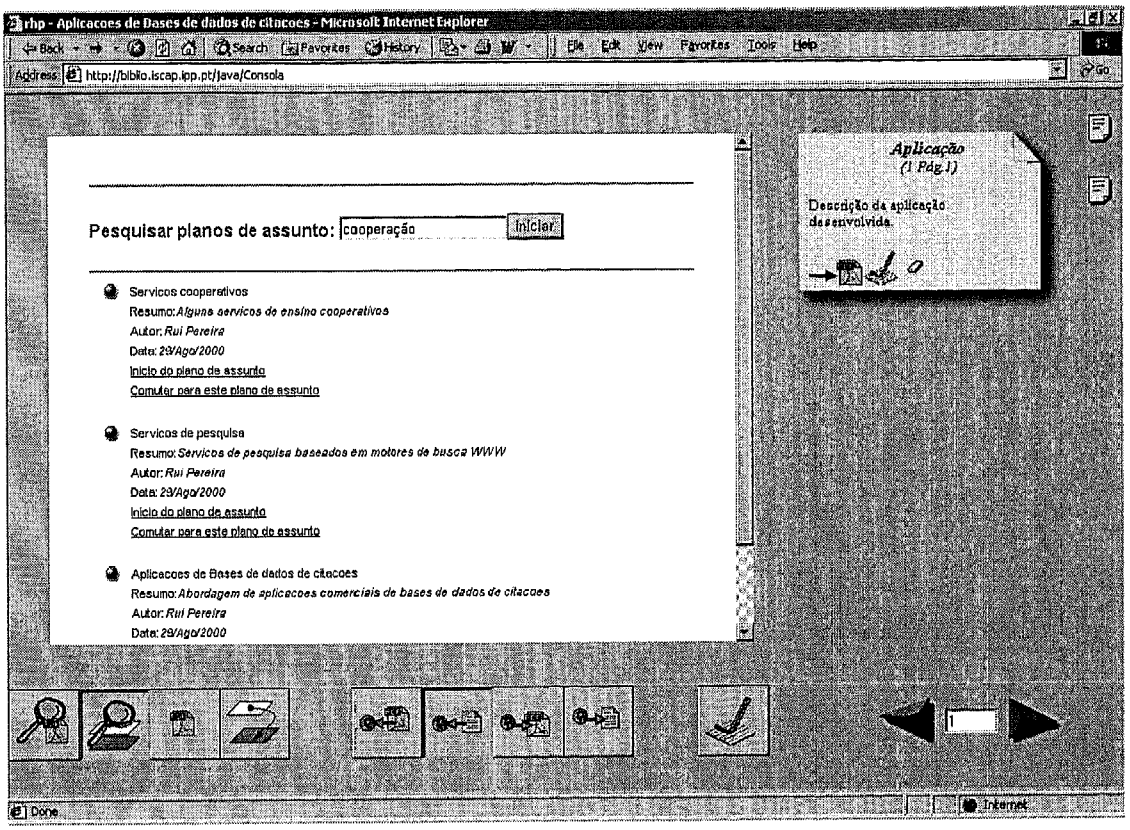


Figura 36 - Pesquisa e selecção de planos de assunto

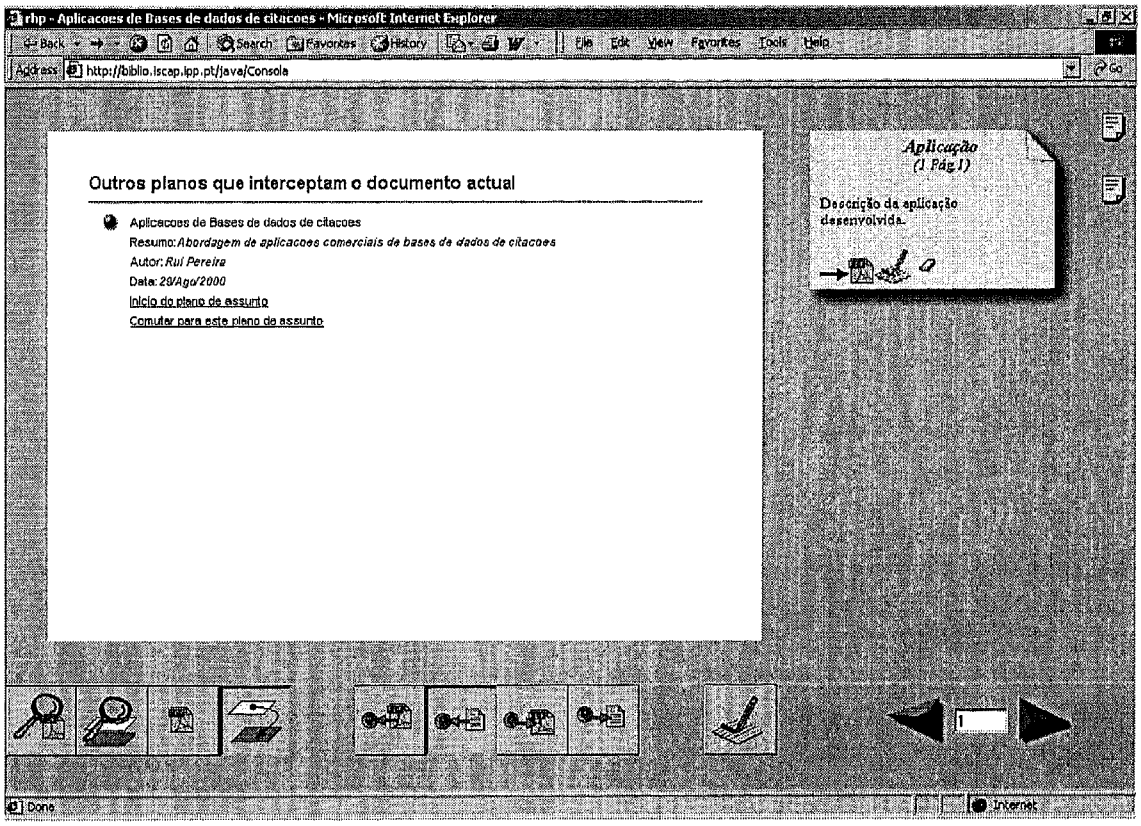


Figura 37 - Comutação de plano de assunto

Na caixa de texto, no topo da Figura 36, permite que o utilizador insira uma palavra chave para a pesquisa em todos os campos de informação acerca dos planos de assunto. Depois de submetida essa palavra na pesquisa, é obtida uma lista de planos. Cada um dos itens do resultado é um plano para o qual o utilizador pode comutar.

Conclusões

Com o volume de informação actualmente disponível na *Internet*, uma frustração comum entre os utilizadores é a falta de ligações expeditas entre documentos fortemente correlacionados. O objectivo deste trabalho é permitir a geração colaborativa de módulos de conhecimento que facilitem o trabalho de pesquisa e a investigação da comunidade científica. É também extremamente importante não permitir que estes módulos se tornem em ilhas, sem ligações interdisciplinares que promovam a disseminação de conhecimentos entre áreas científicas, que tantas vezes conduzem à inovação.

4.4 Conclusões gerais

Todas as abordagens efectuadas no segundo capítulo são unânimes ao afirmar que a *Internet* tem todas as condições para tornar-se no meio universal de partilha e acesso à informação. Em que esse acesso à informação deverá ser livre, sem qualquer tipo de restrição financeira. Também, não há qualquer divergência de opinião relativamente ao facto de que conforme a *Internet* está actualmente organizada, grande parte do seu potencial fica desaproveitado.

Os mecanismos de pesquisa são sobretudo baseados em pesquisa de texto integral e por palavra-chave, com sérias limitações, devidas tanto ao ruído como ao silêncio (ver em 1.1.3). Tudo isto leva a um excessivo consumo de tempo, para que seja encontrado o que se pretende. As pesquisas são demasiado orientadas às palavras e pouco aos conceitos. Consequência das limitações tecnológicas: a inexistência de ferramentas inteligentes; o diversidade formatos de arquivo dos documentos e incapacidade de representação de conceitos (ver em 1.1.3) na maior parte dos formatos.

Muita informação não é sinónimo de conhecimento ou valorização pessoal. A informação necessita de um estado de amadurecimento, obtido com: estrutura, níveis de informação adicionais e serviços associados. Estando a informação devidamente organizada, significa que existe um encadeamento lógico entre os inúmeros segmentos que a compõem. Então, facilita-se a sua compreensão e manipulação [RHRS1993].

Mas, como conseguir manter a informação estruturada, se ela não pára de aumentar? Uma possível resposta é: "Arrumando-a" no devido lugar logo que é produzida. Como essa informação surge de uma forma que ninguém pode prever nem controlar, que sejam os

próprios autores os responsáveis pela sua publicação e classificação. Isto na medida em que, são eles que melhor a conhecem e os mais interessados que esteja bem "arrumada". Naturalmente o processo tem de ser fácil e pouco trabalhoso, por que nem todos estão dispostos a perder muito tempo. O ideal seria que este processo de classificação, porventura o mais demorado, fosse baseado em mecanismos automáticos, como é o caso do aproveitamento das citações (ver em 1.2.1). Outra solução é aproveitar o espírito colaborativo entre utilizadores, estimulando-os a partilharem o conhecimento que têm sobre informação que considerem interessante [VBOD1999].

Mas apesar da actual situação não ser a ideal, existe o reconhecimento [RDC1997] de que muita coisa já mudou, sobretudo ao nível das publicações científicas, as quais, agora muito mais rapidamente são difundidas pelos meios científicos.

O acompanhamento da evolução de um trabalho, obriga a uma constante atenção nos meios difusores de informação, revistas, grupos de noticias, listas de distribuição, etc., implicando um esforço considerável. Todos estes serviços estão dispersos e desarticulados. A integração do maior número possível de serviços, sem dúvida alguma, só traria vantagens para os utilizadores evitando perdas de tempo desnecessárias à procura da informação.

Com a existência de grandes volumes de informação é necessário que esta seja dinâmica, que não fique estagnada no tempo e que acabe por torna-se num mero repositório. O utilizador que, enquanto leitor num meio electrónico, deve ter o poder de "tocar o texto" [JDB1991], isto é, introduzir na obra do autor as suas anotações e ligações, e disseminá-las livremente junto da comunidade científica. Estas anotações e ligações, têm um valor extremamente importante, pois podem constituir níveis adicionais de informação que enriquecem os conteúdos, por si já valiosos em termos científicos. Os níveis adicionais podem estar estruturados de várias formas e deverão contemplar o mais diverso tipo de informação e meta dados. Esta adição de informação, associada aos documentos e às suas interligações, podem esclarecer o utilizador (como leitor) de que forma se relacionam os diversos conceitos presentes nos documentos. Proporciona-se os mecanismos necessários à argumentação, para os autores, e à fácil compreensão das ideias, para os leitores. O resultado final é um meio dinâmico e integrado.

A aquisição e análise estatística de informações sobre o funcionamento dos sistemas, comportamento e preferências dos utilizadores, são funcionalidades que implementadas num serviço permitem aumentar a sua eficiência, resistência a falhas e agradar aos utilizadores, oferecendo-lhes serviços mais inteligentes. Porém, em qualquer sistema, especialmente os colaborativos, a privacidade é algo que tem de ser respeitado, sob a pena dos utilizadores perderem a confiança no serviço.

4.5 Modelo e serviço

4.5.1 Avaliação do modelo

O modelo utilizado na concepção do serviço proposto assenta numa estrutura de informação de três níveis distintos. A categorização da informação, e consequente separação, que o

modelo propõe, permite obter diversos tipos de vantagens: A reutilização de informação – um documento do primeiro nível pode pertencer a vários Planos de Assunto, em contextos completamente diferentes. Por outro lado, as hiperligações podem ser reutilizadas e partilhadas num número ilimitado de serviços, sem qualquer restrição de compatibilidade; A construção de ferramentas específicas para a manipulação das diversas categorias de informação; A gestão de grandes volumes de informação, torna-se mais fácil com estruturas do género daquelas propostas no modelo [RHRS1993].

A organização da informação, nessa estrutura, definida no modelo, permite a criação de funcionalidades bastante diversas como, por exemplo: Bases de dados de citações – recorrendo às hiperligações existentes no segundo nível; Ontologias – Graças às hiperligações e respectivos meta dados; motores de pesquisa – aplicados a qualquer nível de informação; Sistemas de recomendações – Recorrendo às anotações subjectivas (no terceiro nível) e aquelas mais objectivas, os termos em linguagem controlada (no segundo nível). Todas estas funcionalidades podem ser implementadas e integradas num único serviço, baseado numa estrutura como a que é proposta no modelo.

Este modelo correspondeu totalmente às expectativas criadas pelo serviço implementado.

4.5.2 Avaliação do serviço implementado

O modelo utilizado na concepção deste serviço, como já foi referido, não coloca qualquer restrição relativamente ao formato dos documentos no repositório. Este apresenta-se bastante flexível quanto a esse aspecto, porém, para não comprometer-se a implementação do serviço, e dado que grande parte, se não a totalidade, do fundo bibliográfico disponível está no formato PDF, considerou-se suficiente o suporte deste formato apenas. Atendendo que o serviço não pode à partida estar limitado apenas ao suporte deste formato, definiu-se uma arquitectura funcional (ver Figura 28) que não compromete futuras evoluções do serviço de modo a suportar outros formatos.

Outra opção cujas vantagens são mais do âmbito operacional, é a utilização do XML como formato de intercâmbio de informação entre os subsistemas servidor e cliente. Porém, apesar de vantagem efectiva, relativamente à sua utilização na manipulação da informação, tendo em vista a apresentação no GUI, o verdadeiro poder do XML reside na sua utilização como formato de suporte dos próprios documentos. Nessa situação, o documento XML, poderia ele próprio conter de uma forma estruturada a mesma informação dos diversos níveis referidos no modelo.

A *Web* é actualmente uma ferramenta universalmente utilizada, para a partilha de informação pelos seus utilizadores. Num cenário, onde o suporte de informação fosse efectuado recorrendo ao XML, e adicionalmente com mecanismos bem definidos no que toca à identificação e localização universal dos documentos, a *Web* poderia então proporcionar este mesmo serviço, mas, a uma escala muitíssimo maior.

4.6 Trabalho futuro

Relativamente ao modelo, tal como já foi referido, este satisfaz plenamente os requisitos colocados pelo serviço. Sendo assim, qualquer alteração estará sempre dependente do surgimento de novas exigências, colocadas pela evolução do serviço, que poderão permitir encontrar limitações no presente modelo.

Relativamente ao serviço, este foi implementado. A fase seguinte seria colocá-lo em utilização numa comunidade académica. Só assim, eventuais deficiências ao nível do modelo e melhoramentos, a qualquer nível, no serviços poderiam ser detectados. Após um período de seis meses de funcionamento, inquéritos aos utilizadores e a análise do registo de actividades fornecerão dados extremamente importantes para o delinear de evoluções do serviço.

Porém, o suporte a outros formatos além do PDF, tem bastante importância, portanto poderá ser esse um dos objectivos imediatos para a evolução do serviço. Facilidades como por exemplo: Foros de discussão associados a qualquer entidade[1]; Aviso automático ao utilizador interessado sobre a entrada de novas entidades no serviço; Obtenção do perfil do utilizador mediante o registo das suas actividades, tendo em vista a criação de funções inteligentes no serviço; A criação de interfaces gráficos, de navegação e pesquisa, que tirem proveito do conceito de módulo temático (ver em 2.2.5) abordado no terceiro capítulo; E a integração com outras ferramentas permitindo a reutilização dos módulos temáticos através da exportação como *Learning Objects*, que respeitem normas internacionais como a IMS *Content Packaging Specification* [TA2000] ou IEEE LTSC[2]. Desta forma, não só os documentos do repositório são susceptíveis de reutilização, mas também os níveis de informação que os utilizadores lhes acrescentaram. São estas e outras facilidades que poderão ser implementadas.

[1] Considera-se uma entidade um documento, anotação, hiperligação, ou qualquer outro elemento do Hipermeio

[2] Mais informação em http://ltsc.ieee.org/index.html

Referências

[ADOBE1985] PostScript - Language Reference Manual
Adobe Systems Incorporated
1985

[ADOBE1999] Portable Document Format - Reference Manual Version 1.3
Adobe Systems Incorporated
11 de Março de 1999

[AFPKPK1996] Alan O. Freier, Philip Karlton e Paul C. Kocher
The SSL protocol version 3.0
1996
http://www.netscape.com(eng/ssl3/draft302.txt

[ARMR1999] Alberto Riva e Marco Ramoni
LispWeb: a Specialized HTTP Server for Distributed
AI Applications
Computer Networks and ISDN Systems,
Volume 28, issues 7-11, p. 953
1996

[ASTB1993] A. Schubert e T. Braun
Reference standards for citation based assessments
Scientometrics, Vol.26, No. 1, pp. 21-35
1993

[BHCI1998] B. Bos, H. W. Lie, C. Lilley e I. Jacobs
Cascading Style Sheets, level 2 (CSS2) Specification
12 de Maio de 1998
http://www.w3.org/TR/REC-CSS2

[CMO1999] Carlos Manuel Oliveira
Cooperative Learning Centre: concepts, standardization issues and commercial approaches
9th DELOS Workshop - Digital Libraries for Distance Learning
Abril 1999

[CR1998] Carlo Revelli
Intelligence Stratégique sur Internet
Instituto Piaget
1998
ISBN 972-771-276-2

[CTRCORP1997] Search Engine Tecnologies for the World Wide Web and Intranets
Computer Tehnology Research Corp.
1997
ISBN 1-56607-993-4

[DC1982] D. Crocker
Standard for the format of ARPA Internet text messages
RFC 822
Agosto 1982

[DKLM1997] D. Kristol e L. Montulli

HTTP State Management Mechanism
RFC 2109
Fevereiro 1997

[DL1989] D. Lindsey
Using citation counts as a measure of quality in science:
Measuring what's measurable rather than valid Scientometrics
Vol. 15, Nos. 3-4, pp.189-203
1989

[DR1995] Dave Raggett
HyperText Markup Language Specification Version 3.0
Setembro 1995
http://www.w3.org/MarkUp/html3/CoverPage

[DR1997] Dave Raggett
HTML 3.2 Reference Specification
W3C
14 de Janeiro de 1997
http://www.w3.org/TR/REC-html32.html

[DRAHIJ1997] Dave Raggett, Arnaud Le Hors e Ian Jacobs
HTML 4.0 Specification - W3C Recommendation
W3C
18 de Dezembro de 1997
http://www.w3.org/TR/REC-html40-971218

[DRAHIJ1998] Dave Raggett, Arnaud Le Hors e Ian Jacobs
HTML 4.0 Specification - W3C Recommendation
W3C
24 de Abril de 1998
http://www.w3.org/TR/1998/REC-html40-19980424/

[DRAHIJ1999] Dave Raggett, Arnaud Le Hors e Ian Jacobs
HTML 4.01 Specification - W3C Recommendation
W3C
24 de Dezembro de 1999
http://www.w3.org/TR/html4

[DRC1999] Dustin R. Callaway
Inside servlets: server-side programming for Java platform
Addison Wesley Longman, Inc.
1999
ISBN 0-201-37963-5

[EF1998] Eilean Fairholme
Requirements, options & evaluation criteria for providing
an on-line expertise environment on an intranet
Fevereiro 1998

[EG1972] Eugene Garfield
Citation analysis as a tool in journal evaluation
1972

[EG1994] Eugene Garfield
The concept of Citation Indexing:
a unique and innovative tool for navigating the research literature
http://www.isinet.com/hot/essays/1.html

January 1994

[EM1998] Enrico Motta
An Overview of the OCML Modelling Language.
In Proceedings KEML'98: 8th workshop on
knowledge Engineering Methods & Languages
1998

[EPLC1999] Eduardo Pelegrí-Llopart, Larry Cable
JavaServer Pages Specification Version 1.1
Novembro 1999

[GHRCMF1997] Graham Hamilton, Rick Cattell, Maydene Fisher
JDBC Database Access Width Java
Setembro 1997

[HLBB1996] H. W. Lie e B. Bos
Cascading Style Sheets, level 1
17 de Dezembro de 1996
http://www.w3.org/TR/REC-CSS1-961217.html

[JDB1991] Jay David Bolter
Writing Space: The Computer, Hypertext, & the History of Writing
Hillsdale, N.J.: Lawrence Erlbaum Associates
1991

[JDD2000] James Duncan Davidson
Java API for XML Parsing Version 1.0 Final Release
Março 2000

[JFPHJH1997] J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach,
A. Luotonen, E. Sink e L. Stewart
An Extension to HTTP : Digest Access Authentication
RFC 2069
Janeiro 1997

[JP1982] J. Postel
Simple Mail Transfer Protocol
STD 10
RFC 821
USC/ISI
Agosto 1982

[JPJR1985] J. Postel e J. Reynolds
File Transfer Protocol (FTP)
STD 9
RFC 959
USC/ISI
Outubro 1985

[JR2000] J. Reagle
XML Signature Requirements
W3C/MIT
RFC 2807
Julho 2000

[KEB1995] Kipp E.B. Hickman
The SSL Protocol

1995
http://www.netscape.com/eng/security/SSL_2.html

[KSC1998] Kurt D. Bollacker, Steve Lawrence e C. Lee Giles
CiteSeer: Na Automous Web Agent for Automatic Retrievel and Identification of Interesting Publications
1998

[KSLM1994] K. Sollins e L. Masinter
Functional Requirements for Uniform Resource Names
RFC 1737
MIT/LCS
Xerox Corporation
Dezembro 1994

[LAA1999] Lada A. Adamic
The Small World Web
Third European Conference, ECDL'99
1999

[LLH1990] Lowell L.Hargens
Citation counts and social comparisons: Sientists use and evaluation of
Social Science Research, Vol. 19
1990

[NFNB1996] N. Freed e N. Borenstein
Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies
RFC 2045
Innosoft, First Virtual
Novembro 1996

[NW1975] Nicholas Wade
Citation analysis: A new tool for science administrators
Science, Vol. 188, No. 4183 (May 2), pp.429-432
1975

[OD1998] Olivier Dedieu
Pluxy: un proxy Web dynamiquement extensible
In Proceedings: The 1998 NoTeRe Colloquium
Outubro 1998

[PHG1983] Philip Howard Gary
Using science citation analysis to evaluate administrative accountability
American Psychologist, Vol. 38 (January), pp. 116-117
1983

[RDC1997] Robert D.Cameron
A Universal Citation Database as a Catalyst for Reform in Scholarly Communication.
First Monday
Fevereiro 1997
http://firstmonday.org/issues/issue2_4/cameron/index.html

[RHRS1993] Roger Harnden e Roy Stringer
Theseus - A Model for Global Connectivity.

Proceedings, UK Systems Society 3rd International Conference
Paisley
July 1993

[RR1992] R. Rivest
The MD5 Message-Digest Algorithm.
RFC 1321
Abril 1992

[RS1992] Roy Stringer
Theseus: A project at Liverpool Polytechnic to develop
a Hypermedia Library for Open and Flexible Learning.
Int. Fed. of Lib. Ass. vol 18
1992

[SGML1986] Information Processing
Text and Office Systems
Standard Generalized Markup Language (SGML)
ISO 8879
1986
http://www.iso.ch/cate/d16387.html

[SH1998a] Steve Hitchcock
Linking Electronic Journals:
Lessons from the Open Journal Project
D-Lib Magazine
Dezembro 1998
http://www.dlib.org/dlib/december98/12hitchcock.html

[SH1998b] Steve Hitchcock
Open Journal Project: final report to eLib
Novembro 1998
http://journals.ecs.soton.ac.uk/yr3/3rd-year-open.htm

[SHLC2000] Stevan Harnad e Leslie Carr
Integrating and Navigating Eprint Through Citation-Linking
2000
http://www.cogsci.soton.ac.uk/~harnad/citation.html

[SSEMJD1999a] Simon Buckingham Shum, Enrico Motta e John Domingue
Representing Sholary Claims in Internet Digital Libraries:
A knowledge Modelling Approach.
1999
http://kmi.open.ac.uk/techreports/papers/kmi-tr-80.pdf

[SSEMJD1999b] Simon Buckingham Shum, Enrico Motta e John Domingue
ScholOnto:An Ontology-Based Digital Library Server
for Research Documents and Discourse
1999
http://luntan.open.ac.uk/scholonto/docs/ScholOnto-IJoDL-2000.pdf

[SUN1998] JavaMail API Design Specification - Version 1.1
Sun Microsystems, Inc
Agosto 1998

[TA2000] IMS Content Packaging Information Model - Version 1.0
Thor Anderson
2 de Junho de 2000
http://www.imsproject.org/content/packaging/cpinfo10.html

[TB1994] T. Berners-Lee
Universal Resource Identifiers in WWW, A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web"
RFC 1630
CERN
Junho 1994

[TBDC1995] T. Berners-Lee e D. Connolly
HyperText Markup Language Specification -- 2.0
RFC 1866
Novembro 1995

[TBJPCS1998] T. Bray, J. Paoli e C.M. Sperberg-McQueen.
Extensible Markup Language. Recomendation (XML) 1.0
1998
http://www.w3.org/TR/1998/REC-xml-19980210

[TBRFHF1996] T. Berners-Lee, R. Fielding e H. Frystyk
Hypertext Transfer Protocol HTTP/1.0.
MIT/LCS
RFC 1945
Maio 1996

[TBRFHF1997] T. Berners-Lee, R. Fielding e H. Frystyk
Hypertext Transfer Protocol HTTP/1.1.
RFC 2068
MIT/LCS
Janeiro 1997

[TDCA1997] Tim Dierks e Christopher Allen
The TLS protocol version 1.0
1997
ftp://ftp.ietf.org/internet-drafts/draft-ietf-tls.protocol-06.txt

[TLFY1999] Tim Lindholm e Frank Yellin
The Java Virtual Machine Specification
Second Edition
Abril 1999
ISBN 0-201-43294-3

[VBOD1999] Vicent Bouthors and Olivier Dedieu
Pharos, a Collaborative Infrastructure for Web Knowledge Sharing
Third European Coference, ECDL'99
Setembro 1999

[W3C1998a] XML Query Language Proposal
Setembro 1998
http://www.w3.org/Style/XSL/Group/1998/09/XQL-proposal.html

[W3C1998b] W3C - XSL Working Group
The Query Language position Paper of the XSL Working Group
18 de Novembro de 1998
http://www.w3.org/TandS/QL/QL98/pp/xsl-wg-position.html

[W3C1998c] Document Object Model (DOM) Level 1 Specification
Version 1.0 - W3C Recommendation
1 de Outubro de 1998

http://www.w3.org/TR/REC-DOM-Level-1/

[W3C1999a] XSL Transformations (XSLT)
Version 1.0 - W3C Recommendation
W3C - XSL working group
Novembro de 1999
http://www.w3.org/TR/xslt.html

[W3C1999b] XML Path Language (XPath)
Version 1.0 - W3C Recommendation
W3C - XSL working group
Novembro de 1999

[W3C2000] XHTML 1.0: The Extensible HyperText Markup Language
A Reformulation of HTML 4 in XML 1.0 - W3C Recommendation
W3C
26 de Janeiro de 2000
http://www.w3.org/TR/xhtml1/