Fidus: Uma Ferramenta para Busca de Informações Personalizadas na Web

Edeyson Andrade Gomes

Fidus: Uma Ferramenta para Busca de Informações Personalizadas na Web

Edeyson Andrade Gomes

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Informática da Universidade Federal da Paraíba – Campus II, como parte dos requisitos necessários para obtenção do grau de Mestre em Informática.

Área de Concentração: Banco de Dados

Marcus Costa Sampaio
(orientador)

Campina Grande, Paraíba, Brasil
Agosto de 2001

FICHA CATALOGRÁFICA

GOMES, Edeyson Andrade

G633F

Fidus: Uma Ferramenta para Busca de Informações Personalizadas na Web

Dissertação de Mestrado, Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, Coordenação de Pós-Graduação em Informática,
Campina Grande, Paraíba, Agosto de 2001.

132p. Il.

Orientador: Marcus Costa Sampaio

Palavras Chave:

1. Banco de Dados
2. Recuperação de Informação
3. Disseminação Seletiva de Informação

CDU – 681.3.07B

FIDUS: UMA FERRAMENTA PARA BUSCA DE INFORMAÇÕES PERSONALIZADAS NA WEB

EDEYSON ANDRADE GOMES

DISSERTAÇÃO APROVADA EM 20.08.2001

PROF. MARCUS COSTA SAMPAIO, Dr.
Orientador

PROF. ULRICH SCHIEL, Dr.
Examinador

PROF. CARLO EMMANUEL TOLLA DE OLIVEIRA, Ph.D
Examinador

CAMPINA GRANDE – PB

A meus pais, Edeyson e Denise,
por seu grande amor
e pela herança que constroém para mim:
minha instrução.

Agradecimentos

A meus familiares, pelo apoio, dedicação e carinho nessa jornada.

Ao meu orientador, Prof. Marcus Costa Sampaio, pela confiança e empenho na cosntrução desse trabalho.

Aos colegas de mestrado, professores e funcionários.

Obrigado pelo carinho e ajuda sempre oportunos.

Resumo

O modelo de busca de documentos Web utilizado pelas ferramentas convencionais de busca, como Alta Vista, Cadê, Google e Radix, não atende adequadamente às necessidades dos usuários. O principal problema é que elas recuperam informações genéricas e muitas vezes inúteis, em detrimento de informações específicas e relevantes. Apresentamos, neste trabalho, uma solução de busca de informações personalizadas que provê informações relevantes a um usuário, segundo o seu perfil de consumidor de informação. Essa solução é baseada no algoritmo TF-Seno, uma variação do TF-IDF implementado neste trabalho, que permite uma maior precisão de busca de informações perfiladas.

Abstract

The search model used by conventional search engines such as Google, Altavista and Yahoo is intended to satisfy users that have immediate interest for a specific information context instead of users with regular interests. This model also doesn't provide ways to measure the precision of search results. We present a solution for information search that enables users with regular interests to create a personal profile as an information consumer, allowing the maximization of relevance in search results and to measure the precision of search results. This solution is based on TF-Seno, a TF-IDF variation algorithm implemented in this work.

Lista de Tabelas

Lista de Figuras

Sumário

Capítulo 1

Introdução

1.1 Contexto da Pesquisa

Embora a produção de conhecimento tenha crescido em escala geométrica, o acesso ao mesmo estava restrito, até poucos anos atrás, aos grandes centros de tecnologia e pesquisa. Com a popularização da Internet (Web), o acesso às informações produzidas nos mais remotos cantos do planeta tornou-se uma atividade bastante comum.

No entanto, o modelo de busca de informações na Web, como usado por muitas ferramentas de busca, não atende adequadamente às necessidades de um grande número de usuários, no que diz respeito ao requisito de achar rapidamente o que se procura, com eficácia e pouco esforço [POL 97]. Por eficácia, entende-se a capacidade dos mecanismos de busca de filtrar somente informações que sejam potencialmente relevantes às necessidades dos usuários. Pouco esforço significa que os resultados das buscas de informação sejam apresentados aos usuários de forma clara e precisa. Respostas que não fossem relevantes e claras deveriam indicar falha do sistema.

Um desafio a ser resolvido é então a mensuração da qualidade e da relevância das informações presentes na Web, de acordo com as particularidades de cada usuário. Uma informação que atenda aos critérios de busca de um determinado usuário não lhe é, necessariamente, uma informação relevante. O conceito de relevância não é portanto intrínseco à informação, mas dependente do usuário da informação. Mais precisamente, uma mesma informação pode ter um valor de relevância *x* para um usuário *a* e outro valor *y* para um usuário *b*.

Considerando-se a heterogeneidade de habilidades, conhecimento e comportamento dos usuários na manipulação de sistemas computadorizados, torna-se crescente a necessidade do desenvolvimento de aplicações que simplifiquem o processo de busca por

informações. Recursos como "busca avançada", provida por sistemas de busca como *Altavista*[1], ou sistemas com navegabilidade complexa, não são suficientes para resolver os problemas dos usuários, podendo até mesmo dificultar a interação usuário-sistema.

O crescente uso da Internet traz, cada vez mais, usuários leigos em informática ao uso das mais variadas ferramentas. Com isto, a usabilidade, isto é, a simplificação do modelo de interação usuário-sistema através de uma interface simples e intuitiva, torna-se um requisito fundamental para as ferramentas que visem prover a busca personalizada de informações, destinadas a um amplo espectro de usuários. A personalização não deve restringir-se apenas à seleção das informações de interesse particular, mas também contemplar os diferentes comportamentos dos usuários no processo de busca.

No mundo real, as buscas por informações são feitas através de solicitações verbais, termos imprecisos ou por meio da seleção em catálogos. Os usuários não precisam criar um conjunto de palavras-chave para representar suas necessidades, sentindo-se mais confortáveis em procurar informações através de temas hierarquizados que as classifiquem. A adequação deste paradigma de busca para o mundo virtual é uma necessidade premente. O uso de hierarquias de temas que classifiquem as informações deve gerar uma interação mais efetiva entre os usuários e os sistemas de busca, permitindo a obtenção de resultados mais precisos e em conformidade com as características particulares dos usuários.

1.2 Sistemas de Informação Personalizada

A principal motivação para a construção de Sistemas de Informação Personalizada (SIPs) advém da dificuldade atual dos sistemas convencionais de busca na obtenção de informações que realmente atendam às necessidades específicas dos usuários, num universo disperso e heterogêneo como a Internet. Em [LAR 97], apresenta-se a necessidade de elaboração de sistemas de busca alternativos aos existentes, que sejam mais precisos e eficazes, pois a crescente expansão do universo de busca de informações on-line faz com que o número de informações relevantes ou irrelevantes satisfazendo os critérios de uma consulta cresça constantemente.

[1] www.altavista.com

Como exemplo desta explosão de informações, as buscas pelos nomes "Jesus" e "Beatles" retornaram, respectivamente, 6.520.000 e 1.110.000[2] referências para documentos na ferramenta de busca *Google*[3]. Esta ferramenta não leva em consideração o contexto particular da busca e, dentro do conjunto resultado, a filtragem de que informações são efetivamente relevantes fica a cargo dos usuários. O mesmo acontece com outras ferramentas busca líderes do mercado, como *Altavista* e *Yahoo*[4].

O cenário apresentado reflete a situação de uma nova era, a era da disseminação do uso dos computadores, em que se produz e dissemina mais informação que o homem jamais pensaria ser possível há apenas alguns anos.

Qualquer pessoa com acesso à Internet pode disponibilizar seus conhecimentos, idéias, críticas, estudos, etc. a milhões de outras, com uma facilidade jamais vista. Como exemplo, o site da *Altavista* recebe cerca de 20.000 notificações de novas páginas Web por dia [TOR 98]. Em sua maioria, o conteúdo é uma reprodução de documentos já existentes ou de material de baixa qualidade, pois o nível de cuidado que se tem com a preparação de informações para publicação na Web varia muito. Independentemente da qualidade das informações, as ferramentas de busca providenciam a sua indexação, tornando-as acessíveis a seus usuários.

O desafio subjacente à nova geração de ferramentas de busca, os SIPs, é a seguinte: retornar milhões de referências a documentos como resultado de uma consulta, ou milhares, ou até mesmo centenas ou dezenas de referências, dificilmente seria aceitável, porque isto não viria de encontro aos requisitos de eficácia dos documentos recuperados, e de pouco esforço para 'digeri-los'. É também consensual que o desafio só será vencido com a interação usuário-sistema, para a definição das preferências dos consumidores de informação. Mais precisamente, usuários com interesse regular por informações em determinadas áreas de pesquisa bibliográfica necessitam de ferramentas que lhes permitam definir as informações de interesse, seus tipo e contexto, bem como as informações que não lhes interessam.

---

[2] Pesquisa feita em janeiro de 2001.
[3] www.google.com
[4] www.yahoo.com

1.3 Objetivos e Relevância do Trabalho

O objetivo principal deste trabalho é o desenvolvimento de um SIP, o *Fidus*[5], que capacitará seus usuários a determinar o espectro das informações de seus interesses. Com base no espectro, o *Fidus* garante a eficácia das buscas e o pouco esforço dos usuários para processar as informações recuperadas.

A relevância do *Fidus*, que o distingue de outras propostas de SIPS, reside em sua rica interação com os usuários, visando definir seus espectros de informação e na implementação de um algoritmo próprio para mensurar a relevância de documentos – TF-Seno.

Entre outros elementos importantes dessa interação usuário-sistema, destacamos: construção de hierarquias de classificação de informações; determinação automática da relevância potencial de um documento, com base na hierarquia de classificação de informações; apresentação dos documentos recuperados por ordem decrescente de relevância potencial; e avaliação, pelos usuários, da relevância potencial. No final da interação, todos os documentos recuperados são relevantes, em comum acordo usuário-sistema. Isto resolve a primeira parte do problema da busca eficaz.

Como levar em conta o aspecto extremamente dinâmico da Web? Através do conceito do *Fidus* de incremento de informações. Suponha que um usuário interage 'hoje' com o *Fidus* para definir uma hierarquia de informações. O resultado, segundo essa hierarquia, é um conjunto X de informações relevantes. 'Amanhã' o usuário faz nova consulta ao *Fidus*, para a mesma hierarquia de informações. O resultado é: $X + \Delta X$, em que ΔX é um conjunto, possivelmente vazio, de informações potencialmente relevantes. Através de nova interação com o usuário, é gerado um conjunto $\Delta X' \subseteq \Delta X$ de informações relevantes (além do conjunto X de informações relevantes).

[5] Amigo Fiel, em Latim.

1.4 Organização da Dissertação

No capítulo 2, apresentamos alguns conceitos relacionados à disseminação seletiva de informações, aos processos de busca e filtragem de informação e sobre sistemas de informações personalizados. Tais conceitos constituem a base estrutural do *Fidus.*

No capítulo 3, apresentamos os requisitos funcionais e não funcionais do *Fidus*, seu modelo de interação com os usuários e seu modelo arquitetural. Detalhamos o projeto e a implementação do *Fidus* no capítulo 4, onde também exemplificamos suas funcionalidades.

A avaliação experimental do *Fidus,* apresentando seus resultados e comparações com outro sistema similar, o *WebMate*, é detalhada no capítulo 5.

No capítulo 6, apresentamos as conclusões e as perspectivas para os trabalhos futuros.

Capítulo 2

Disseminação Seletiva de Informação

2.1 Introdução

A maioria das aplicações destinadas a localizar informações que atendam aos interesses dos usuários, como as ferramentas de busca *Google*, *AltaVista* e *Radix*[6], requerem que os usuários expressem claramente tais interesses. Logo, o usuário que deseja obter informações sobre um determinado contexto, ou área de pesquisa bibliográfica, deve saber representar tal contexto, caracterizando os possíveis documentos que podem lhe ser relevantes.

Comumente, para capturar o contexto da busca, os usuários devem fornecer critérios de busca como palavras-chave ou frases de consulta que orientem a pesquisa.A partir dos critérios de busca, as ferramentas de busca efetuam a seleção de documentos.

Quanto mais clara e consistente a especificação das necessidades de informação, mais concernente será a seleção de documentos. Caso os resultados apresentados não supram as necessidades dos usuários, estes devem iniciar uma nova busca, com os critérios iniciais refinados, ou com novos critérios de busca.

Um ponto negativo de tal processo é que, nessas novas interações, documentos já analisados e descartados por não serem relevantes podem ser exibidos novamente. Praticamente, inexiste nas ferramentas de busca mais conhecidas um controle sobre que informações são relevantes ou descartáveis.

É lícito então inferir que, tal como é feito atualmente, o processo de busca de informações é pouco eficaz. O desejável seria que os sistemas fossem pró-ativos, selecionando novas informações que pudessem interessar aos usuários, filtrando aquelas que já foram analisadas e descartadas, convergindo para um conjunto de documentos

[6] www.radix.com.br

realmente relevante. Sistemas com tais características devem ser vistos como verdadeiros disseminadores de informação.

Muitos requisitos são necessários aos sistemas de disseminação seletiva de informação. Pró-atividade nas buscas de informação, algoritmos que analisem a relevância potencial das informações recuperadas, interações para julgamento de relevância, e análise de históricos de busca, são alguns desses requisitos.

2.2 Provedores de Informação

Os sistemas de busca de informação precisam determinar quais as suas fontes de informação. Quanto mais rico o universo de busca, mais informações estarão potencialmente à disposição dos usuários.

A Web pode ser considerada como uma provedora de um espectro 'infinito' de informação. Deve ser ressaltado, porém, que a maioria das informações publicadas na Web é produzida com pouco cuidado na descrição do seu conteúdo e contexto. É suficiente citar um grande número de páginas HTML dispondo de poucos recursos para tal fim (quando eles existem). Com o surgimento da tecnologia XML [RUM 99] para a confecção de documentos-Web, essa limitação vai sendo pouco a pouco resolvida. Entretanto, XML exige o desenvolvimento de complexos mecanismos de busca.

Outro problema para o uso da Web como provedora de informação é a ausência de compromisso com a padronização de conteúdo, por parte dos autores de informação. Acrescente-se a isso a instabilidade dos documentos: páginas HTML podem ser alteradas, removidas ou movidas (alteração de localização) livremente pelos autores.

2.3 Busca de Informação: Aspectos Comportamentais

Normalmente, os usuários das ferramentas de busca de informação procuram suprir uma necessidade de informação imediata ou regular. A busca regular requer uma certa estabilidade no interesse por informações de um mesmo contexto, o que não ocorre com a busca imediata. O desdobramento disso é que o comportamento dos usuários, no primeiro caso - busca regular -, é passivo, enquanto que, no segundo caso - busca imediata -, é ativo.

O comportamento ativo é caracterizado por interações usuário-sistema, para o refinamento das buscas, até que as informações mais relevantes às necessidades imediatas do usuário sejam encontradas. Ferramentas de busca tradicionais como *Google e Altavista* são mais apropriadas para buscas imediatas, exigindo uma grande atividade dos usuários.

A atividade dos usuários é menor quando o seu interesse é regular. A razão para isso é que o refinamento das buscas atinge um limite ótimo para a recuperação de informações relevantes e, com isso, o usuário pode centrar-se na avaliação de informações novas.

Quando a busca é regular, as necessidades de informação dos usuários podem ser representadas através de uma estrutura bem definida, chamada de *perfil*. Um perfil serve de guia para os sistemas disseminadores de informação, a fim de determinar se, dada uma informação, ela corresponde ou não ao perfil de um usuário, enquanto consumidor de informação.

2.4 O Processo de Busca de Informação

O processo de busca de informação consiste em localizar e extrair informações de meios de armazenamento normalmente distribuídos e heterogêneos, segundo determinados critérios de busca.

2.4.1 Arquitetura Típica do Processo de Busca de Informação

A arquitetura típica do processo de busca de informação na Web é apresentada na figura 2.1.

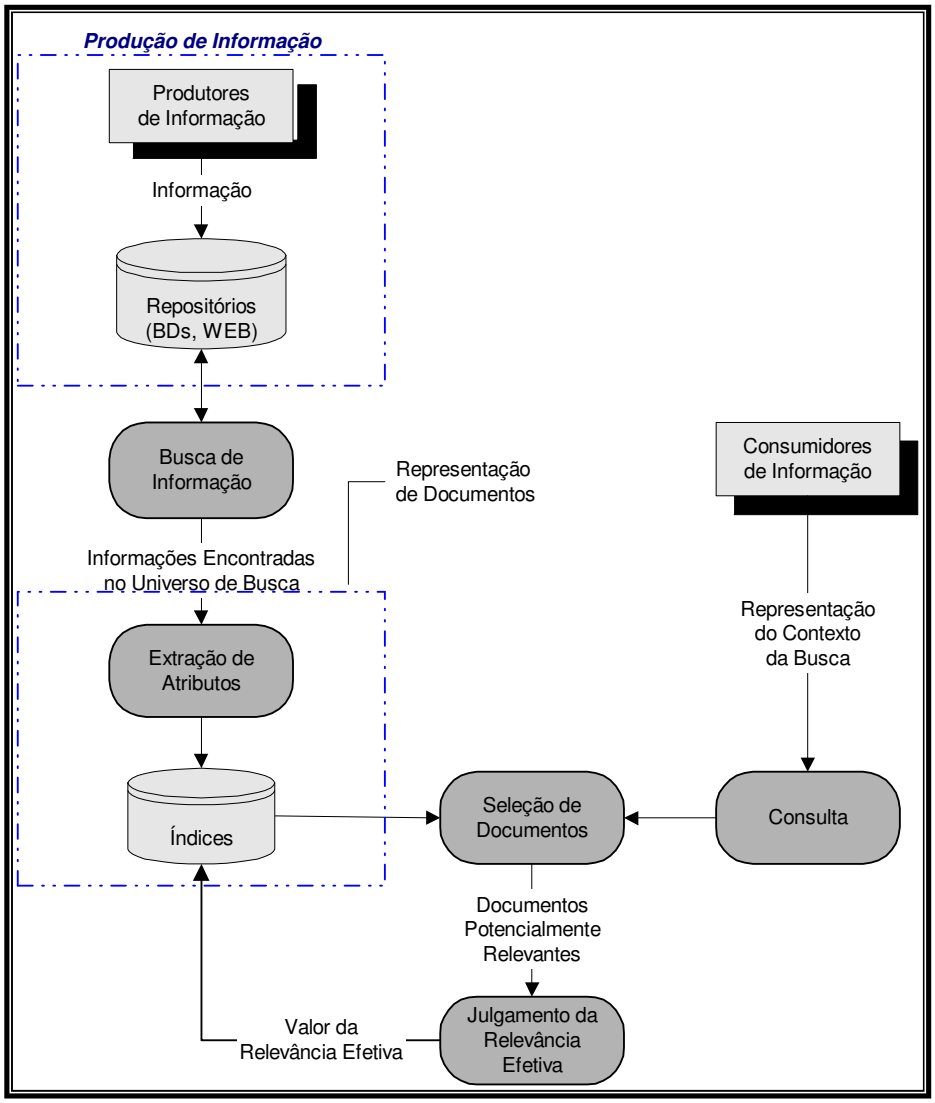


Figura 2.1: Processo de Recuperação de Informação na Web.

Consumidores de informação são usuários buscando informações para suprir uma necessidade imediata ou atender a um interesse regular numa determinada área da pesquisa bibliográfica.

O *contexto da busca* precisa ser formalmente representado para o sistema de busca, com base nos recursos oferecidos por este (caracterização através de palavras-chave, escolha de temas, etc). Esta representação deve permitir ao sistema compor *consultas* que permitam a recuperação das informações mais relevantes para atender às necessidades dos usuários.

A *busca de informação*, normalmente efetuada por *agentes de pesquisa,* é responsável por localizar novos documentos oriundos dos *produtores de informação*. Para estes novos documentos, o sistema de busca deve extrair seus atributos (título, descrição, termos mais freqüentes[7], etc) que serão usados posteriormente como seletores.

Os *produtores de informação* (entidades ou indivíduos que publicam informações na Web) têm seus documentos armazenados em bancos de dados ou sistemas de arquivos, como páginas HTML, por exemplo. Estes documentos, após serem localizados e extraídos pelos *agentes de pesquisa*, são indexados, normalmente, através de seus termos mais freqüentes.

A *seleção de documentos* corresponde à localização dos documentos indexados que possuam maior potencial de relevância para a consulta efetuada. Para determinar a relevância potencial dos documentos, os termos da consulta são comparados aos termos dos índices através de algoritmos que determinam o grau de similaridade entre os mesmos (seção 2.4.4).

O *julgamento da relevância efetiva,* efetuado pelos usuários para os documentos selecionados como potencialmente relevantes, consiste em determinar a relevância efetiva destes, ou seja, o quão relevante tais documentos são para atender suas necessidades.

[7] A freqüência do termo representa o número de vezes que o mesmo ocorre no documento.

2.4.2 Extração de Atributos

Os termos (atributos) que comporão os índices dos documentos devem ser criteriosamente selecionados para evitar problemas futuros na localização destes. Termos com baixa seletividade tornam os conjuntos de resultado muito grandes, enquanto termos de alta seletividade, se excluídos, podem anular a possibilidade de recuperação do documento.

A extração é feita, normalmente, através de um processo de filtragem de termos relevantes para a caracterização do documento (contexto). Este processo segue os seguintes passos:

1. Seleção de termos.
2. Descarte de controles ou símbolos de formatação de documento.
3. Descarte dos termos muito freqüentes e que não são seletivos o suficiente para descrever o contexto do documento ("*stopwords*").
4. Redução dos termos a um radical comum.

A seleção de termos é feita para as palavras que podem, dentro do documento, identificar seu contexto. Para documentos HTML, por exemplo, as palavras reservadas ("tags") são descartadas, como visto a seguir:

```
<html> <head> <title>
Este é um teste de extração de termos de um documento HTML.
</title>
Estas palavras serão filtradas para a escolha dos termos que
comporão os índices.
</head> </html>
```

Neste exemplo, os termos em verde são eliminados no passo 2; os termos em vermelho ("*stopwords*") são descartados no passo 3.

Stopwords são termos considerados irrelevantes para a determinação ou identificação de documentos, pois tais termos não são específicos o suficiente para

representá-los. Artigos e preposições, além de termos que aparecem em muitos documentos, são exemplos de "*stopwords*".

É comum a construção, pelos sistemas de indexação, de listas de "*stopwords*" que contém todos os termos a descartar dos índices. Sua construção pode ser manual ou automática. No primeiro caso, o especialista deve selecionar que palavras devem ser descartadas. No caso de construção automática, o sistema deve monitorar quais os termos que aparecem com maior freqüência na maioria dos documentos. Esses termos têm pouco poder de seletividade e, por isto, podem ser considerados como "*stopwords*".

Finalmente, no passo 4, aspectos gramaticais como gênero, grau, desinência, sinônimo, etc., devem ser analisados para reduzir os termos a um radical comum — redução de termos. Assim, plurais e palavras derivadas podem ser reduzidos a um termo comum, simplificando a elaboração dos índices e maximizando o poder de localização de documentos sobre o mesmo contexto.

Um outro aspecto muito importante a considerar é o uso de um vocabulário controlado para o processo de indexação e redução de termos, garantindo que apenas termos presentes no dicionário poderão ser inseridos nos índices. Desta forma, termos escritos erroneamente ou que não consigam ser reduzidos a termos do dicionário são excluídos dos índices.

2.4.3 Indexação de Documentos

Os elementos-chave do processo de busca de informação são os índices de acesso às informações. Os candidatos a chave de índice devem ser extraídos dos critérios de busca dos usuários (mais abrangentemente, de seus perfis de consumidores de informação). As chaves devem ser suficientemente seletivas para permitir que, a partir de termos que caracterizam um determinado conteúdo, os documentos correspondentes sejam localizados.

Dada a natureza extremamente dinâmica da Web, os índices devem ser continuamente atualizados. Isto é feito por meio de componentes de software chamados de agentes de pesquisa e captura de novos documentos.

Os agentes de pesquisa e captura possuem duas estratégias básicas para a procura de informações:

a. Busca em "*sites"* específicos.
b. Navegação livre na Web em busca de novos documentos.

As diferenças principais entre estas estratégias estão no tempo gasto na busca e no tamanho dos índices gerados para os documentos encontrados.

A primeira estratégia possui o universo de busca restrito e controlado, permitindo a recuperação de informações de alta qualidade ou de qualidade estimada. Sistemas que implementam tal estratégia tendem a prover alta precisão e eficácia nas buscas.

A segunda estratégia possui um universo de busca virtualmente ilimitado, com dimensão sempre crescente, que é a Web. Isto torna sua busca mais demorada, pois o número de documentos a recuperar é muito grande e os índices a gerar e atualizar são enormes. A qualidade dos documentos não é estimável, pois qualquer documento encontrado, independente de quem o produziu, é indexado.

O processo de indexação pode ser aplicado aos atributos de todo o documento ou de algumas partes deste. Neste último caso, devem ser determinadas quais as partes mais relevantes do documento, isto é, aquelas capazes de determinar, com grande precisão, seu contexto. Para isto, dois procedimentos mutuamente exclusivos são normalmente sugeridos:

1. Análise dos n ($\cong 100$) primeiros termos do documento, admitindo-se que em seu início devem se concentrar os atributos de maior relevância [HAR 93].
2. Eliminação de 10% dos termos mais freqüentes e 10% dos menos freqüentes, porque têm baixo poder seletivo do contexto [CRE 95]. Este valor percentual é dependente do tamanho e das características das coleções de documentos analisados. A freqüência dos termos é determinada pelo número de vezes que o mesmo aparece num documento.

Estes procedimentos visam reduzir o custo computacional da indexação e da posterior recuperação dos documentos, embora diminuam um pouco sua eficiência.

2.4.4 Relevância Potencial

O módulo de seleção de documentos, fazendo parte da arquitetura mostrada na Figura 1, deve localizar, nos índices, os documentos de maior potencial de relevância para atender aos requisitos dos usuários.

As principais técnicas propostas para selecionar documentos potencialmente relevantes nos índices baseiam-se em métodos estatísticos para busca e seleção de documentos. Nesta abordagem, vetores de termos são gerados para representar documentos [SAL 83]. Cada elemento do vetor representa um termo do documento e sua freqüência (número de vezes que o termo apareceu no documento).

Um documento ***d*** pode ser representado como:

$d = \{(t_1, f_1), (t_2, f_2), \ ..., (t_n, f_n)\}$, em que:

t_n é o *n-ésimo* termo do documento ***d***.

f_n é a freqüência do termo n no documento ***d***.

Este modelo de documento é conhecido como *Modelo de Vetor de Termos*. Suas principais variações estão na elaboração do conjunto de termos que servem como entrada para o vetor e como atribuir pesos a estes, para um determinado contexto.

Caso o sistema de recuperação utilize um dicionário de termos para validar a grafia destes, ou sua existência, a representação do documento ***d*** sofre uma modificação, como apresentado a seguir:

$d = \{f_1, f_2, \ ..., fn)\}$, em que:

f_n é a freqüência do termo n do dicionário no documento ***d.***

Considerando que este dicionário possui N termos, temos que $n \leq N$.

Caso o termo$_i$ não ocorra no documento, $f_i = 0$.

O uso do dicionário faz com que todos os documentos sejam representados por um vetor de dimensão N. Termos não encontrados no dicionário são descartados.

Uma variação do vetor de termos substitui a freqüência de um termo pelo seu peso no documento. O peso é a medida de quão seletivo o termo é para representar documentos do contexto.

Existem diversas formas de se atribuir pesos aos termos. Dado um termo, algumas fórmulas propostas (mutuamente exclusivas) para o cálculo de seu peso são:

a. O peso é igual a 1 se o termo estiver presente no documento, 0 em caso contrário.
b. O peso do termo corresponde ao número de vezes que o mesmo ocorre no documento (freqüência de termos).
c. O peso é igual ao inverso do número de ocorrências do termo numa coleção de documentos (freqüência inversa).

Os principais algoritmos de cálculo de peso determinam que o peso de um termo é diretamente proporcional à sua freqüência no documento (*TF - Term Frequency*) onde ele ocorre, e inversamente proporcional à sua freqüência na coleção de documentos do contexto (*IDF - Inverse Document Frequency*). TF-IDF [ALM ??], [GOF 99] e [TOR 98] é o algoritmo mais citado na literatura. O peso de um termo *t* para um documento *d*, em função de uma coleção de documentos *D,* é calculado da seguinte forma:

$$\text{Peso}_{td} = TF_{(td)} * \log (N/N_t)$$

$TF_{(td)}$ é o número de ocorrências do termo *t* no documento *d*.

N é o número total de documentos do conjunto *D*.

N_t é o número de documentos do conjunto *D* onde o termo *t* ocorreu.

Os pesos devem ser normalizados para que seus valores estejam contidos no intervalo [0, 1]. A normalização serve como fator de correção no cálculo de pesos onde a freqüência dos termos (TF) pode variar com a estrutura e o tamanho dos documentos.

Uma breve análise do TF-IDF demonstra que o peso de um termo que aparece em todos os documentos ($N = N_t$) é zero (possivelmente um "*stopword*"). Quanto maior o peso de um termo em um documento (isto é, quanto mais próximo de 1), mais relevante é este termo para a identificação do documento.

Relevância potencial é um valor que determina a probabilidade de uma informação recuperada atender a necessidade do usuário. Documentos cujos termos de maior peso correspondem aos termos da consulta que representa os requisitos dos usuários possuem

alta relevância potencial. Cabe aos usuários a análise dos documentos para determinarem a relevância efetiva dos mesmos, confirmando ou contestando a relevância potencial.

2.5 Composição de Consultas

As consultas que os usuários elaboram para representar suas necessidades de informação descrevem de alguma forma os documentos que podem lhes ser relevantes. A precisão da descrição é dependente do usuário e do sistema.

Diversos modelos de consulta têm sido propostos para localizar os documentos de maior potencial de relevância para atender às consultas. Apresentamos alguns destes modelos.

Modelo Booleano

O Modelo Booleano baseia-se na álgebra booleana para localizar documentos potencialmente relevantes à consulta. Cada documento é visto como um conjunto de termos. O universo de documentos é a união de todos os documentos armazenados e indexados.

A consulta é elaborada através de um conjunto de termos que visam descrever os documentos de interesse. O conjunto-resultado da consulta contém documentos potencialmente relevantes, segundo a consulta.

Não existe distinção, no conjunto de resultados, do grau de relevância dos documentos, isto é, todos os documentos localizados são igualmente relevantes para a consulta. Isto é uma desvantagem, por não corresponder à realidade. Uma outra desvantagem do modelo é que os termos da consulta são considerados igualmente significativos para a identificação dos documentos potencialmente relevantes.

As expressões booleanas que podem ser elaboradas para compor as consultas utilizam os operadores *AND*, *OR* e *NOT*. As ferramentas de busca mais conhecidas, além destes, ainda oferecem o uso do +, indicando os termos que devem estar presentes nos documentos, e do -, indicando termos que não devem estar presentes.

Estes operadores podem ser combinados para a elaboração de consultas mais complexas. Porém, a complexidade na elaboração de consultas muito refinadas faz com que os usuários não utilizem muito tal facilidade.

Modelo Probabilístico

O Modelo Booleano não distingue as informações recuperadas por grau de relevância potencial, obrigando os usuários a analisarem todo o conjunto resultado para encontrar uma informação realmente relevante para seus critérios particulares.

Necessita-se, então, de um modelo que qualifique as informações selecionadas pelo grau de relevância que estas podem ter para os critérios da consulta.

O modelo probabilístico visa estimar o quão relevante um documento seria para uma consulta — relevância potencial.

Para calcular a probabilidade de uma informação ser relevante a uma consulta, necessita-se determinar o grau de similaridade entre esta consulta e os documentos indexados.

De maneira simplificada, a similaridade entre documentos e consultas pode ser calculada, simplesmente, pela determinação de quantos termos eles possuem em comum. Neste cálculo simplificado, porém, não são considerados os pesos dos termos.

Uma técnica mais refinada para mensurar graus de similaridade utiliza vetores de termos com pesos [KOB 99]. Para um documento *d* representado pelo vetor $\{(td_1, pd_1), (td_2, pd_2), \ldots (td_n, pd_n)\}$ e uma consulta *c*, representada pelo vetor $\{(tc_1, pc_1), (tc_2, pc_2), \ldots (tc_n, pc_n)\}$, a função mais comum para efetuar o cálculo de similaridade entre *d* e *c* é a seguinte:

$$\text{Similaridade } (c,d) = \sum_{k=1,n} (p_{ck} * p_{dk}) \Big/ \sqrt{\sum_{k=1,n} (p_{ck})^2 * \sum (p_{dk})^2}$$

Onde: *c* é o vetor de termos que representa a consulta.

d é o vetor de termos que representa o documento.

p_{ck} são os pesos dos termos da consulta.

p_{dk} são os pesos dos termos do documento.

A consulta *c* precisa ser elaborada informando-se os pesos que cada atributo deve ter na seleção de documentos. Tanto as consultas quanto os documentos devem ser representados por vetores de iguais dimensões.

A precisão do modelo probabilístico é dependente da correta elaboração de consultas, pelos usuários, com a atribuição adequada de pesos aos termos que as representam. Caso isto não aconteça, a relevância pode ser computada de forma errônea, pois o algoritmo é fortemente dependente de pesos dos termos.

Consultas Orientadas por Categoria

Algumas ferramentas de busca de informação, como *Altavista* e *Cadê*[8], permitem que seus usuários elaborem suas consultas tanto através de coleções de termos (palavras-chave) e operadores (*booleanos*, "+" e "-") como por meio da seleção de temas organizados hierarquicamente para classificar informações — categorias de informação.

A consulta orientada por categoria requer que os usuários informem quais os temas de seu interesse. A associação entre os temas e as informações relacionadas é feita pelo sistema. Esta forma de busca é muito usada pelos usuários, comprovando a afirmação feita em [POL 97] de que os usuários são muito mais felizes procurando por informações através de categorias hierárquicas do que através de termos.

Tal afirmação é baseada na análise feita num universo de usuários com heterogeneidade de interesses e comportamentos na busca de informação. Para os usuários que mantêm um interesse regular numa determinada área de pesquisa bibliográfica, as ferramentas que categorizam as informações tornam suas buscas mais simples, pois essas buscas podem sempre ser orientadas pelas categorias. Ressalta-se, ainda, que a simplicidade nas buscas é um fator tão importante para o usuário quanto a efetividade das mesmas. Um sistema muito efetivo, que apresenta informações altamente relevantes, mas possui uma interface pouco amigável ou complexa, é pouco atrativo aos usuários [CRO 95].

Outro fator que contribui para a utilização de consultas orientadas por categoria é que a capacidade associativa natural das pessoas torna fácil a categorização de informações.

No entanto, como o modelo mental de organização e classificação de informações é particular a cada indivíduo, a categorização não deve ser imposta aos usuários como hoje ocorre nas ferramentas mais conhecidas. Ao contrário, tal categorização deve ser provida através de uma interface simples que permita ao usuário construir sua classificação de informações, segundo critérios próprios.

[8] www.cade.com.br

2.6 Avaliação da Eficiência da Recuperação

O desenvolvimento de técnicas efetivas de recuperação tem sido o núcleo das pesquisas em recuperação de informação por mais de 30 anos [CRO 95]. Neste período, várias métricas foram propostas para determinar a efetividade das buscas, as mais utilizadas sendo *abrangência* ("recall") e *precisão* ("precision") .

Abrangência

A abrangência determina a efetividade do sistema de busca na localização do maior número possível de documentos relevantes, dentro do universo de busca, para atender aos critérios de busca de um usuário.

A abrangência é calculada pela seguinte fórmula:

$$\text{Abrangência} = \frac{\text{Total de informações relevantes selecionadas}}{\text{Total de informações relevantes do universo de busca}}$$

A abrangência é um valor difícil de ser calculado, pois depende do conhecimento prévio de quantos documentos são realmente relevantes no universo de busca para atender aos critérios de uma consulta.

Precisão

A precisão determina a efetividade das buscas na recuperação de informações realmente relevantes, calculada pela proporção de itens relevantes recuperados contra o total de itens recuperados.

A precisão é calculada pela fórmula:

$$\text{Precisão} = \frac{\text{Total de informações realmente relevantes selecionadas}}{\text{Total de informações selecionadas}}$$

A precisão é uma métrica que pode ser calculada com base no *feedback de relevância*. Após o usuário receber o conjunto resposta para sua consulta, pode proceder com a análise de relevância informando quais as informações lhe são realmente relevantes e, com base nisto, o sistema pode calcular a precisão.

2.7 Sistema de Informação Personalizado

A adequação para o mundo virtual do paradigma de busca por informações que permita aos usuários procurar informações através de temas hierarquizados que as classifiquem, retornando todos os documentos relevantes do universo de busca, e apenas eles, é uma necessidade premente. Este é o objetivo de uma nova área de pesquisa, Sistemas de Informação Personalizada (SIP).

Ainda muito imprecisamente, um SIP é um sistema que provê informações a um usuário, segundo o seu perfil de consumidor de informação. O perfil de um usuário é a representação deste como consumidor de informação [FRE 99], informando os contextos de informação que lhe interessam. Através de um perfil, um SIP deve ser capaz de descobrir informações realmente relevantes ao usuário, e somente elas. Dada a natureza extremamente dinâmica da Web, é imperioso também que um SIP seja capaz de ajustar continuamente seu processo de busca.

O uso de modelos simples para a representação dos perfis e o *feedback de relevância* são requisitos importantes para garantir a eficácia dos SIPs.

Tais modelos devem possibilitar, ao usuário, a seleção de temas de interesse ou a construção dos mesmos. O retorno de relevância deve ser usado para auferir um dos principais objetivos dos SIPs, que é a seleção apenas das informações realmente relevantes ao usuário.

2.7.1 Perfis de Usuário

O paradigma de busca por informações através da seleção de temas hierarquizados que as classifiquem requer um modelo simples para tal classificação.

Os sistemas tradicionais de busca, como *Altavista e Yahoo*, oferecem estruturas hierárquicas pré-definidas para classificar suas informações. Todos os usuários de tais sistemas, independente de características particulares, dispõem da mesma classificação para selecionar informações. Porém, como usuários distintos diferem na forma de organizar e estruturar informações, os SIPs devem prover mecanismos que permitam aos usuários

criarem sua estrutura particular de temas. Esta capacidade é chamada de *personalização de informação*. Para a personalização, é importante definir os perfis dos usuários.

O tema tratado em [FRE 99] se aproxima do nosso conceito de SIP. Os autores discorrem sobre a necessidade de se construir coleções de informações de interesse dos usuários. Inspiramo-nos nessas idéias para a construção do que chamamos de perfis de usuários.

O perfil de um usuário é a representação deste como consumidor de informação, devendo conter a coleção de categorias de informação que atendem às suas necessidades particulares e todos os atributos que sirvam para caracterizá-lo.

Através do perfil, o SIP pode analisar as informações realmente relevantes ao usuário para poder ajustar seu processo de busca. Esta análise permite ao SIP determinar quais os critérios mais relevantes para a busca das informações em cada categoria selecionada pelo usuário.

Além do suporte a perfis, um SIP deve atender a outros requisitos.

2.7.2 Requisitos de SIPs

Um SIP ideal deve atender aos seguintes requisitos funcionais:

- ✓ ***Flexibilidade***

Os usuários devem dispor de um grande número de opções de configuração, para que o SIP possa se adaptar ao seu "gosto". Um alentado metabanco de dados deve prover as categorias, tipos, formatos, etc. das informações disponíveis.

Categorias de informação podem ser estruturadas hierarquicamente pelos usuários para respeitar suas individualidades e suas particularidades na associação de informações, pois uma estrutura hierárquica que classifique informações pode fazer sentido para um usuário e não fazer para um outro. Resumindo, o SIP deve suportar um modelo de metadados sob o qual devem ser construídos os esquemas de metadados específicos de cada usuário ou grupo de usuários.

Um exemplo de uma hierarquia de categorias de informação é o seguinte:

Figura 2.2: Categorias de Informação.

Neste exemplo, *Software, Banco de Dados, Internet*, etc. representam categorias de informação. Todas as informações sobre bancos de dados devem estar representadas sob a categoria *Banco de Dados*. Informações mais específicas, sobre temas mais específicos como *Data Warehouse*, são encontradas nos subníveis da hierarquia.

- ✓ ***Abrangência***

Os usuários desejam ter acesso à mais variada gama de informações possível, requerendo que os SIPs permitam que qualquer categoria de informação seja representada e que as informações correspondentes sejam localizadas. Para tanto, deve ser capaz de interagir com os mais diversos sistemas provedores de informações para sua localização e extração.

Por outro lado, as informações devem poder ser obtidas tanto em "estado bruto" como convenientemente tratadas. Informações em "estado bruto" são aquelas fornecidas sem transformações ou manipulações, típicas das ferramentas de busca atuais. De outro modo, as informações podem ser transformadas (mudança de formato e/ou de idioma, por exemplo), antes de sua apresentação aos usuários.

- ✓ ***Eficiência e eficácia nas buscas***

Além da agilidade dos sistemas em responder as suas consultas, os usuários requerem que elas sejam eficazes, isto é, que os conjuntos de respostas só tragam informações que lhes sejam relevantes. Diferentemente das ferramentas tradicionais de busca, em que o usuário monta os critérios de busca a cada consulta/interação, um SIP deve

ser pró-ativo: o momento em que a busca das informações é feita não necessiita a interferência do usuário (buscas off-line, ou antecipadas).

Para garantir a eficácia nas buscas, o SIP pode interferir dinamicamente nos critérios de busca das categorias de cada usuário. Isto é feito através da análise de documentos relevantes ao usuário, aplicando-se algoritmos que extraiam os termos mais seletivos destes documentos, como o TF-IDF.

- ✓ ***Segurança***

Os metadados concernentes a um usuário devem ser privativos e a disponibilidade dos mesmos deve estar vinculada a um mecanismo de autenticação. Além do mais, dados estatísticos e padrões de comportamento dos usuários, gerados pelo sistema, devem ser protegidos.

Como citado em [SUM 98], a segurança pode estar integrada às de sites que possuam mecanismos de busca com autenticação própria (como o ACM[9] ou IEEE[10]) . Desta forma, os SIPs poderão recuperar informações nestes sites.

- ✓ ***Atualização***

Como os provedores de informações da Web estão inseridos em ambientes autônomos, muitas informações podem ser removidas, alteradas ou ter sua localização modificada. Como conseqüência, informações não existentes ou desatualizadas podem ser fornecidas aos usuários.

Cabe então, aos SIPs, verificar e validar a atualidade e localização das informações.

- ✓ ***Compartilhamento***

Sendo as hierarquias de categorias de informação tão importantes aos SIPs, usuários que tenham refinado bastante suas categorias podem desejar compartilhá-las com outros que possuam interesses comuns. Este procedimento deve ser garantido sem afetar os requisitos de segurança.

---

[9] www.acm.org

- ✓ ***Compatibilidade***

Um SIP deve atingir uma vasta gama de usuários através da Internet e ser independente de plataforma.

2.7.3 A Pesquisa em SIPs

Um componente essencial de SIPs é a sua ferramenta de busca de informações personalizadas. Uma tal ferramenta deve ser capaz de determinar o contexto dos documentos recuperados por um usuário, ou o seu perfil de consumidor de informação, ao longo do tempo.

Comentaremos três das mais representativas ferramentas de busca de informações personalizadas. Assinalamos de antemão que, em todas elas, os perfis dos usuários não são gerados e mantidos de forma persistente.

WebMate

O *WebMate* é uma ferramenta para recuperação de informações personalizadas na Web, desenvolvido na Universidade de Carnegie Mellon [CHE 98].

Interceptando todas as páginas Web que serão apresentadas no *browser* dos usuários, o *WebMate* modifica-as, adicionando controles para que os mesmos efetuem o julgamento de relevância. Para cada documento julgado relevante, o *WebMate* aplica um algoritmo similar ao TF-IDF para atualizar o perfil do usuário, que consiste num vetor de termos com pesos representando o contexto dos documentos selecionados.

Como um primeiro passo para a busca personalizada, *WebMate* faz busca genérica, utilizando um conjunto de ferramentas de busca genérica, como *Altavista, Lycos, Yahoo*, etc. Em novas consultas, o usuário pode solicitar ao *WebMate* que busque documentos condizentes com seu perfil.

Podemos citar diversos problemas com o *WebMate,* que são:

---

[10] www.ieee.org

1. A seleção manual de documentos relevantes. Um SIP deveria ser capaz de descobrir, ao menos semi-automaticamente, documentos relevantes, baseando-se em perfis de usuário.
2. Os perfis gerados não são estruturados, o que dificulta enormemente a recuperação parcial de documentos, segundo uma determinada classificação.
3. No WebMate as consultas são fechadas, não permitindo que o usuário interfira na determinação dos critérios utilizados para as buscas.
4. O *WebMate* apenas permite que o usuário informe quais os documentos lhe são relevantes, elaborando apenas o perfil positivo.
5. Apenas um contexto de interesse é representado no perfil de cada usuário.

Syskill & Webert

Syskill & Webert é uma ferramenta projetada na universidade da Califórnia e que permite ao usuário julgar a relevância dos documentos consultados na Web à medida que este vai navegando [PAZ 9?].

Mantém múltiplos perfis, segundo cada tema de interesse do usuário, que contém critérios para determinar, por similaridade, novos documentos potencialmente relevantes.

Com base nos perfis, *Syskill & Webert* provê aos usuários endereços na Web que lhe são potencialmente relevantes e documentos encontradas através da ferramenta de busca Lycos[11] .

Syskill & Webert pode ser considerado uma evolução do *WebMate* em dois aspectos: ele é capaz de descobrir automaticamente documentos relevantes a um usuário, e pode produzir múltiplos perfis, segundo cada tema de interesse do usuário.

Infelizmente, os perfis ainda são não estruturados, as consultas permanecem fechadas, e a única ferramenta de busca genérica que pode ser acoplada ao Syskill & Webert é a Lycos.

[11] www.lycos.com

WebSail

WebSail [CHE 00] é uma ferramenta de busca personalizada que fornece aos usuários resultados de busca classificados por ordem de relevância potencial.

Através do julgamento da relevância efetuada por um usuário, aplica seu algoritmo similar ao TF-IDF para determinar, dentre os documentos realmente relevantes, quais os termos que podem ser incluídos no perfil do usuário.

Mantém os mesmos problemas anteriores de consultas fechadas, não persistência de perfis e gerência de contexto único.

2.8 Conclusão

Do exposto neste capítulo, pode-se chegar à conclusão de que SIPs são um assunto de pesquisa ainda incipiente, apesar de sua extrema importância e necessidade.

A agenda de pesquisa para melhorar os SIPs tem como elementos mais importantes os algoritmos de cálculo de peso dos termos e de cálculo de similaridades para a determinação da relevância potencial.

O julgamento da relevância efetiva das informações apresentadas aos usuários em resposta às suas consultas deve servir de base para que os sistemas ajustem possíveis discrepâncias da relevância potencial e avaliem sua precisão.

Nossa proposta de SIP, *Fidus*, tem como principal objetivo prover a busca efetiva de informações através de perfis bem estruturados e configuráveis pelo usuário. Os resultados das consultas serão ordenados pela relevância potencial e as informações deverão ser julgadas para que o usuário determine sua relevância efetiva.

O *Fidus* manterá um perfil independente para cada usuário e permitirá que o mesmo interfira na determinação dos critérios utilizados para as buscas, o que o torna um SIP *aberto*. Múltiplas ferramentas de busca genérica podem ser acopladas ao *Fidus*, maximizando sua abrangência.

Atendendo a todos os requisitos de SIPs, o *Fidus* deve ser um precursor de uma nova geração de ferramentas de busca de informação, efetivas e eficazes.

Capítulo 3

Fidus: Uma Nova Ferramenta de Busca Personalizada

3.1 Introdução

Ferramentas de busca de informação, como *WebMate, Syskill & Webert e WebSail*, não podem ser confundidas com um verdadeiro sistema de informação personalizado — SIP —, pois não atendem a seus requisitos, enumerados no Capítulo 2 [POL 97].

Um SIP é muito mais que uma ferramenta de busca, agregando estratégias que suportam recuperação inteligente de informação ([POL 97], [FUC 98], [BEL 96]). Essa inteligência é conseguida através da estreita interação SIP-usuário, na definição das preferências do usuário quanto aos seus interesses por informação e no julgamento de resultados obtidos. Em [BEL 98], o autor destaca a importância da interação SIP-usuário para a determinação da relevância dos resultados obtidos.

Neste capítulo, descrevemos nosso projeto de SIP — *Fidus* —, sua arquitetura e sua interação com os usuários.

3.2 Objetivos do Fidus

O *Fidus* é um SIP que, baseado nos principais conceitos de disseminação seletiva de informação, deve permitir a seus usuários o acesso personalizado e preciso às informações que atendam seus interesses.

O usuário alvo do *Fidus* é aquele com interesse regular por informações em determinadas áreas de pesquisa bibliográfica. Essas áreas de interesse podem ser estruturadas em categorias hierárquicas particulares para representar os interesses individuais, compondo o perfil de cada usuário como consumidor de informação.

O *Fidus* deve manter as informações já analisadas pelos usuários para evitar que as mesmas sejam reapresentadas em resultados de novas consultas. Os outros recursos do *Fidus* serão paulatinamente mostrados, no decorrer do capítulo.

3.3 Requisitos Funcionais do *Fidus*

Os requisitos funcionais exprimem as principais responsabilidades do *Fidus* para atender aos interesses dos usuários, compreendendo:

- ✓ **Personalização de informação**

  O *Fidus* deve permitir que os usuários construam sua própria classificação hierárquica de informações, descrevendo seus temas de interesse segundo critérios pessoais.

- ✓ **Auxílio na construção de hierarquias de informação.**

  O *Fidus* deve dispor de um administrador com privilégio de criar uma classificação hierárquica de informações que sirvam de exemplo para seus usuários.

- ✓ **Precisão e abrangência na busca de informação**

O *Fidus* deve prover aos usuários todas as informações relevantes localizáveis em seu universo de busca, e somente elas. Esse universo de busca não deve se limitar a uma única ferramenta básica (meta-busca).

- ✓ **Visão consistente e organizada das informações**

As informações devem ser apresentadas ordenadas pelo grau de relevância potencial estimada pelo *Fidus*.

- ✓ **Julgamento de relevância**

Os usuários deverão interagir com o *Fidus* para efetuar o julgamento de relevância efetiva das informações apresentadas nos resultados às consultas. Confrontando este julgamento com a relevância potencial estimada, o *Fidus* deve poder ajustar possíveis divergências em suas estatísticas de cálculo.

- ✓ **Gerência de atualidade de informação**

Diferenciando-se das ferramentas de busca tradicionais, onde o refinamento de uma consulta pode trazer as mesmas informações já apresentadas aos usuários, cada iteração periódica de busca do *Fidus* deverá localizar apenas novas informações, isto é, aquelas que ainda não foram analisadas pelos usuários.

- ✓ **Adequação de perfil**

As informações julgadas pelos usuários como realmente relevantes devem servir ao *Fidus* para o cálculo de pesos dos termos. Os mais significativos desses termos irão representar a correspondente categoria de informação no perfil.

- ✓ **Aprendizado de contexto de interesse**

Com base na navegação do usuário na Web e no julgamento de informações como realmente relevantes, o *Fidus* deve determinar o contexto de interesse do usuário para localizar e prover informações semelhantes.

3.4 Requisitos não Funcionais do *Fidus*

O projeto do *Fidus* assumiu compromisso com a eficiência, a eficácia, e a escalabilidade, contempladas nos seguintes requisitos não funcionais:

- ✓ **Capacidade de Evolução**

  O *Fidus* possui módulos independentes que implementam suas principais funcionalidades, permitindo o simples acréscimo e/ou substituição de módulos para garantir sua evolução.

- ✓ **Segurança**

  A privacidade dos dados armazenados no perfil é preservada através de mecanismo de identificação e autenticação dos usuários. Cada usuário deve acessar apenas as informações de seu próprio perfil.

- ✓ **Escalabilidade**

  O *Fidus*, para manipular um alto volume de dados e usuários, adota uma implementação com capacidade de evolução no número de recursos disponibilizáveis e manipuláveis.

- ✓ **Usabilidade**

  Um dos mais importantes requisitos do *Fidus* é o de garantir a seus usuários a máxima simplicidade e facilidade no uso de seus recursos, maximizando a efetividade no uso do sistema.

3.4 Arquitetura do *Fidus*

Distingue-se na Web dois tipos de ferramenta de busca de informação. O primeiro tipo corresponde às ferramentas de busca genérica, como *Altavista* e *Google*, que se propõem a prover informações localizáveis em toda a Web, garantindo uma grande abrangência. Essas ferramentas genéricas operam num universo de busca que contém informações heterogêneas e provêem grandes coleções de resultados, com baixa precisão e difícil seleção, pelos usuários, de documentos realmente relevantes.

O segundo tipo corresponde às ferramentas de busca específica, que se propõem a prover informações de domínios especializados e restritos. Essas ferramentas são bastante precisas e normalmente apresentam pequenas coleções de resultados, como o Researchindex[12], da NEC.

A escolha inadequada de uma ferramenta de busca pode dificultar a localização de informações, causada pelo excesso de documentos nos resultados ou pela falta de documentos realmente relevantes. A escolha entre precisão e abrangência na busca pode levar os usuários a utilizarem múltiplas ferramentas de busca, num processo pouco eficaz

Diante deste cenário, projetou-se a arquitetura do *Fidus* para suportar a busca genérica ou especializada, fundada no conceito de meta-busca.

Uma arquitetura de meta-busca descreve uma solução que integra múltiplas ferramentas de busca, genéricas ou específicas, para prover o acesso unificado às informações [MEN 9?]. Essa arquitetura simplificada é apresentada na Figura 3.1.

[12] www.researchindex.com

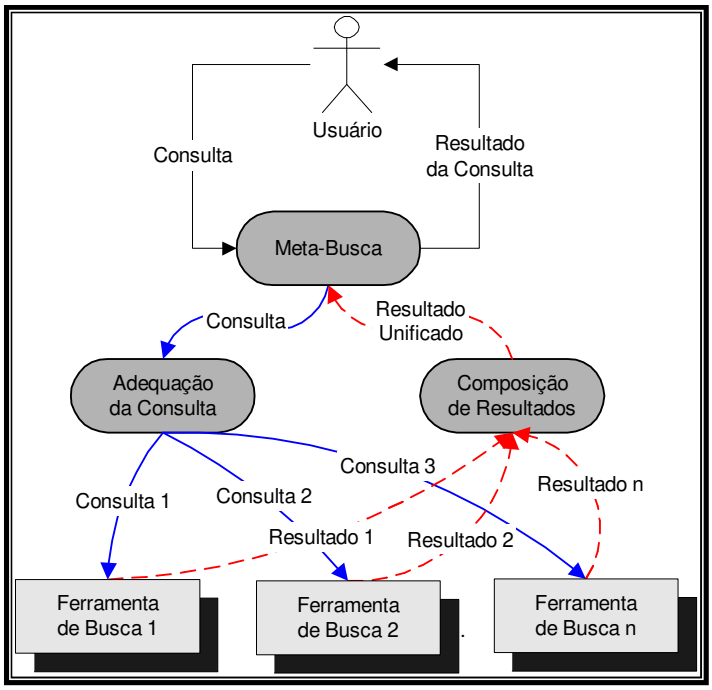


Figura 3.1: Arquitetura simplificada de meta-busca.

A *consulta* elaborada para a meta-busca deve ser composta seguindo seu modelo próprio de representação. O processo de *adequação da consulta* é responsável pela conversão dessa consulta para o modelo adotado pela ferramenta de busca para a qual a busca efetiva será delegada.

O processo de *composição de resultados* é responsável pela integração dos múltiplos resultados oriundos das ferramentas de busca, eliminando informações duplicadas.

A relevância potencial das informações recuperadas, calculada pelas ferramentas de busca, deve ser descartada. Ferramentas de busca distintas possuem diferentes algoritmos para o cálculo dessa relevância. O processo de *meta-busca* deve centralizar a análise de relevância dos documentos recuperados, para determinar os valores de relevância potencial baseados em seu algoritmo.

O universo de busca na arquitetura de meta-busca corresponde à união do universo de busca de todas as ferramentas de busca integradas.

[MEN 9?] propõe uma arquitetura de meta-busca mais refinada, apresentada na Figura 3.2.

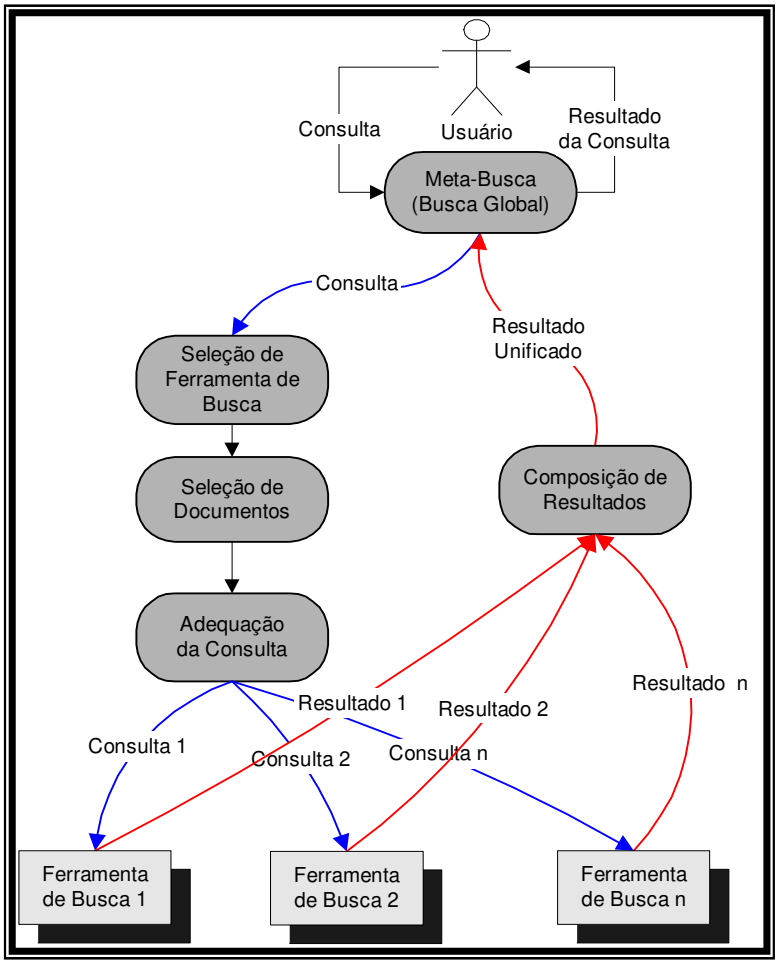


Figura 3.2: Arquitetura refinada de meta-busca.

A expansão do universo de busca, nessa arquitetura de meta-busca, pode ser alcançada através da adição de novas ferramentas de busca. Embora isso aumente a abrangência da meta-busca, sua eficiência será comprometida, pois cada nova consulta deverá ser propagada a um número maior de ferramentas de busca. Também, o conjunto de resultados a compor se tornará muito extenso e o número de informações irrelevantes crescerá.

A solução desse problema é responsabilidade do processo de *seleção de ferramenta de busca,* determinando quais são as *ferramentas de busca* mais precisas para atender à *consulta*. Precisa-se determinar o contexto da consulta e qual a precisão de cada ferramenta de busca para esse contexto, isso é, quais ferramentas de busca retornam mais informações

determinadas como relevantes pelo *processo central de análise de relevância* da meta-busca.

Caso a consulta pertença ao $contexto_1$, e as ferramentas de busca 1 e 2 retornem em seus resultados um percentual maior de informações relevantes que as ferramentas 3 e 4 para atender à consulta, o processo de *seleção de ferramenta de busca* deve direcionar a consulta para as ferramentas 1 e 2, maximizando a eficiência da meta-busca.

A gerência dos resultados recuperados pelas ferramentas de busca permite que o *Fidus* analise sua precisão individual e proceda, em interações futuras, com o processo de *seleção de ferramenta de busca.*

O processo de *composição de resultados* descarta as informações de baixa relevância potencial recuperadas pelas ferramentas de busca.

Ferramentas de busca especializadas no contexto da consulta, como ferramenta de busca em biblioteca digital de medicina para consultas de temas médicos, são mais efetivas, nesse caso, que ferramentas de busca genérica como *Google* e *Altavista*. Caso o processo de *seleção de ferramenta de busca* selecione essas ferramentas na meta-busca, o processo de *seleção de documentos* deve priorizar os resultados providos pelas ferramentas de busca mais precisas.

[FRE 99] e [MEN 9?] afirmam que a meta-busca deve capturar a consulta do usuário, enviá-la a cada sistema de busca cliente, coletar os resultados e apresentá-los aos usuários numa forma inteligente, ordenando-os por relevância potencial e eliminando os resultados de baixa relevância. E este é um dos principais objetivos do *Fidus*, a composição inteligente de resultados.

A Figura 3.3 apresenta a arquitetura do *Fidus* com seus principais processos. Esta arquitetura possibilita a busca de informações sobre todos os contextos disponíveis no *universo de busca*, bem como sua manipulação e análise local. O universo de busca visível ao *Fidus* é determinado pelos seus componentes de busca, que são responsáveis pela localização e extração de informações.

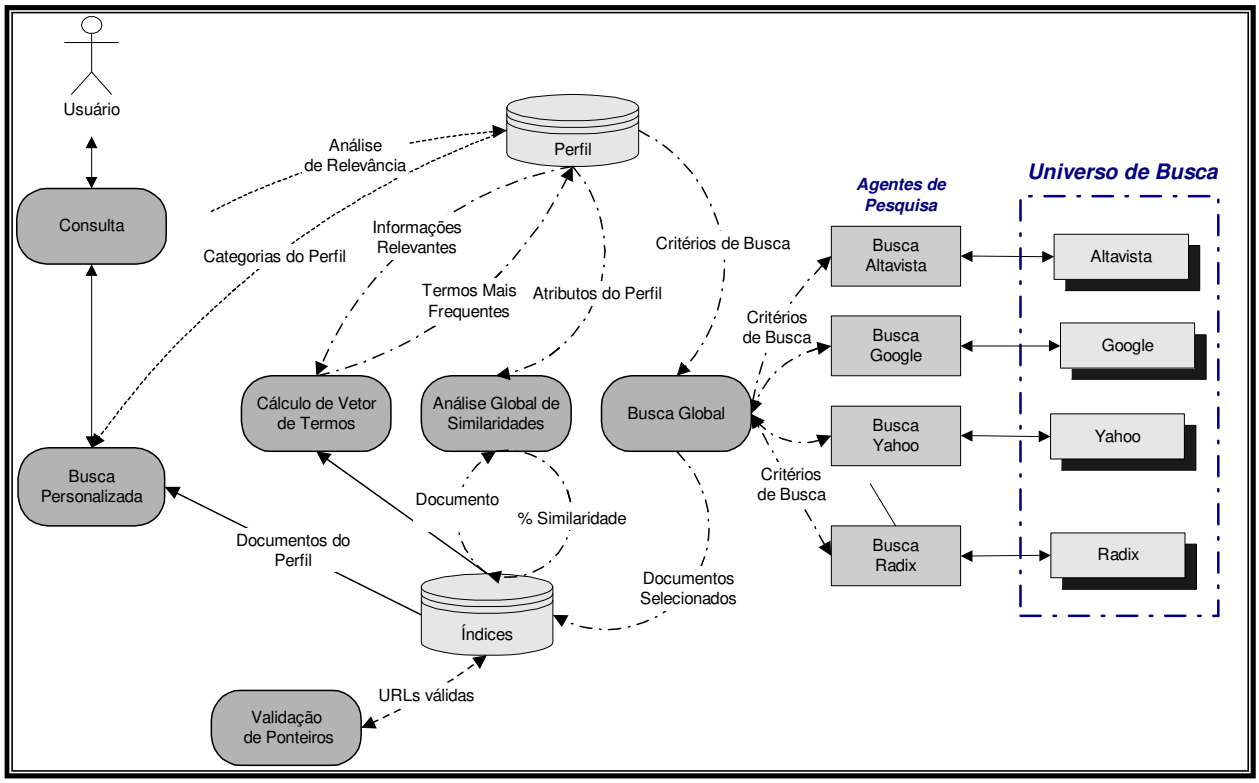


Figura 3.3: Modelo arquitetural do *Fidus*.

A busca de informação para atender às necessidades do usuário é coordenada pelo processo de *busca global*. Esse processo extrai, do *perfil* do usuário, a representação de sua necessidade de informação - coleção de critérios de busca – e solicita a todos os *agentes de pesquisa* – componentes responsáveis pela interação com as ferramentas de busca - que localizem e recuperem as informações que satisfaçam tais critérios.

O *universo de busca* do *Fidus* é integrado por múltiplas ferramentas de buscas, como *Altavista, Google, Radix*, *Lycos*, *Yahoo*, etc. Essas ferramentas são chamadas de provedores de informação do *Fidus*.

Existe um *agente de pesquisa* para cada provedor, pois tais agentes são especializados em interagir com estes provedores, adequando a consulta a seu modelo particular e recuperando seus resultados.

A *busca global* deve integrar todos os resultados obtidos pelos *agentes de pesquisa*, compondo um resultado único e consistente. Os documentos deste resultado serão

armazenados e representados como vetores de termos. Caso uma das ferramentas de busca utilizadas pelo *Fidus*, como o *Google*, retorne 20.000 documentos para uma consulta, apenas os 50 primeiro documentos ainda não armazenados o serão. Juntamente com esses documentos, serão armazenados os ponteiros para as próximas 20 páginas de resultados. Dessa forma, o *Fidus* pode proceder a recursão de busca posteriormente sem estressar seu o *cache*.

O processo *cálculo de vetor de termos* cria um vetor de termos para representar cada documento [SAL 83]. Detalharemos o algoritmo que cria esse vetor na seção 4.3.3 do capítulo 4. Como o *Fidus* não possui um dicionário controlado para a redução de termos, apenas os termos encontrados no documento, após a redução para radicais comuns, são inseridos nesse vetor.

Inconsistências geradas pela inexistência da informação ou por apontador inválido para a informação são resolvidos pelo *componente de validação de ponteiros,* detalhado na seção 4.3.3 do capítulo 4.

A *análise global de similaridades*, com base nos termos mais freqüentes dispostos nos vetores, deve determinar qual o percentual de similaridade entre o documento recuperado e o contexto para o qual ele foi selecionado. A relevância potencial sugerida pelos provedores é descartada, pois um novo cálculo precisa ser efetuado para todas as informações localizadas.

Descrevemos o conceito de similaridade entre documento e contexto na seção 4.3.3 do capítulo 4, onde detalhamos o algoritmo de cálculo de similaridades.

O processo *Busca Personalizada* do *Fidus* estrutura os dados do perfil para que a interface de consulta do usuário seja simplificada.

3.5 Uma Sessão do Fidus

Uma sessão do *Fidus* é iniciada com o usuário compondo seu perfil como consumidor de informação. Um *perfil* consiste numa árvore, que representa a hierarquia de tipos de informação que interessam ao usuário. Cada nó da árvore representa uma categoria de informação.

Para auxiliar os usuários na composição do perfil, o *Fidus* disponibiliza uma árvore que exemplifica a classificação de informação. Os nós dessa árvore podem ser copiados para a árvore do perfil do usuário, pelo mesmo, caso a categoria de informação representada no nó o interesse.

Na Figura 3.4, a árvore da esquerda corresponde à classificação de informação sugerida pelo *Fidus*. Essa árvore é montada pelo administrador do *Fidus* e serve para nortear a construção inicial do perfil.

A árvore da direita corresponde à estrutura do perfil de um usuário. Para sua construção, o usuário copiou o nó *Banco de Dados* da árvore da esquerda. Essa cópia transfere toda a estrutura inferior ao nó, como os nós *Relacional*, *Objeto Relacional* e *OO*.

As opções de cópia permitem que um nó seja copiado para a árvore da direita como um novo nó raiz, caso dos nós *Banco de Dados* e *VB*, ou como um novo nó folha.

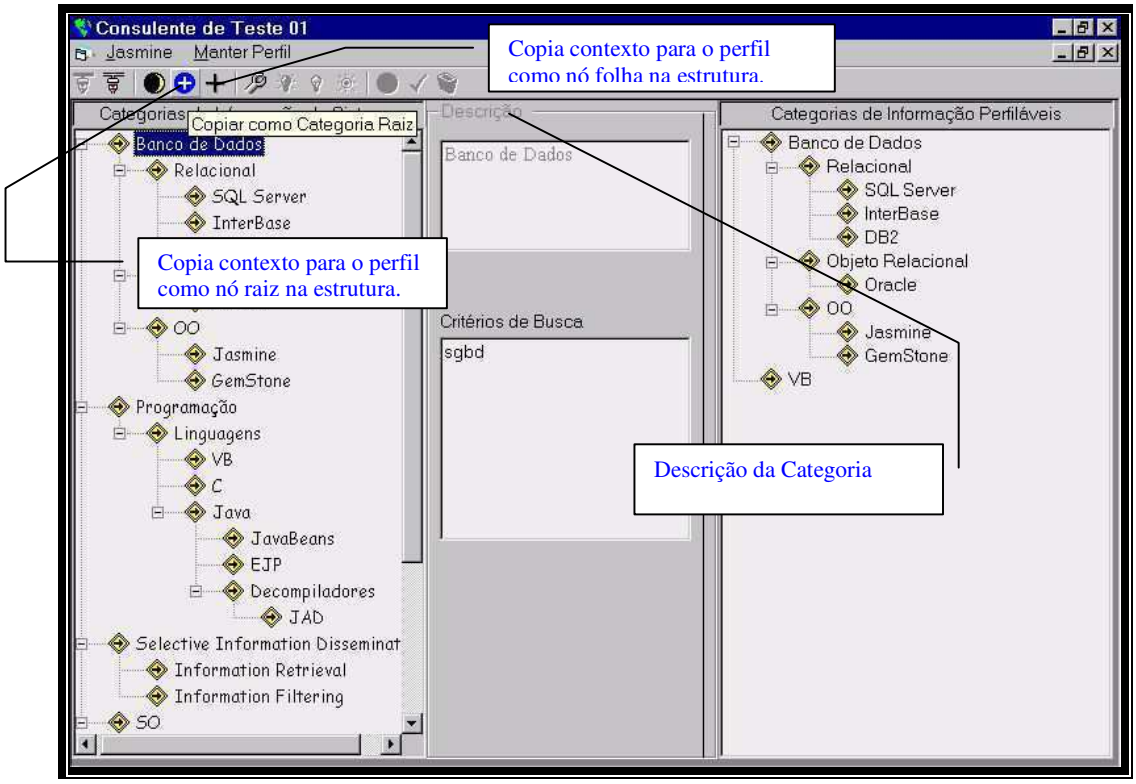


Figura 3.4: Exemplo de classificação de informações.

No exemplo da figura, *Banco de Dados* representa um contexto ou categoria de informação. A seleção da categoria apresenta, nas caixas de texto do centro, sua *descrição* – Banco de Dados - e seus *critérios de busca* –sgbd - e disponibiliza as operações de cópia. Cada categoria pode possuir subcategorias, como *Relacional* e *OO*, diminuindo a granularidade de sua representação.

Critérios de busca são termos que descrevem os documentos do contexto, sendo usados, pelos *agentes de pesquisa*, para localizá-los. As buscas pelos documentos são tão precisas quanto forem os termos usados para descrevê-los.

Para atender ao requisito do *Fidus* de *personalização de informações*, esta estrutura proposta inicialmente deve servir apenas como orientação e sugestão ao usuário. Cada usuário deve poder construir sua própria classificação de informação e alterar os critérios de busca para refinar o contexto desejado, como apresentado na Figura 3.5.

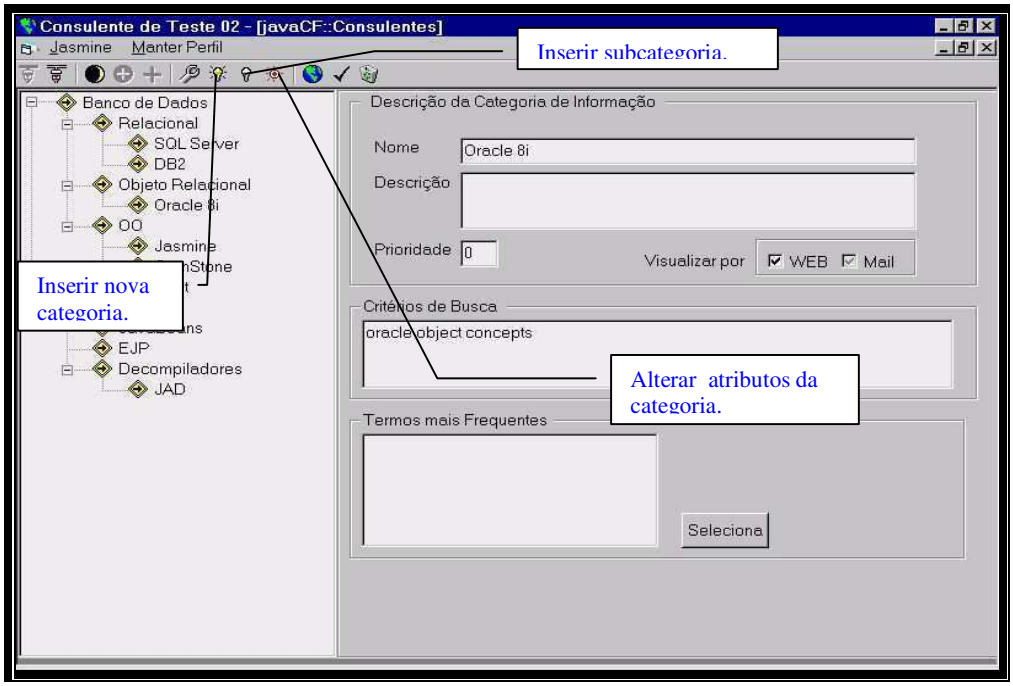


Figura 3.5: Manutenção do perfil.

Cada categoria ou subcategoria de informação possui atributos personalizados, que são:

- ***Nome***:
  - Nome dado ao contexto da informação ou tema de pesquisa bibliográfica.
- ***Descrição***.
  - Descrição da categoria.
- ***Prioridade***:
  - Valor que determina a ordem de classificação das categorias de informação na apresentação de resultados.
- ***Visualização***:
  - Forma de apresentação das informações, que determina como os resultados da consulta serão apresentados - via Web, Mail, etc -.
- ***Critérios de busca***:
  - Palavras-chave que norteiam os *agentes de pesquisa* na localização de informações condizentes e relevantes à categoria.
- ***Termos mais freqüentes***:

- Termos mais freqüentes na coleção de documentos encontrados para a categoria.

Após a construção da árvore que representa o perfil, o *Fidus* pode proceder à busca de informações. Para isto, o *componente global de busca* deve extrair os *critérios de busca* de todas as categorias – nós da árvore - de cada perfil para efetuar a meta-busca. Em seguida, os *agentes de pesquisa* devem localizar as informações potencialmente relevantes, cabendo ao *componente global de busca* a integração dos resultados numa coleção única.

A apresentação desta coleção única de resultados é demonstrada na Figura 3.6:

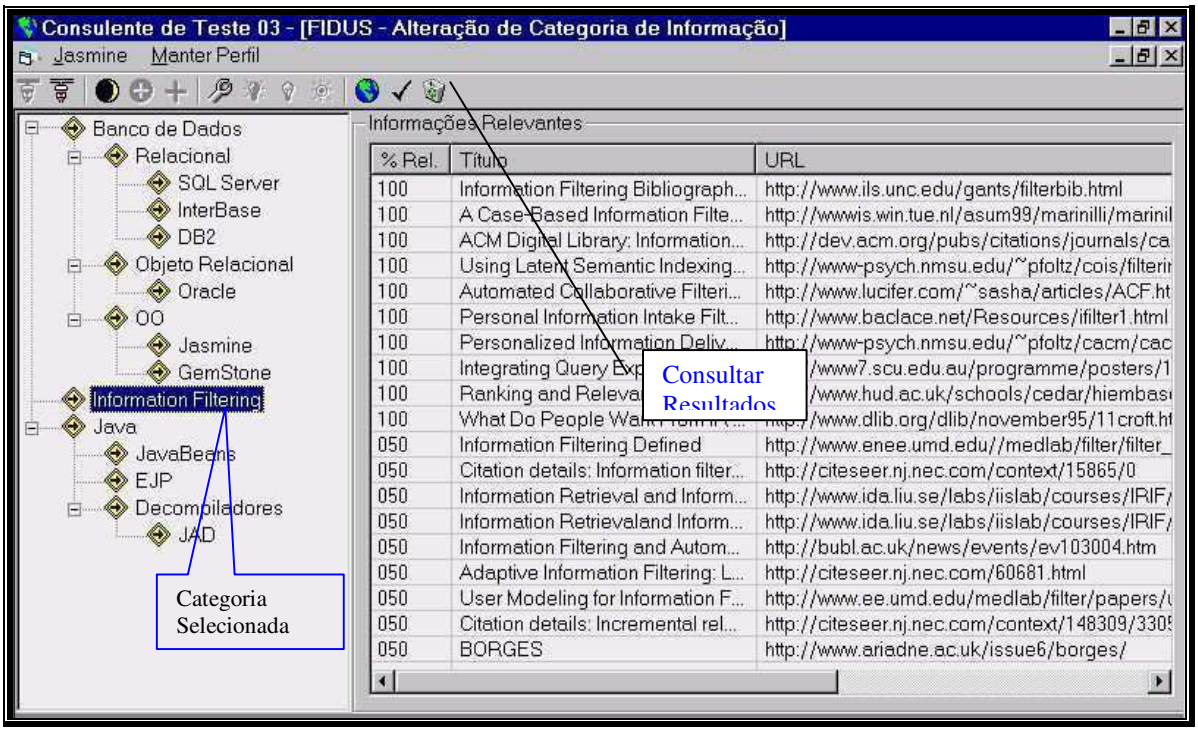


Figura 3.6: Apresentação da coleção de resultados.

A consulta aos resultados da meta-busca do *Fidus* é iniciada com a seleção de uma categoria de informação. No exemplo da figura, a categoria selecionada *é Information Filtering*.

Na tabela da direita são apresentadas todas as informações potencialmente relevantes recuperadas.

A tabela de resultados possui três colunas:

1. % Rel – potencial de relevância da informação para atender aos critérios do usuário. A probabilidade da informação ser realmente relevante ao usuário e diretamente proporcional a esse valor. Seu cálculo será detalhado no capítulo 4.
2. Título – título que descreve a informação.
3. URL – URL onde o conteúdo do documento pode ser encontrado.

Caso uma das informações apresentadas na tabela da direita seja selecionada, o *Fidus* apresenta seu conteúdo numa tela similar a um "browser", como apresentado na Figura 3.7, com botões que permitam ao usuário julgar sua relevância.

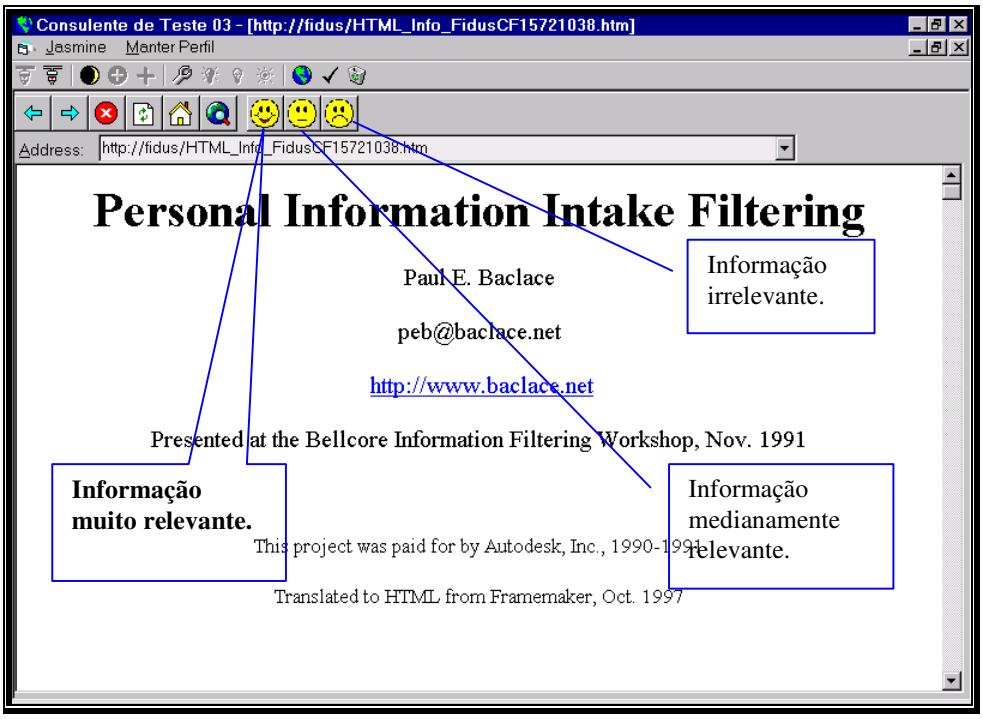


Figura 3.7: Visualização e julgamento de informações selecionadas

A visualização do conteúdo da informação selecionada permite que sua relevância efetiva seja julgada pelo usuário.

A interface do *Fidus* permite a distinção de três níveis de relevância, permitindo que o usuário julgue cada informação como *muito relevante*, *medianamente relevante* ou *irrelevante*. O grau de relevância será usado pelo *Fidus* para computar o peso dos termos nos documentos selecionados. O algoritmo de cálculo de peso dos termos é detalhado na seção 4.3.3 do capítulo 4.

A consulta a informações julgadas como relevantes é apresentada na Figura 3.8. Para a visualização do resultado da consulta, seleciona-se a categoria desejada na árvore da esquerda.

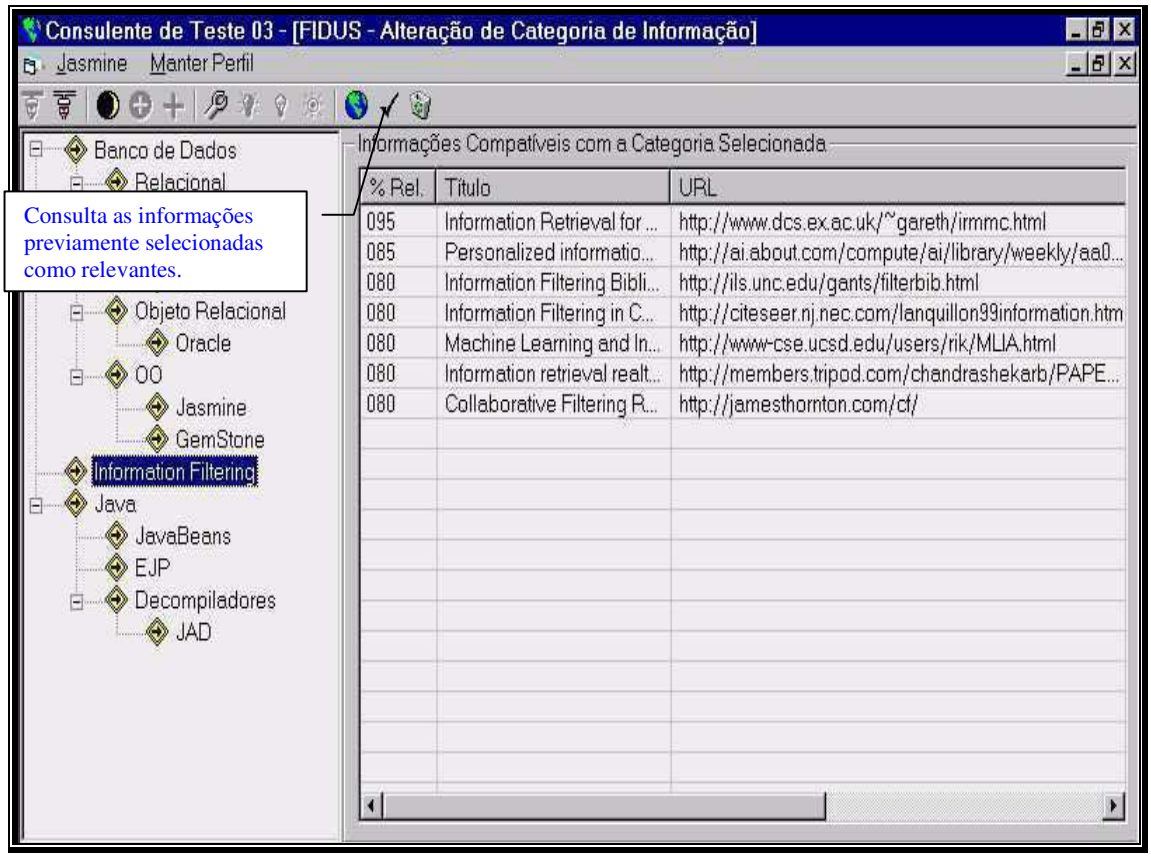


Figura 3.8: Consulta a informações relevantes.

Os resultados sempre são ordenados decrescentemente pelo potencial de relevância, simplificando a consulta. A seleção de uma informação, na tabela da direita, permite a visualização de seu conteúdo e o julgamento de relevância.

As informações descartadas são apresentadas na Figura 3.9.

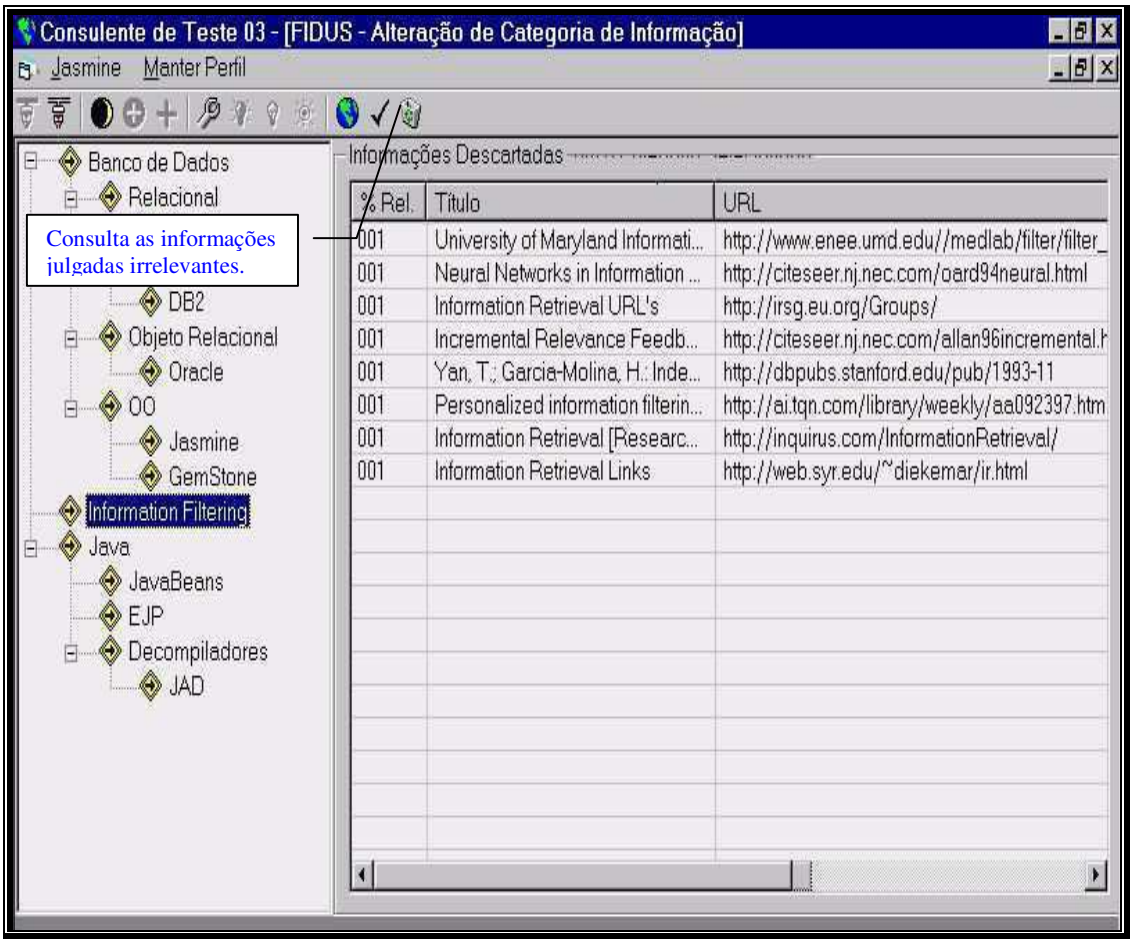


Figura 3.9: Informações descartadas.

Os termos de maior peso – calculado por algoritmo similar ao TF-IDF - no vetor que representa o documento, por identificam seu contexto, são significativos na elaboração dos critérios de busca. O cálculo do peso dos termos é detalhado na seção 4.3.3 do capítulo 4.

O *Fidus* apresenta ao usuário a lista dos termos de maior peso para cada categoria como auxílio na elaboração dos critérios de busca utilizados na meta-busca. A Figura 3.10 apresenta os critérios de busca da categoria selecionada – *Information Filtering* – e a lista dos termos de maior peso.

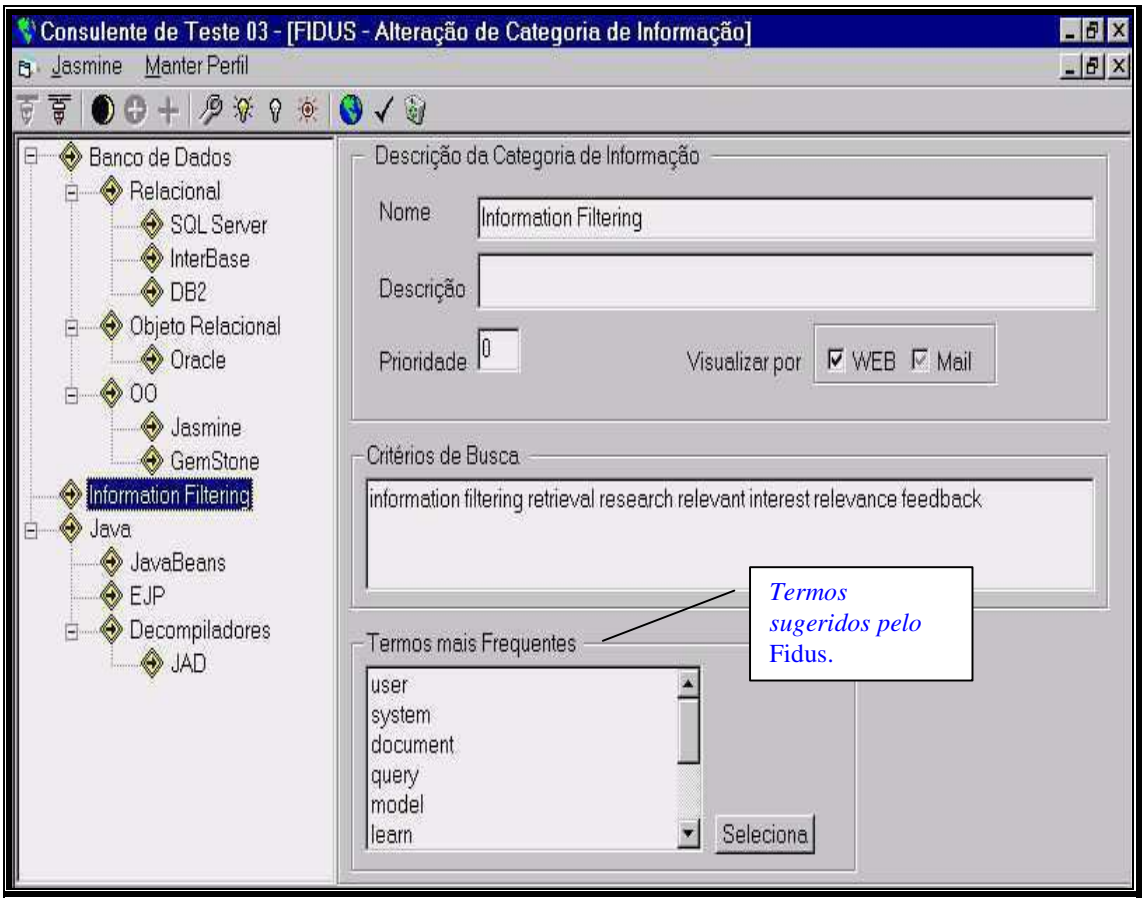


Figura 3.10: Sugestão de termos para os critérios de busca.

A lista de *termos mais freqüentes* é construída pelo *Fidus* e apenas os termos da lista que não existem nos critérios de busca são apresentados.

A composição dos *critérios de busca* é feita pela seleção de um termo, incluído-o nos *critérios de busca* da categoria, ou por alteração manual.

3.6 Conclusão

O *Fidus*, como uma proposta inovadora de ferramenta de busca personalizada, incorpora todos os requisitos de um verdadeiro SIP.

Como inovação, provê a busca de informações através de perfis estruturados e configuráveis pelos usuários e a apresentação de resultados ordenados pela relevância potencial. Sua precisão na busca de informação é ajustável em função da interação com o usuário para o julgamento da relevância efetiva das informações.

A arquitetura de meta-busca do *Fidus* permite integrar um universo de busca heterogêneo, distribuído e em expansão. Um amplo espectro de interesse de informação pode ser atendido por esta arquitetura.

A modularização da arquitetura do *Fidus* permite que seus componentes sejam desenvolvidos separadamente e que novos componentes sejam agregados, garantindo alto poder de evolução.

Para simplificar o processo de busca, a interface do *Fidus* foi projetada para ser bastante simples e intuitiva.

Capítulo 4

Desenvolvimento do Fidus

4.1 Introdução

Para o desenvolvimento do *Fidus*, foi utilizado um processo simplificado de desenvolvimento de software, iterativo e incremental, e a linguagem de modelagem UML. Diversos incrementos do *Fidus* foram produzidos, como protótipos, para validar suas funcionalidades.

Detalhamos, neste capítulo, o modelo de projeto do *Fidus* e todos os detalhes de sua implementação.

4.2 Modelo de Projeto do Fidus

O modelo de projeto do *Fidus* especifica uma solução computacional que melhor atende a seus requisitos, apresentando as principais abstrações – classes - do domínio do problema, seus relacionamentos e atributos.

Classes do Projeto

As principais classes identificadas na fase de projeto do *Fidus* são:

- ***Usuário***
  - Pessoa habilitada a utilizar o Fidus.
- ***Administrador***
  - Usuário habilitado a administrar o *Fidus*, como apresentado na seção 3.3 do capítulo 3.
- ***Consulente***
  - Usuário habilitado a consultar o *Fidus* e a compor um *Perfil*.
- ***Perfil***
  - Representação das características do consulente como consumidor de informação.
- ***Categoria de Informação***
  - Classificação da informação por tipo, determinados seu contexto e atributos.
- ***Categoria Perfilável***
  - Especialização da *Categoria de Informação*.
  - Permite que o consulente construa sua hierarquia particular de categorias, criando sua árvore de contextos de interesse – perfil.
- ***Informação***
  - Informação encontrada no universo de busca e potencialmente relevante ao perfil de um consulente.
- ***Termo Freqüente***
  - Termo encontrado num documento com seus respectivos atributos de freqüência e peso.

Diagrama de Classes

A Figura 4.1 apresenta o diagrama das classes que compõem o projeto do *Fidus* e seus relacionamentos, utilizando a notação UML.

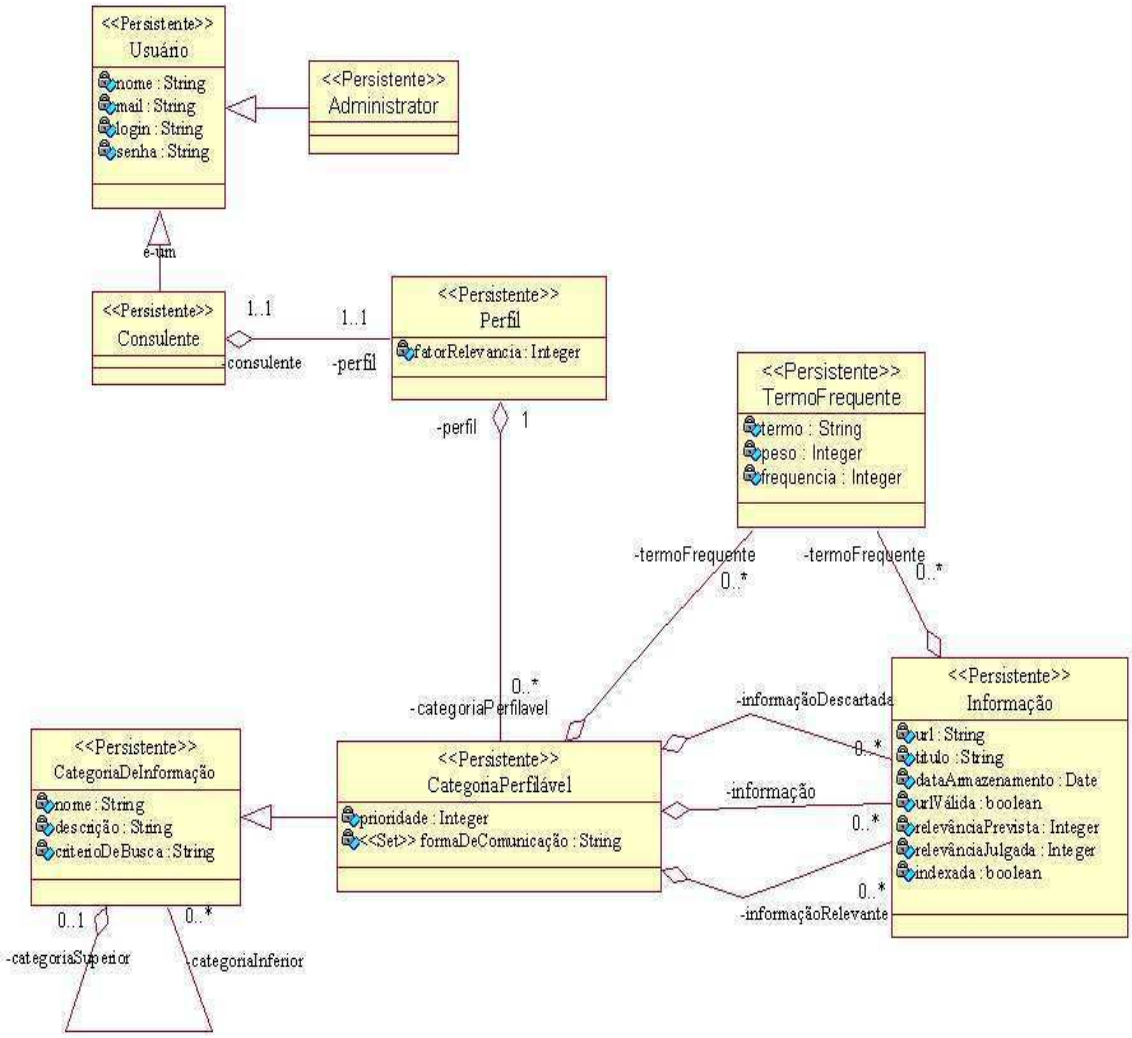


Figura 4.1: Diagrama de classes do Fidus.

A classe *Usuário* representa o usuário cadastrado no *Fidus*. Possui duas especializações: *Consulente e Administrador.*

O *consulente* possui um único *perfil*, que o caracteriza como consumidor de informação. O atributo *fatorRelevancia* da classe *Perfil* determina o percentual mínimo de

relevância potencial para que uma informação seja apresentada como resultado de uma consulta.

A classe *CategoriaDeInformação*, com seus relacionamentos de agregação, representa a estrutura hierárquica de categorias utilizada para estruturar e classificar as informações de interesse do usuário. Cada categoria representa um tema da pesquisa bibliográfica que interessa ao usuário. Para essa classe, o atributo *categoriaSuperior* determina a categoria imediatamente superior na hierarquia. O atributo coleção *categoriaInferior* determina as categorias inferiores na hierarquia.

A classe *CategoriaPerfilável* é uma especialização da classe *CategoriaDeInformação,* permitindo ao consulente atribuir prioridade e forma de comunicação a cada categoria perfilável. Essa prioridade determina uma ordem possível de classificação das informações da categoria para apresentação ao consulente. A forma de comunicação determina se a apresentação das informações será via Web, mail, etc.

Cada *CategoriaPerfilável* agrega objetos do tipo *TermoFrequente*, que representa o vetor de termos da categoria com seus respectivos pesos. Definimos o vetor de termos da categoria na seção 4.3.3.

Cada *Perfil* possui uma coleção de *CategoriaPerfilável,* compondo a hierarquia de categorias – árvore de categorias - particular ao consulente.

A informação localizada pelo *Fidus* na meta-busca é tornada persistente como instância da classe *Informação*. Cada informação possui como atributos:

1. URL - informa sua localização na Web.
2. Título – título da informação.
3. dataArmazenamento - representa a data de localização da informação pela meta-busca.
4. relevânciaPrevista – relevância potencial da informação para a categoriaPerfilável correspondente. É calculada pelo *Fidus* e depende da categoria à qual a informação está associada Esse cálculo é detalhado na seção 4.3.3.
5. relevânciaJulgada - relevância efetiva da informação informada pelo usuário.

4.3 Implementação

O *Fidus* foi implementado utilizando-se uma arquitetura de 3 camadas. A Figura 4.2 apresenta os elementos que compõem cada camada:

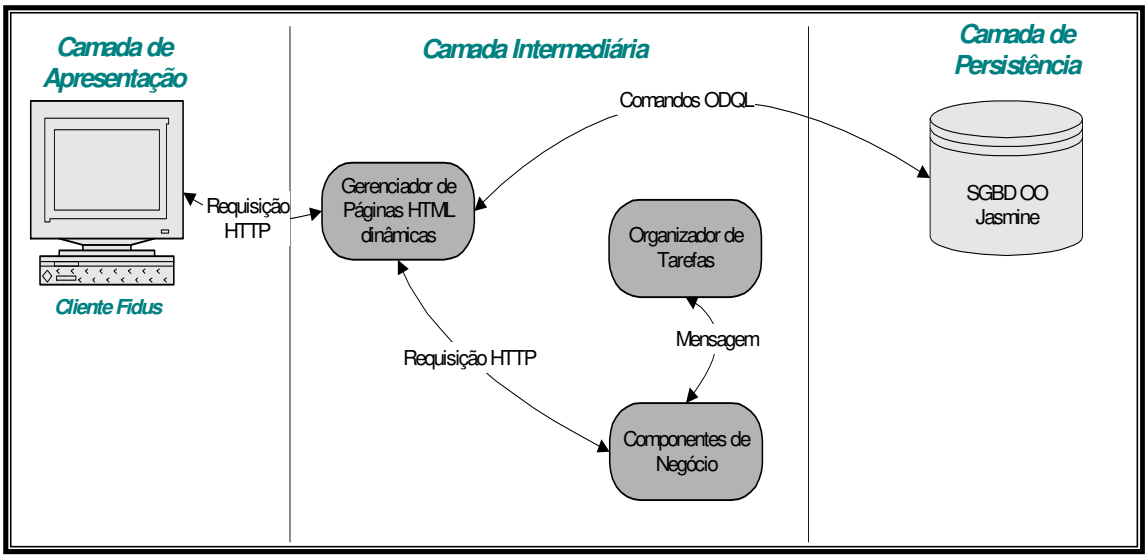


Figura 4.2: Camadas da arquitetura do Fidus.

A *camada de apresentação* da arquitetura do *Fidus* é composta por uma interface, denominada *Cliente Fidus,* desenvolvida em Visual Basic, com a qual o usuário interage para compor seu perfil de consumidor de informação e para consultar e julgar as informações recuperadas pelo *Fidus*.

Como o *Cliente Fidus* foi projetado para manipular requisições HTTP para comunicar-se com a camada intermediária, o desenvolvimento de uma nova interface que a substitua é bastante simples. Uma nova interface para a Web, por exemplo, só precisa conhecer as requisições HTTP que provêem todos os recursos do *Fidus*.

A *camada intermediária* é responsável pelo processamento da lógica do sistema e pela gerência do acesso a dados.

4.3.1 Gerente de Páginas Dinâmicas

O *Cliente Fidus* manipula os objetos persistentes da camada de persistência através de requisições HTTP ao *gerente de páginas HTML dinâmicas*. Esse gerente disponibiliza páginas HTML dinâmicas que recebem parâmetros e retornam os resultados requeridos. O acesso efetivo aos dados é feito através do envio de comandos ODQL[13], embutidos nas páginas dinâmicas, ao SGBD OO *Jasmine* [KHO 99].

Para viabilizar esse processo, utilizamos o WebLink [KHO 99], uma ferramenta que permite a construção de páginas HTML dinâmicas com capacidade de manipulação de objetos armazenados no *Jasmine*. Detalhamos o WebLink nessa seção devido a sua grande importância para o desenvolvimento do *Fidus*.

WebLink

A dinâmica das páginas HTML gerenciadas pelo WebLink é construída através de *scripts*, que são conjuntos de comandos ODQL embutidos nessas páginas.

Descrevemos, a seguir, um exemplo de página HTML com *script* WebLink:

```
[01] <!DO "defaultCF FidusCF">
[02] <!VAR "CategoriaDeInformacao set" sCat>
[03] <!VAR "CategoriaDeInformacao" cat>
[04] <!VAR String cOID>

[05] <html> <head> <title> Retornar a coleção de Categorias de
Informação </title> </head>

[06] <!DO "sCat = select c from CategoriaDeInformacao alone c">

[07] <!FOREACH sCat cat>
[08]    [Nome:] [<!REPLACE cat.nome>] <BR>
[09]    [Descrição:] [<!REPLACE cat.descricao>] <BR>
[10]    [Critérios:] [<!REPLACE cat.criterioDeBusca>] <BR>
        ...
[11] <!/FOREACH>

[12] </body> </html>
```

[13] Linguagem de definição e manipulação de objetos do Jasmine, detalhada no Apêndice A.

Nesse exemplo, o *script* correspondendo aos comandos ODQL está destacado em azul. A parte estática da página HTML está destacada em verde.

A linha 01 define qual *família de classes* o *script* pode manipular. Uma família de classes representa uma coleção de classes correlatas e assemelha-se ao conceito de "*database*" nos bancos relacionais mais conhecidos, como SQL Server.

As linhas 02, 03 e 04 definem variáveis manipuláveis pelo *script*. A variável *sCat* representa uma coleção – *Set* - de objetos do tipo *CategoriaDeInformacao.* A variável *cat* referencia objetos desse tipo e variável *cOID* referencia objetos do tipo *String*.

A Linha 05 corresponde à parte estática da página HTML, com seus *tags* de início e título da página.

Na linha 6, um comando de seleção de objetos é efetuado para a população da coleção representada por *sCat*. As linhas 07 a 11 varrem a coleção *sCat*, extraindo seus elementos e referenciando-os através da variável *cat*. As linhas 08, 09 e 10 combinam a parte estática do HTML com a parte dinâmica do *script*. O comando REPLACE, do *script*, altera a página estática inserindo *strings* recuperadas do Jasmine.

A interpretação dessa página HTML dinâmica, pelo WebLink, produz uma página HTML estática no seguinte formato:

```
<html> <head> <title> Retornar a coleção de Categorias de Informação
</title>
</head>

[Nome:] nome 01 <BR>
[Descrição:] descrição 01 <BR>
[Critérios:] critérios 01 <BR>
[Nome:] nome 02 <BR>
[Descrição:] descrição 02 <BR>
[Critérios:] critérios 02 <BR>
        ...
</body> </html>
```

Todos os comandos do *script* foram interpretados para produção de uma página puramente HTML. As linhas destacadas em negrito apresentam dados recuperados do Jasmine pelas linhas 7 a 10 do *script*.

O processo executado pelo WebLink para a interpretar um *script* e produzir uma nova página HTML estática é mostrado na Figura 4.3:

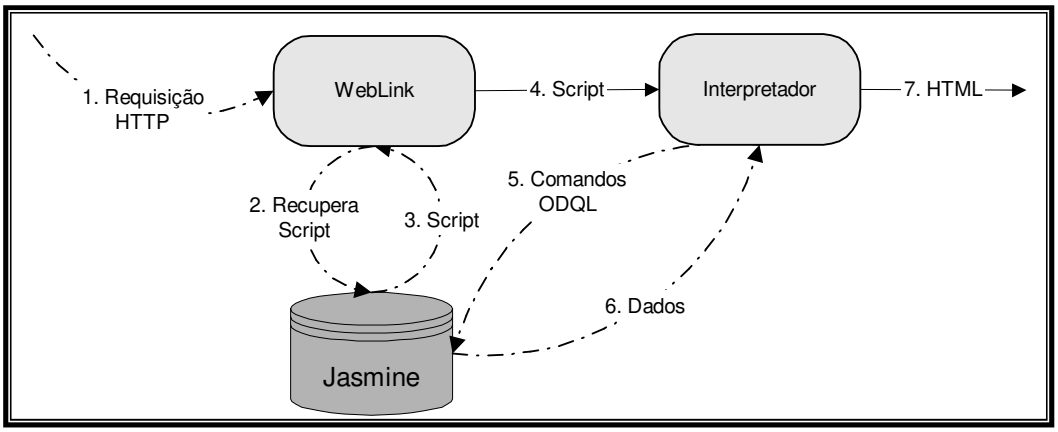


Figura 4.3: Interpretação de um script WebLink.

O passo 1 corresponde à referência a um *script* gerenciado pelo WebLink. Todo *script* WebLink é armazenado no Jasmine.

Essa referência a um *script* possui o seguinte formato: *http://endereço_ip/cgi-bin/odb-get.exe?WIT_template=nome_script[&parametro1=valor1&...&parâmetrox =valorx]*". Os parênteses informam os argumentos opcionais.

Endereço_ip corresponde ao endereço IP da máquina onde o WebLink está instalado. Odb-get.exe é um processo que recebe como parâmetro *WIT_tem*plate o nome do *script.* Parâmetro_x é o nome do parâmetro passado ao script e valor_x, o valor desse parãmetro.

O WebLink recupera, no Jasmine, o conteúdo do *script* referenciado - passos 2 e 3 -, e o envia a seu interpretador - passo 4 – com os respectivos parâmetros. Os comandos ODQL, contidos no *script*, são enviados ao Jasmine - passo 5. Após a execução desses comandos, os dados são formatados numa página HTML estática, que é provida como resultado - passo 7.

4.3.2 Componentes de Negócio

Os *componentes de negócio* executam as funcionalidades do sistema. São responsáveis por tarefas como a busca especializada de dados, a elaboração de vetores de termos, o cálculo de pesos e a análise de similaridades, como já apresentados nos capítulos 2 e 3.

Os componentes desenvolvidos para o *Fidus* utilizam a tecnologia de componentes COM - Component Object Model -, da Microsoft [COM 95], e foram implementados com as linguagens Visual Basic e C/C++. Os principais componente de negócio são:

Componente Global de Busca

O *Componente Global de Busca* implementa a *Busca Global* da arquitetura do *Fidus.*

Suas principais responsabilidades são:

1. Selecionar os critérios de busca das categorias dos *perfis* representados no *Fidus*.
2. Enviar os critérios de busca aos *agentes de pesquisa* - definidos na seção 3.4 do capítulo 3 - para que estes localizem informações potencialmente relevantes.
3. Integrar os resultados dos *agentes de pesquisa*, compondo um resultado único e consistente.

Não foi implementado, na versão atual do *Fidus*, o processo de *seleção de ferramenta de busca,* apresentado na seção 3.4 do capítulo 3. A busca global envia a requisição de busca a todos os *agentes de pesquisa* que compõem o *Fidus* para interagir com as ferramentas de busca.

A proposta do projeto do *Fidus* é que todas as ferramentas de busca que compõem seu universo de busca sejam apresentadas ao usuário, na composição do perfil, para que o mesmo escolha as ferramentas utilizadas na meta-busca para cada categoria. Dessa forma, o usuário pode determinar o uso das ferramenta *x* e *y* na categoria$_1$ e das ferramentas *y* e *w* na categoria$_2$.

Componente de Busca

Os *componentes de busca* do *Fidus* implementam as funcionalidades *dos agentes de pesquisa* e interagem com ferramentas de busca de informação para recuperar informações.

Cada ferramenta de busca de informação possui uma interface própria, atende a uma requisição HTTP específica e, normalmente, estruturam seus resultados de forma particular. Como exemplo, o Altavista atende a requisições no formato: http://www.altavista.com/sites/search/web?q=”*critérios de busca”*. Seus resultados aparecem numa página HTML particularmente estruturada. Por sua vez, o Radix atende a requisições no formato: *http://www.radix.com.br/busca/consultaBR?exp=”critérios de busca”*. Sua página de resultados possui estrutura diferente das demais ferramentas de busca.

A extração dos documentos, ou apontadores aos mesmos, apresentados por essas ferramentas de busca em suas páginas de resultado exige um processamento especializado pelo *componente de busca*, pois cada página de resultado possui uma estrutura diferente.

A Figura 4.4 apresenta o modelo simplificado de funcionamento de um componente especializado de busca:

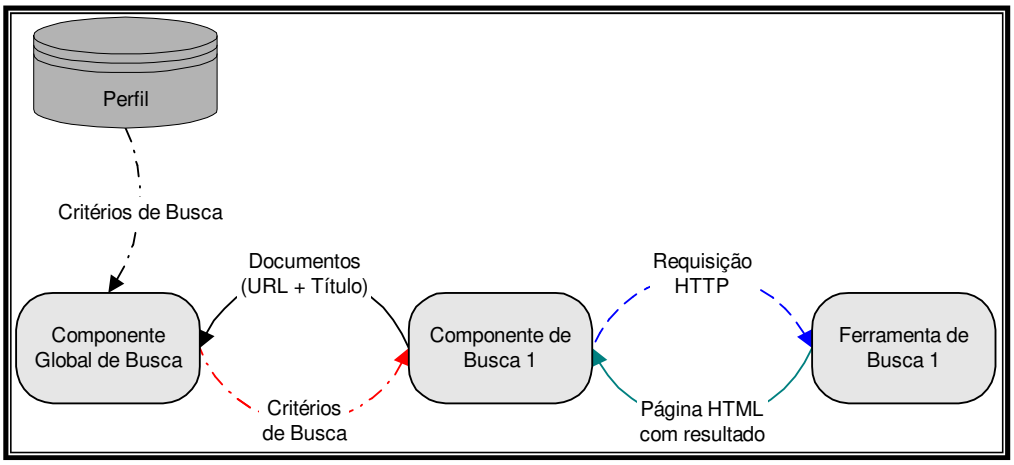


Figura 4.4: Modelo simplificado de um componente especializado de busca.

Nesse modelo simplificado, o *componente global de busca* seleciona os critérios de busca das categorias dos perfis e os passa ao componente de busca 1. Esse componente executa o processo de *adequação da consulta*, da arquitetura de meta-busca apresentada na

seção 3.4 do capítulo 3, para compor a requisição HTTP adequada à ferramenta de busca com a qual interage.

O resultado da ferramenta de busca, uma página HTML com apontadores de informação, é analisado para a extração desses apontadores – URLs - e do título correspondente à informação – normalmente esse título é extraído da descrição dada pela ferramenta de busca ao apontador.

O *componente de busca* retorna ao *componente global de busca* uma lista de documentos representados por sua URL e seu título.

Componente de Representação de Documento

Documentos recuperados pelo *componente global de busca* são representados no *Fidus* como vetores de termos com pesos. Essa representação é detalhada na seção 2.4.4 do capítulo 2.

O processo que o *Fidus* utiliza para a representação de documentos como vetores de termos consiste nos seguintes passos:

1. Recuperar o conteúdo HTML do documento com base em sua URL.
2. Contar o número de ocorrências de cada termo do documento e determinar os mais freqüentes.
3. Eliminar os termos que pertencem à lista de *stopwords*[14] mantida pelo *Fidus*.
4. Reduzir termos para radicais comuns.
    a. O *Fidus* utiliza um processo bastante simplificado de redução de termos, atuando basicamente sobre plurais.
5. Calcular o peso normalizado dos termos.

O *componente de representação de documento* utiliza uma árvore-*B* para representar o vetor de termos de cada documento. Os nós dessa árvore possuem os seguintes atributos:

[14] A lista de *stopwords* do *Fidus,* em sua versão atual, contempla apenas palavras da língua inglesa. Para o correto tratamento de documentos em outras línguas essa lista deve ser estendida.

- ✓ *Termo* – termo reduzido do documento.
- ✓ *DF* - número de documentos relevantes onde o termo ocorreu. É determinado para cada categoria de informação, pois categorias diferentes possuem diferentes coleções de documentos relevantes.
- ✓ *TF* - número de vezes que o termo ocorreu no documento.
- ✓ *TF Normalizado* – normalização do *TF* para equalizar a representação de documentos de tamanhos diferentes. Seu cálculo é detalhado ao final desta seção.
- ✓ *Peso* – peso do termo.
- ✓ *Peso Normalizado* - normalização do *Peso* para equalizar a representação de documentos de tamanhos diferentes. Seu cálculo é detalhado ao final desta seção.

Os atributos *DF* e *TF* são necessários para o cálculo de pesos através do algoritmo TF-IDF, como descrito na seção 2.4.4 do capítulo 2.

Através de simulações efetuadas com o *Fidus*, descobriu-se que o algoritmo TF-IDF não é completamente adequado aos seus requisitos de representação de contextos, pois atribui alto peso a termos que aparecem com baixa freqüência e baixo peso a termos muito freqüentes. Como apresentado na seção 2.4.4 do capítulo 4, o TF-IDF calcula os pesos dos termos através do produto entre o *TF* (freqüência do termo no documento) e o *IDF* (freqüência inversa do termo nos múltiplos documentos do contexto).

O valor do *IDF* é maior para termos que aparecem em menos documentos, e é zero para termos que aparecem em todos os documentos. Nesse caso do valor do IDF ser igual a zero, o valor do peso do termo também será igual a zero, como resultado do produto entre o *TF* e o *IDF*.

A representação de contexto do *Fidus* consiste na elaboração de um vetor de termos para representar a coleção de documentos de uma categoria de informação. O peso de cada termo, nesse vetor, significa seu poder de seletividade na identificação do contexto dessa coleção. Termos que aparecem em todos os documentos de um contexto, para o *Fidus*, são altamente seletivos e devem ter alto peso.

A adequação do peso do termo a seu poder de seletividade requer que o cálculo do peso seja feito tornando-o diretamente proporcional à freqüência do termo - *TF*. Como o TF-IDF não atende a esse requisito, tornando-o proporcional ao *IDF*, implementou-se, para o *Fidus*, uma variação do algoritmo TF-IDF, que chamamos de TF-Seno.

TF-Seno

O TF-Seno é um algoritmo empírico que surgiu da necessidade de normalização de pesos em função de sua freqüência. O uso da função *seno* é devido à mesma retornar apenas valores entre zero e um para ângulos entre zero e noventa graus. Como a normalização de pesos também deve retornar valores entre zero e um, decidiu-se por seu uso.

Sua fórmula é:

TF-Seno = TF Normalizado * Sin(ângulo) * Relevância

ângulo = 90 * (DF / N)

No cálculo, *N* representa o total de documentos da categoria de informação.

O valor de *Relevância* é 1 para os documentos julgados como muito relevantes e ½ para os julgados medianamente relevantes, pelos usuários. Dessa forma, corrige-se a importância do peso do documento em função da relevância do documento analisado.

Para o cálculo do *TF Normalizado*, ajustando-o ao intervalo [0, 1], divide-se a freqüência de cada termo do documento pela freqüência máxima no documento. A mesma lógica é usada no cálculo do peso normalizado. A normalização de pesos é descrita na seção 2.4.4 do capítulo 2.

Para o primeiro documento relevante de uma categoria (N = 1, DF = 1 e relevância = 1) os pesos dos termos serão iguais à sua freqüência normalizada.

Para os demais documentos relevantes de uma categoria, o valor de TF Normalizado é calculado como o valor da freqüência do termo na coleção dos documentos. A fórmula utilizada para seu cálculo é:

1. TF Normalizado Corrente = TF / frequenciaMaxima;
2. Se (DF > 1) ***O termo já possui freqüência anterior.***
3. TF Normalizado = ((TF Normalizado * (DF -1)) +
TF Normalizado Corrente) / DF;
Senão
5. TF Normalizado = TF Normalizado Corrente;

Para cada termo de um documento analisado, calcula-se o valor *do TF Normalizado Corrente*, como apontado na linha 1. Esse valor corresponde ao valor normalizado do *TF* para cada termo do corrente documento.

Na linha 2, determina-se se o termo já ocorreu em outro documento (DF > 1) da coleção analisada. Se sim, o *TF Normalizado* utilizado no cálculo do peso do termo é igual ao *TF Normalizado Corrente* – linha 5. Caso contrário, o *TF Normalizado* calculado para a coleção sem o documento corrente é depreciado em função de DF – (TF Normalizado * DF – 1) **-**, para minimizar o valor histórico no cálculo do valor corrigido – linha 3.

A comparação entre os algoritmos TF-IDF e TF-Seno é apresentada na Figura 4.5:

Comparação entre TF-Seno e TF-IDF

Termo	DF	TF	TF Norm	Peso TF-Seno	Peso Seno Norm	Peso TF-IDF	Peso TF-IDF Norm
retrieval	55.	2.	61.38498783	61.38498783	95.88600397	0	0
information	53.	9.	64.12329078	64.01871443	100	1.03153982975349	5.54192782065224
document	51.	0.	29.65840101	29.46507931	46.02572918	.972573512433484	5.22513241924224
system	50.	12.	27.77715623	27.49442458	42.94748008	1.14977108241922	6.17712294305032
relevance	48.	10.	35.34140885	34.63749886	54.10526991	2.08943541113114	11.2254514080865
user	44.	150.	25.76000988	24.49922562	38.26884925	2.49640289255846	13.4118763452233
query	44.	148.	40.52954018	38.5458827	60.21033525	3.92771826605434	21.1015905566461
text	43.	4.	12.49888241	11.77200228	18.38837713	1.33605846000634	7.17794828271084
feedback	42.	10.	29.81727719	27.78564692	43.40238273	3.49200268351877	18.7607170013102
search	42.	85.	20.1981917	18.82196814	29.40072715	2.36547888558655	12.7084896453589
research	41.	8.	8.944766223	8.239235729	12.87004203	1.14116283417119	6.13087529552686
number	40.	65.	7.961413264	7.241956145	11.31224856	1.10108493562802	5.91555755933947
work	40.	2.	5.673889443	5.16115129	8.061941713	.78471421906408	4.21586199238847
relevant	39.	1.	8.789842576	7.887990773	12.32138276	1.31230664179014	7.05034209789211
result	37.	15.	8.957199752	7.799446583	12.18307316	1.54208015682903	8.28479587147776
term	35.	154.	17.37708002	14.61852938	22.83477485	3.41102775616424	18.3256807673835
technique	34.	1.	6.563269347	5.417794362	8.462829143	1.37096046081986	7.36545860827056
database	34.	0.	6.748691946	5.570855364	8.701916784	1.40969223279683	7.57354430546912
approach	33.	1.	4.519682378	3.656499833	5.711610988	1.00268588422238	5.38691055533476

Figura 4.5: Comparação entre TF-Seno e TF-IDF

A primeira coluna da figura contém o termo analisado. A segunda coluna, *DF,* representa o número de documentos onde o termo aparece e a terceira, ***TF,*** a freqüência do termo no documento analisado.

A quinta coluna, *Peso TF-Seno,* informa o peso do termo na coleção de documentos, calculado pelo *TF-Seno*. As colunas *TF Norm, Peso Seno Norm e Peso TF-IDF Norm* representam os valores normalizados do *TF, do Peso Seno e do Peso TF-IDF,* respectivamente.

A coluna *Peso TF-IDF* representa o peso do termo calculado com o *algoritmo TF-IDF*.

Em um dos testes do *Fidus*, foram analisados 55 documentos HTML recuperados por componentes de busca. Os termos estão ordenados decrescentemente pelo *DF*.

O primeiro termo – *retrieval* - apareceu nos 55 documentos analisados, com peso calculado pelo *TF-IDF* igual a zero. Segundo esse algoritmo, um termo que aparece em todos os documentos de uma coleção pode ser considerado uma *stopword,* devido a sua baixa seletividade para a representação do contexto analisado. O segundo termo da lista – *information* - apresenta peso TF-IDF igual a 1.03. Em contrapartida, o termo – *term* – muito freqüente no documento 55 – último documento analisado na coleção e para o qual o valor de *TF* na lista é apresentado – mas pouco freqüente na coleção, possui peso *TF-IDF* maior.

Como o *Fidus* requer que os termos muito freqüentes em muitos documentos da coleção analisada tenham alto poder de representação do contexto dos documentos [CRE 95], como apresentado na seção 2.4.4 do capítulo 2, eles precisam ter alto peso. Esse resultado é obtido calculando-se o peso com o algoritmo TF-Seno.

A estratégia adotada é a seguinte: existem termos no documento HTML, destacados por “tags” especiais como “title” e “meta”, que podem determinar explicitamente seu contexto. Esses termos devem ter seu peso ajustado para ressaltar sua relevância [ZHA 9?].

No cálculo de pesos do *Fidus*, incrementamos a freqüência dos termos em função dessas “tags” especiais, o que implica no ajuste dos pesos, pois esses são proporcionais às freqüências. Como exemplo, os termos que aparecem entre as “tags” de título tem sua freqüência incrementada em 10 – como se o termo ocorresse mais 10 vezes no documento.

Os incrementos usados têm seus valores baseados no algoritmo de cálculo de pesos do WebMate [CHE98].

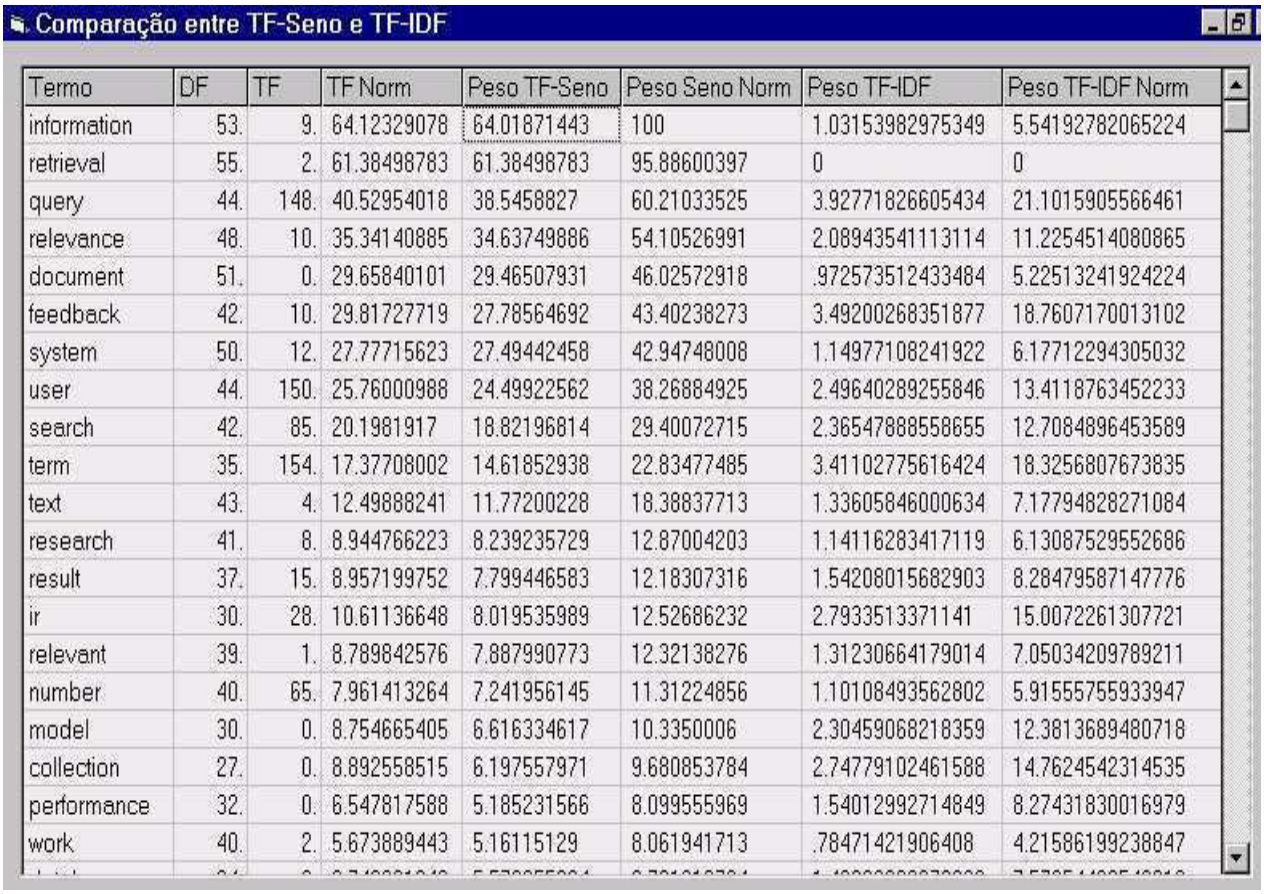
A correta análise dos dados apresentados na figura 4.5 é feita pela comparação entre os pesos normalizados, *Peso TF-Seno Norm* e o *Peso TF-IDF Norm*. Os resultados apresentados provam que o *TF-Seno* é mais adequado aos requisitos do *Fidus* para atribuir pesos aos termos que ocorrem com alta freqüência em muitos documentos.

Componente de Cálculo de Similaridade

O componente de cálculo de similaridade do *Fidus* analisa cada documento recuperado pelos *componentes de busca* para determinar o grau de similaridade entre o documento e a respectiva categoria do perfil do usuário. Esse grau de similaridade determina o potencial de relevância do documento.

Esse cálculo consiste numa comparação que determina quantos dos 20 termos de maior peso do vetor de termos que representa o documento coincidem com os 20 termos de maior peso do vetor que representa a categoria de informação para a qual o documento foi recuperado.

A decisão do uso de 20 termos pode ser explicada com a análise da figura 4.6:



Termo	DF	TF	TF Norm	Peso TF-Seno	Peso Seno Norm	Peso TF-IDF	Peso TF-IDF Norm
information	53.	9.	64.12329078	64.01871443	100	1.03153982975349	5.54192782065224
retrieval	55.	2.	61.38498783	61.38498783	95.88600397	0	0
query	44.	148.	40.52954018	38.5458827	60.21033525	3.92771826605434	21.1015905566461
relevance	48.	10.	35.34140885	34.63749886	54.10526991	2.08943541113114	11.2254514080865
document	51.	0.	29.65840101	29.46507931	46.02572918	.972573512433484	5.22513241924224
feedback	42.	10.	29.81727719	27.78564692	43.40238273	3.49200268351877	18.7607170013102
system	50.	12.	27.77715623	27.49442458	42.94748008	1.14977108241922	6.17712294305032
user	44.	150.	25.76000988	24.49922562	38.26884925	2.49640289255846	13.4118763452233
search	42.	85.	20.1981917	18.82196814	29.40072715	2.36547888558655	12.7084896453589
term	35.	154.	17.37708002	14.61852938	22.83477485	3.41102775616424	18.3256807673835
text	43.	4.	12.49888241	11.77200228	18.38837713	1.33605846000634	7.17794828271084
research	41.	8.	8.944766223	8.239235729	12.87004203	1.14116283417119	6.13087529552686
result	37.	15.	8.957199752	7.799446583	12.18307316	1.54208015682903	8.28479587147776
ir	30.	28.	10.61136648	8.019535989	12.52686232	2.7933513371141	15.0072261307721
relevant	39.	1.	8.789842576	7.887990773	12.32138276	1.31230664179014	7.05034209789211
number	40.	65.	7.961413264	7.241956145	11.31224856	1.10108493562802	5.91555755933947
model	30.	0.	8.754665405	6.616334617	10.3350006	2.30459068218359	12.3813689480718
collection	27.	0.	8.892558515	6.197557971	9.680853784	2.74779102461588	14.7624542314535
performance	32.	0.	6.547817588	5.185231566	8.099555969	1.54012992714849	8.27431830016979
work	40.	2.	5.673889443	5.16115129	8.061941713	.78471421906408	4.21586199238847

Figura 4.6: Comparação entre TF-Seno e TF-IDF ordenado por Peso Seno Norm

A Figura 4.6 apresenta o mesmo resultado da Figura 4.5, ordenado pelo *Peso Seno Norm*. Esse resultado representa o comportamento das coleções analisadas para os testes do

Fidus: poucos termos são realmente representativos – seletivos - para o contexto da coleção, como resultado do algoritmo TF-Seno. Determinou-se, então, empiricamente, que os 20 termos de maior peso seriam suficientes para determinar a similaridade entre um documento e o contexto analisado.

Embora outros algoritmos de cálculo mais eficientes sejam propostos, eles baseiam-se na representação de documento e categorias como vetores de igual dimensão [ZHA 9?]. Para isso, todos os termos indexáveis num documento precisam estar presentes num dicionário, que é representado como um vetor T, de dimensão *n:* $T <t_1, t_2, ..., t_n>$. Documentos e contextos são representados por vetores de dimensão *n*, indicando-se a cada posição *i* se o *i-ésimo* termo do dicionário existe ou não no documento ou contexto.

A similaridade entre um documento *D* e um contexto *C* de uma coleção de documentos, com essa representação, é normalmente calculada pelo produto dos vetores de *D* e *C* ou de seus vetores normalizados.

Como o *Fidus* não possui um dicionário para determinar todos os termos indexáveis, mantendo apenas uma lista de termos a não indexar – lista de stopwords -, a representação de um documento *D* e de um contexto *C* não garantem que o *i-ésimo* termo do vetor de *D* e *C* são iguais. Logo, seu cálculo é bastante simplificado, embora eficiente, como será discutido na seção 5.3 do capítulo 5.

Componente de Validação de Ponteiro

Toda informação mantida pelo *Fidus* mantém um título e a URL de onde seu conteúdo pode ser recuperado – *ponteiro* -.

O componente de validação de *ponteiro* é responsável por determinar se a URL da informação armazenada permanece válida – permanece apontando para o documento existente -, pois informações na Web podem ser removidas ou movidas de forma não controlada pelo *Fidus*.

Seu processo consiste em selecionar todas as informações persistentes, efetuar uma requisição HTTP endereçada com a URL de cada informação e analisar o conteúdo da página HTML retornada. Se essa página contiver uma informação como: “Not Found", "Sorry, but the page", "we couldn't find the page"), ou algo parecido – o *Fidus* mantém uma lista de mensagens semelhantes - a mesma é considerada inválida – *ponteiro* inválido -.

Organizador de Tarefas

As tarefas executadas pelos componentes de negócio do *Fidus,* como o *processo de busca global,* são coordenadas pelo *organizador de tarefas*[15]*,* que determina quando cada tarefa deve ser executada através do envio de *mensagem* ao componente correspondente.

O Organizador de Tarefas foi implementado como uma coleção de serviços do Windows NT. Esses serviços do NT são processos que executam no Windows NT de forma independente – não dependem de interação com o usuário - e assíncrona.

[15] Tarefa, nesse contexto, representa um *processo gerenciado pelo Windows NT.*

4.4 Persistência

A persistência dos dados, responsabilidade da *camada de persistência*, é garantida pelo SGBD OO *Jasmine*. A razão determinante da escolha do *Jasmine* foi seu suporte ao modelo de dados orientado a objeto e sua a maturidade tecnológica.

As classes persistentes são definidas e codificadas apenas em ODQL; seus métodos são codificados em ODQL puro e em ODQL com C/C++ embutido.

A definição de uma classe persistente no Jasmine é apresentada a seguir:

```
[1] defineClass Consulente super: Usuario {
[2] class:
[3]       String className default: "Consulente";

[4] instance:
            /* Atributos */
[5]       Perfil perfil default: NIL;

            /* Métodos */
[6]       Perfil getPerfil();
[7]       Void setPerfil(Perfil perfil);
};
```

A linha 1 define a classe *Consulente* como sendo uma especialização da classe *Usuário*. A linha 2 define que os atributos e métodos que a seguirem terão o escopo de *classe*. A linha 4 define o escopo de atributos e métodos de instância.

Maiores detalhes sobre o Jasmine encontram-se no Apêndice A.

4.5 Conclusão

A implementação do *Fidus*, baseada em componentes, torna-o facilmente alterável para o acréscimo ou substituição de funcionalidades. O componente de cálculo de similaridades, por exemplo, pode ser substituído facilmente por outro que implemente um novo cálculo de similaridades mais preciso, bastando, para isso, respeitar a interface definida para o mesmo.

O componente global de busca, para aumentar a abrangência da meta-busca, necessita apenas da construção de novos componentes especializados de busca com a mesma interface dos demais. Assim, um componente especializado de busca pode ser desenvolvido para obter informações de um sistema X – independente de como tal sistema apresenta suas informações a seus usuários – e sua integração com o componente global de busca é, praticamente, automática.

O desenvolvimento do *TF-Seno* é considerado um grande diferencial do *Fidus*. Esse algoritmo garante que o peso dos termos, calculado em função de sua freqüência, tem grande poder de representar a seletividade dos mesmos em grandes coleções de documentos.

Outro diferencial do *Fidus,* implementado no *TF-Seno,* é a correção do peso dos termos em função da relevância do documento. Dessa forma, a seletividade do termo torna-se proporcional à relevância efetiva do documento.

A utilização do Jasmine permitiu que o paradigma de orientação a objetos fosse mantida durante todas as fases do desenvolvimento. O modelo de projeto foi implementado no Jasmine de forma direta e sem diferença semântica.

Capítulo 5

Avaliação Experimental do Fidus

5.1 Introdução

Os objetivos da avaliação experimental do *Fidus* são dois: (1) determinar sua efetividade na busca e gerência de informações; (2) verificar como sua meta-busca, capturando resultados de múltiplos provedores de informação, interfere nessa efetividade.

Para avaliar a efetividade do *Fidus*, duas métricas são analisadas:

1. Precisão da busca – determinada pelo percentual de informações efetivamente relevantes no conjunto de resultados. Esta métrica é detalhada na seção 2.6 do capítulo 2.
2. Precisão da relevância potencial – refina a primeira métrica, restringindo o conjunto de resultados para conter somente documentos potencialmente relevantes, cada documento tendo relevância potencial superior ou igual a 40%.

Sua fórmula é a seguinte:

$$\text{Precisão da Relevância Potencial} = \frac{\text{Total de informações realmente relevantes}}{\text{Total de informações com Relevância Potencial} >= 40\%}$$

O valor mínimo de 40% para a relevância potencial foi escolhido com base nos resultados de diversos testes efetuados com o *Fidus,* onde determinou aos documentos uma alta probabilidade de ser efetivamente relevante. Esse valor significa que, dos 20 termos de maior peso no vetor de termos de um documento, 8 deles estão presentes entre os 20 termos de maior peso do vetor de termos do contexto- cálculo de similaridades, explicado na seção 4.3.2 do capítulo 4 -.

5.2 Modelo de Teste

Os testes efetuados para a avaliação do *Fidus* simulam o comportamento de usuários que visam encontrar documentos pertencentes a uma determinada área de pesquisa bibliográfica, através da seleção de alguns termos considerados como mais representativos.

Múltiplas iterações de testes devem ser utilizadas, cada uma com um propósito específico, como testar a abrangência da meta-busca com um número variável de provedores de informação, testar a precisão e seletividade dos termos, etc.

O modelo de teste proposto consiste de um conjunto de passos:

Passo 1 - Criação das categorias de informação que representam a área de pesquisa bibliográfica de interesse. Determinadas as categorias, selecionar os critérios de busca mais representativos das mesmas.

Passo 2 - Busca de informações que satisfaçam aos critérios das categorias.

Passo 3 - Determinação da relevância potencial das informações.

Passo 4 - Análise da relevância efetiva das informações – determinar se as informações recuperadas pelo Fidus são realmente importantes e úteis ao contexto e ao usuário.

4.1 Seleção de documentos relevantes.

4.2 Descarte de documentos irrelevantes.

Passo 5 - Construção do Vetor de Termos para a categoria.

Passo 6 - Refinamento de critérios de busca com a seleção dos termos sugeridos pelo *Fidus* ou com adição de novos termos.

Enquanto a iteração não estiver concluída, deve-se retornar ao Passo 2.

Passo 7 - Determinação da precisão do *Fidus*, isto é, a precisão da busca e a precisão da relevância potencial.

Passo 8 - Análise dos fatores que interferem na precisão, como a abrangência provida ao *Fidus* por múltiplos provedores de informação, baixa seletividade dos termos utilizados na busca, etc.

Esse modelo de teste permite ao *Fidus* efetuar a busca com múltiplos provedores de informação, como *Google* e *Altavista*, isoladamente ou em conjunto, possibilitando a análise da influência da quantidade e qualidade dos resultados desses provedores nos resultados finais, integrados pelo *Fidus*.

5.3 Primeira Iteração

Para a execução do conjunto de testes apresentado, foi criado um perfil com uma única categoria de informação, representando a área de pesquisa bibliográfica de interesse: recuperação de informações. Todas as buscas de informação foram feitas para essa categoria. Esse conjunto de testes é dividido em iterações, que retornam conjuntos de informações diferenciados por critérios de busca ou provedores da meta-busca.

A primeira iteração dessa fase de testes foi efetuada apenas com o *Google* como provedor de informação. Os critérios de busca adotados foram:

[*information retrieval*]

A tabela 5.1 apresenta os documentos localizados pelo *Fidus* através de seu processo de meta-busca:

#	☑	Título	URL
1	0	BCS Information Retrieval Group	http://irsg.eu.org/
2	0	Information Retrieval	http://www.dcs.gla.ac.uk/Keith/Preface.html
3	0	The Information Retrieval in Chemistry	http://macedonia.chem.demokritos.gr/
4	0	Glasgow Information Retrieval Home Page	http://ir.dcs.gla.ac.uk/
5	0	CIIR at UMass	http://ciir.cs.umass.edu/
6	0	Survey of Information Retrieval	http://www.cs.jhu.edu/~weiss/ir.html
7	0	IR and IE on the Web using hypertext meta-data and ...	http://www.mri.mq.edu.au/~einat/web_ir/
8	0	Information Retrieval Links	http://web.syr.edu/~diekemar/ir.html
9	0	Cross-Language Information Retrieval Resources	http://www.clis.umd.edu/dlrg/clir/
10	0	Modern Information Retrieval	http://www.sims.berkeley.edu/~hearst/irbook/

Tabela 5.1 – Resultados da primeira iteração de busca.

A primeira coluna da tabela, identificada pelo símbolo "#", numera os documentos localizados. A segunda coluna, identificada pelo símbolo ☑, apresenta a relevância potencial do documento, estimada pelo *Fidus*.

Nessa iteração, a relevância potencial é zero para todos os documentos, pois ainda não houve nenhum julgamento de relevância efetiva.

A cores que diferenciam as linhas representam o seguinte:

Cor	Significado
AZUL	Documento julgado pelo usuário como muito relevante Relevância igual a **1** para o cálculo do peso pelo *TF-Seno*[16].
VERDE	Documento julgado pelo usuário como medianamente relevante. Relevância igual a **0,5** para o cálculo do peso pelo *TF-Seno*.
VERMELHO	Documento descartado pelo usuário por baixa relevância efetiva.
CINZA	Documento com conteúdo HTML não encontrado.

Tabela 5.2 – Significado das cores na representação dos resultados dos testes.

Embora a arquitetura do *Fidus* apresente um componente que valida os ponteiros dos documentos, linhas da cor cinza poderão aparecer nos resultados, pois não existe um padrão efetivo para determinarmos quando uma página não foi encontrada. Páginas que indicam a falha no acesso ao conteúdo HTML não possuem padrão.

Documentos com conteúdo HTML não encontrado não serão considerados nos cálculos das métricas de precisão.

Nessa iteração, a *precisão da busca* foi de 50%. Como os critérios não estão refinados, a busca ainda é muito abrangente e pouco precisa.

Após a avaliação da relevância efetiva dos documentos - passo 4 - e da construção do vetor de termos - passo 5 -, o *Fidus* determinou os termos de maior peso – calculado com o *TF-Seno* - para os documentos julgados como efetivamente relevantes, permitindo a execução do passo 5. Esses termos são:

information retrieval resource work research processe text ir language university system group theory bibliography document project conference workshop experiment analysis

[16] **TF-Seno = TF Normalizado * Sin(ângulo) * Relevância**

A aplicação do passo 5 a essa coleção de termos resulta num novo conjunto de termos que servirá de base para uma nova iteração, provavelmente mais precisa.

5.3.1 Primeiro Refinamento

O primeiro refinamento da primeira iteração selecionou os seguintes critérios de busca:

[*information retrieval relevance feedback*]

Os documentos localizados pelo *Fidus,* ainda com o *Google* como provedor de informação, estão apresentados na Tabela 5.3:

#	☑	Título	URL
1	50	Affording relevance feedback in information retrieval ...	http://www.cl.cam.ac.uk/~kr205/irhci.html
2	50	Machine Learning for Information Retrieval: Neural Networks, ...	http://ai.bpa.arizona.edu/papers/mlir93/mlir93.html
3	50	Trans. KDE: Abstract: Enabling Concept-Based Relevance Feedback	http://www.computer.org/tkde/tk1999/k0595abs.htm
4	40	Supporting Interactive Information Retrieval Through Relevance ...	http://www.uni-paderborn.de/StaffWeb/chi96/ElPub/WWW/chi96www/doctoral/Koenemann/Jk2_txt1.htm
5	40	Naive Bayesian Parameter Estimation for Information Retrieval	http://www.ils.unc.edu/~losee/para.html
6	40	Citation details: Relevance Feedback in Information Retrieval ...	http://citeseer.nj.nec.com/context/29926/0
7	30	Citation details: Improving retrieval performance by relevance ...	http://citeseer.nj.nec.com/context/21661/0
8	25	IRHCI Workshop	http://www.dcs.gla.ac.uk/irhci/
9	15	Slide 2.8: Information retrieval: Relevance feedback	http://www.eng.auburn.edu/users/wenchen/course/7120/proj/2/8.html
10	10	Information Retrieval References (Arnon Rungsawang)	http://www-inf.enst.fr/~rungsawa/irrs.html

Tabela 5.3 – Resultados do primeiro refinamento da primeira iteração de busca.

Nesse refinamento, a *precisão da busca* foi de 77%. O incremento de precisão observado pode ser atribuído ao refinamento dos critérios de busca e ao fato do *Fidus* só apresentar novas informações a cada iteração, como descrito na seção 3.3 do capítulo 3.

A coluna 2 apresenta a relevância potencial de cada documento, pois já houve um julgamento de relevância efetiva na primeira iteração, possibilitando o cálculo. *A precisão da relevância potencial*, nesse refinamento, foi de 100%. Todos os documentos com relevância potencial igual ou superior a 40% foram considerados relevantes.

Com o aumento da amostragem dos documentos julgados como efetivamente relevantes o algoritmo de cálculo de relevância pode determinar, com maior precisão, o potencial de relevância dos novos documentos localizados pelo *Fidus*.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

> information retrieval feedback relevance document query system user term performance relevant search interactive weight algorithm technique learn research text ir

Como abordado na seção 3.4 do capítulo 3, o fato do *Fidus* não possuir um dicionário controlado de termos e possuir um mecanismo simplificado para a redução de termos a radicais comuns faz com que, como na coleção de termos acima, os termos "relevance" e "relevant" sejam considerados como distintos. O ideal é que esses termos fossem considerados como um só, possivelmente como "relevant", o que aumentaria a precisão do cálculo de pesos e de similaridades.

5.4 Segunda Iteração

Essa iteração foi efetuada com os mesmos critérios do refinamento da primeira iteração e manteve apenas o *Google* como provedor de informação. Seu objetivo é validar a precisão do *Fidus* com múltiplas buscas para os mesmos critérios, avaliando seu comportamento para resultados mais extensos. Para isso, aumentou-se o número de documentos na coleção de resultados de 10 para 20.

Os documentos localizados pelo *Fidus,* como resultado dessa iteração, estão apresentados na Tabela 5.4:

#	☑	Título	URL
1	60	Integrating Query Expansion and Conceptural Relevance Feedback	http://www7.scu.edu.au/programme/posters/1887/com1887.htm
2	55	Using Relevance Feedback over Multiple Search Engines	http://www.comp.nus.edu.sg/~wanghao/new/ic52c1_report.htm
3	45	Boolean Queries, Fuzzy Logic, Relevance Feedback, and ...	http://www.ils.unc.edu/~losee/cnf.html
4	45	The Use of Relevance Feedback on the Web: Implications for ...	http://jimjansen.tripod.com/academic/pubs/webnet99.html
5	45	User Modeling for Information Retrieval on the Web	http://wwwis.win.tue.nl/asum99/wilkinson.html
6	35	Information Retrieval	http://www.slais.ubc.ca/courses/arstlibr512/00-01-wt2/informat.htm
7	35	725: Information Retrieval: Algorithms and Models	http://www.cs.sc.edu/acadinfo/gradcourses/725.shtml
8	35	COMP 336 Information Retrieval [3-0-0:3]	http://www.cs.ust.hk/faculty/dlee/336/
9	30	Efficiency Considerations for Scalable Information Retrieval	http://jodi.ecs.soton.ac.uk/Articles/v01/i05/Frieder/
10	30	CAIVL'97: Abstract: A Relevance Feedback Architecture for ...	http://www.computer.org/proceedings/caivl/7981/79810082abs.htm
11	25	Machine Learning and Information Retrieval (Belew/Shavlik)	http://www-cse.ucsd.edu/users/rik/MLIA.html
12	25	Information Retrieval	http://www.cs.brandeis.edu/~niksa/OTHER/other1.html
13	25	CS 838 – Machine Learning and Information Retrieval (Belew/ ...	http://www.cs.wisc.edu/~shavlik/cs838.html
14	20	GIR – Geographic Information Retrieval, SSC	http://www.ssc.se/gir/
15	10	IR Vocabulary	http://www.cs.jhu.edu/~weiss/glossary.html

16	10	Comparing Neural and Probabilistic Relevance Feedback in an ...	http://www.dcs.gla.ac.uk/fermi/tech_reports/Crestani2/article.html
17	5	600.466 – Information Retrieval and Web Agents	http://www.cs.jhu.edu/~yarowsky/cs466.html
18	5	ACM Digital Library: Supporting interactive information ...	http://riss.keris.or.kr:33333/pubs/citations/proceedings/chi/257089/p49-koenemann/
19	0	Stündel, Frank: Entwicklung einer Metasuchmaschine mit ...	http://dol.uni-leipzig.de/pub/1998-90
20	0	Information Retrieval: CHAPTER 11: RELEVANCE FEEDBACK AND ...	http://matrix.nbu.bg/books/book5/chap11.htm

Tabela 5.4 – Resultados da segunda iteração de busca.

Nessa iteração, os documentos válidos, excetuando aqueles com conteúdo HTML não localizado, são 15.

A *precisão da busca* é de 47%. Todavia, embora a *precisão da busca* tenha diminuído, a *precisão da relevância potencial* manteve-se em 100%. Logo, pode-se inferir que, embora a abrangência da busca retorne documentos não condizentes ao contexto da procura ou, como o documento 19, com títulos em inglês e conteúdo em alemão, a análise de similaridades é um fator adequado de correção da precisão do sistema.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

> information retrieval feedback relevance query document user system search model relevant result performance database term text research engine base

5.4.1 Primeiro Refinamento

Mantendo apenas o *Google* como provedor de informação, esse refinamento tem o objetivo de demonstrar a precisão do *Fidus* com critérios de busca de exclusão, informando os termos que não devem estar presentes nos documentos a localizar.

Os critérios de busca utilizados foram:

[*information retrieval relevance feedback system –course –student -instructor*]

Os critérios de exclusão "course", "student" e "instructor" foram escolhidos pois alguns dos documentos descartados na iteração anterior representavam material de aula com esses termos.

Os documentos localizados pelo *Fidus,* como resultado desse refinamento, estão apresentados na Tabela 5.5:

#	☑	Título	URL
1	70	What Do People Want From IR by B. Croft	http://www.dlib.org/dlib/november95/11croft.html
2	60	AIRE (Advanced Information Retrieval Engine)	http://www.ir.iit.edu/~abdur/research/aire/AIRE.html
3	60	Information Retrieval	http://www.pitt.edu/~korfhage/irwork.html
4	60	An Inductive Machine Learning Approach to Information ...	http://ai.bpa.arizona.edu/papers/id3ir93/id3ir93.html
5	60	Integrating Query Expansion and Conceptural Relevance Feedback ...	http://www.csie.ncu.edu.tw/~chia/pub/www7/353.html
6	55	Citation details: Relevance Feedback in Information Retrieval ...	http://citeseer.nj.nec.com/context/206393/0
7	55	Technical description to the Relevance Feedback Module for ...	http://ls6-www.informatik.uni-dortmund.de/~klas/tech_doc.one/tech_doc.html
8	50	Natural Language Processing in Information Retrieval	http://www.onlineinc.com/onlinemag/OL1999/feldman5.html
9	45	IR Papers	http://www.cs.jhu.edu/~weiss/papers.html
10	40	Citation details: Relevance feedback in information retrieval (...	http://citeseer.nj.nec.com/context/1176039/0
11	20	Overhead-Folie: Information Retrieval - System Driven Approach	http://www.iud.fh-darmstadt.de/iud/wwwmeth/publ/slide/irsda1.htm
12	15	Lexikon-Stichwort: Information Retrieval-Anwendungen ('95)	http://www.iud.fh-darmstadt.de/iud/wwwmeth/publ/paper/iranw95/paper1.htm
13	15	Intelligentes Information Retrieval	http://www.ifs.univie.ac.at/~gq/IIRW3.html
14	5	Information Retrieval	http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/hks/inf_ret.html

15	5	LIC Current Awards - information retrieval	http://www.lic.gov.uk/awards/ir-curpj.html
16	5	LIC Research - Information Retrieval Call results	http://www.lic.gov.uk/research/information_retrieval/ir-calla.html
17	5	People: Glasgow Information Retrieval Group	http://ir.dcs.gla.ac.uk/people.html
18	5	www.idi.ntnu.no/~mlh/papers/keogh99relevance.pdf	http://www.idi.ntnu.no/~mlh/papers/keogh99relevance.pdf
19	5	Concept: Relevance feedback	http://www.dcs.gla.ac.uk/~iain/keith/data/concepts/254.htm
20	5	Learning from Relevance Feedback in Latent Semantic Indeixing	http://www.dsv.su.se/~rick/SAIS.html

Tabela 5.5 – Resultados do primeiro refinamento da segunda iteração de busca.

A *precisão da busca,* nesse refinamento, é de 77%. Esse novo resultado justifica-se pela exclusão dos critérios que aparecem em documentos irrelevantes.

Com o crescimento da amostragem dos documentos julgados, a relevância potencial indicada pelo *Fidus* já é bastante representativa. A permanência da precisão da relevância potencial em 100% é um forte indicador que o cálculo de similaridades possui grande efetividade.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

information retrieval relevance feedback query document system user search term ir research base technique model relevant database text work automatic result

5.4.2 Segundo Refinamento

Os critérios de busca foram estendidos, selecionando-se termos sugeridos pelo *Fidus,* compondo a seguinte coleção:

[*information retrieval relevance feedback system* document relevant

–course –student –instructor]

Os documentos localizados pelo *Fidus,* como resultado desse refinamento, estão apresentados na Tabela 5.6:

#	☑	Título	URL
1	60	An Information Retrieval Query Interface	http://www.crema.unimi.it/mirror/scheme/thesis/node111.html
2	55	Information Retrieval	http://inf2.pira.co.uk/top018.htm
3	55	Information Retrieval, Homeosemy, Natural Language	http://www.intellimatch.com/search/dissertation/node10.html
4	55	SMI Information Retrieval Group	http://www-smi.stanford.edu/people/felciano/research/ir/
5	50	Recent developments in Indexing, Searching and Information ...	http://www.iihe.ac.be/scimitar/J0799/isir.html
6	50	Khoo	Http://aztec.lib.utk.edu/libres/libre7n2/khoo.html
7	50	DART: Information Retrieval	Http://www.uk.research.att.com/dart/information_retrieval.html
8	45	2. Information retrieval	Http://dollar.biz.uiowa.edu/~fil/Thesis/node71.html
9	45	Improving Retrieval Performance by Relevance Feedback -- ...	Http://www.northlink.com/~derekb/databrain/3524.htm
10	45	Cross-Language Information Retrieval: An Analysis of Errors	Http://www.cs.uiowa.edu/~mruiz/papers/iowa-asis98.html
11	45	Intelligent Information ...	Http://ir.csie.ncku.edu.tw/glossary.htm
12	40	Active by Accident: Relevance Feedback in Information ...	Http://citeseer.nj.nec.com/lewis95active.html
13	35	rfc1689 - A Status Report on Networked Information Retrieval ...	http://www.faqs.org/rfcs/rfc1689.html
14	35	Topic Tracking with the PRISE Information Retrieval System	http://www.nist.gov/speech/publications/darpa99/html/tdt350/tdt350.htm
15	30	Genetic-Based Adaptation of Matching Functions for Effective ...	http://www-personal.umich.edu/~wfan/matching-function.html
16	25	Ordliste for Information Retrieval	http://www-bib.hive.no/felles/le/ir/
17	20	Advanced Information Retrieval Methods	http://www.uni-karlsruhe.de/~uo3f/seminarir/SeminarWeb.html
18	10	Ranking and Relevance Feedback Extensions to a View-based ...	http://www.hud.ac.uk/schools/cedar/hiembasepaper/rankrel.htm

19	10	Information Retrieval for Mixed-Media Collections	http://www.dcs.ex.ac.uk/~gareth/irmmc.html
20	0	LIC: Research Bulletin 2 – Current Research in Information ...	http://www.lic.gov.uk/publications/bulletin/issue2/review2.html

Tabela 5.6 – Resultados do segundo refinamento da segunda iteração de busca.

Nesse refinamento, a *precisão da busca* é de 70% e a *precisão da relevância potencial* é de 83%. Se desconsiderarmos o documento 11, cujo título está em inglês e o conteúdo em chinês, a precisão da relevância potencial sobe para 90%. Esse tipo de filtro, embora não implementado no *Fidus*, pode ser desenvolvido com base na distribuição de termos do documento.

O problema do documento 11 recai na relevância do título e na normalização de pesos dos termos, como afirmado na seção 4.3.2 do capítulo 4.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

information retrieval relevance feedback query document system user search text relevant research term base result database performance model technique ir

5.5 Terceira Iteração

Essa iteração de busca foi efetuada com o *Google* e o *Altavista* como provedores de informação, tornando-a mais abrangente. A maior abrangência da busca pode diminuir sua precisão, pois a probabilidade de localização de documentos irrelevantes tende a crescer.

Os critérios de busca utilizados foram os mesmos do segundo refinamento da segunda iteração.

Os documentos localizados pelo *Fidus,* como resultado dessa iteração, estão apresentados na Tabela 5.7:

#	☑	Título	URL
1	50	Citation details: Relevance feedback: A power tool in interactive content-base	http://citeseer.nj.nec.com/context/52231/324854
2	50	Oingo:	http://www.oingo.com/topic/10/10293.html
3	40	On the Net, Rising Relevance in Search Engines	http://www.onlineinc.com/onlinemag/OL1999/net5.html
4	35	The Effect of Adding Relevance Information in a Relevance Feedback Environment	http://citeseer.nj.nec.com/buckley94effect.html
5	35	Information Retrieval with Human Indexing	http://idm.internet.com/features/humanindex-1.shtml
6	25	Faxback Information	http://support.tandy.com/faxback.htm
7	25	E-Muni – disclosure documents, news, information and research relating to the	http://www.smkc.com/
8	25	Hawaii Pesticide Information Retrieval System Home Page	http://pestworld.stjohn.hawaii.edu/cfdocs/test/hpirs.htm
9	20	WebCounter Information Query	http://www.digits.com/query.html
10	20	FDIC: Real Estate Retrieval System	http://www2.fdic.gov/drrore/
11	20	Air Pollution – Aerometric Information Retrieval System (AIRS)	http://www.epa.gov/docs/airs/airs.html
12	15	Missing WebCounter Document	http://www.digits.com/missing.html
13	15	About BSCW	http://bscw.gmd.de/about.html
14	15	Adobe Acrobat Reader	http://www.adobe.co.uk/products/acrobat/readstep.html
15	05	Welcome to the US Army Corps of Engineers' Project Information Retrieval System	http://pirs.mvr.usace.army.mil/

16	05	1998 Guest Information System ~ MGH NeuroScience Care Units	http://gis.mgh.harvard.edu/
17	0	Center for Aquatic and Invasive Plants	http://aquat1.ifas.ufl.edu/welcome.html
18	0	ViaVet: Relevance	http://www.vet.uu.nl/english/research/programma/large_animal_medicine/relevance
19	0	Center for Aquatic and Invasive Plants	http://aquat1.ifas.ufl.edu/
20	0	ViaVet: Relevance	http://www.vet.uu.nl/english/research/programma/pathology/relevance

Tabela 5.7 – Resultados da terceira iteração de busca.

Nessa iteração, a *precisão da busca* é de 11%. A *precisão da relevância potencial* é de 33%.

Os documentos 7 e 8 exemplificam bem a queda da *precisão da busca* pelo aumento da abrangência causada pelos múltiplos provedores de informação. Tais documentos pertencem a um contexto de "recuperação de informações", atendem aos critérios especificados, mas não pertencem a mesma área de interesse pressuposto. Essa área, de recuperação de informações textuais com base no retorno de relevância, torna-se mal representada ou caracterizada por poucos critérios de busca quando a abrangência da meta-busca aumenta.

A queda da *precisão da relevância potencial* é justificada pelo mesmo fator. O documento 1 é direcionado para a recuperação de imagens, enquanto o documento 2 é o resultado de uma outra ferramenta de busca, Oingo, como uma página HTML estática. Nesses dois casos, a análise de similaridades procede, pois os termos mais relevantes condizem com os termos do contexto da categoria, embora o julgamento da relevância efetiva seja negativa.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

> information retrieval relevance feedback query document system user search relevant text research term database result performance base technique ir model

5.5.1 Primeiro Refinamento

Esse refinamento foi efetuado com os seguintes critérios de busca:

[*relevance feedback information retrieval system filtering document relevant query ir -course -student -instructor -next -prev*]

Os critérios de exclusão "next" e "prev" foram inseridos para excluir os slides que foram localizados como relevantes na iteração anterior e julgados efetivamente como irrelevantes, o que decrementou a precisão. Novos termos foram inseridos para diminuir a abrangência das buscas e, consequentemente, maximizar a precisão de busca e de relevância.

Os documentos localizados pelo *Fidus,* como resultado desse refinamento, estão apresentados na Tabela 5.8:

#	☑	Título	URL
1	70	A Case for Interaction: A Study of Interactive Information ...	http://www.uni-paderborn.de/StaffWeb/chi96/ElPub/WWW/chi96www/papers/Koenemann/jk1_txt.htm
2	65	Modern Information Retrieval – Glossary	http://www.sims.berkeley.edu/~hearst/irbook/glossary.html
3	60	JILT: The Revolution in Legal Information Retrieval or: The ...	http://elj.warwick.ac.uk/jilt/99-1/schweigh.html
4	55	Information Retrieval, Homeosemy, Natural Language	http://www.wyle.org/~mfw/diss/node10.html
5	50	IR Projects	http://www.cs.jhu.edu/~weiss/projects.html
6	50	Citation details: Implementation of the SMART Information ...	http://citeseer.nj.nec.com/context/23764/0
7	45	A Model for Multimodal Information Retrieval	http://www.cs.cmu.edu/~benhdj/papers/icme2000_online_paper/icme.html
8	40	Evaluation criteria for information retrieval systems.	http://www.shef.ac.uk/~is/publications/infres/paper62.html
9	30	Text Retrieval Conference (TREC) Data	http://trec.nist.gov/data.html
10	30	INFS 242 – Information Retrieval Systems – Resources	http://skipper.gseis.ucla.edu/faculty/jfurner/00-01/242/242res.html
11	30	TEKTRAN – Technology Transfer Automated Retrieval System	http://www.nal.usda.gov/ttic/tektran/tektran.html
12	25	Air Pollution – Aerometric Information Retrieval System (AIRS)	http://www.epa.gov/airs/
13	25	E-Muni – disclosure documents, news, information and research	http://www.emuni.com/

		relating to the	
14	20	Text Retrieval Conference (TREC) Home Page	http://trec.nist.gov/
15	20	Redirection Page for GNIS U.S. and Territories Query Form	http://www-nmd.usgs.gov/www/gnis/gnisform.html
16	20	Redirection Page for GNIS Antarctic Query Form	http://mapping.usgs.gov/www/gnis/antform.html
17	20	Redirection Page for GNIS U.S. and Territories Query Form	http://mapping.usgs.gov/www/gnis/gnisform.html
18	20	The Information Retrieval in Chemistry	http://macedonia.nrcps.ariadne-t.gr/
19	10	Internet filtering legislation information	http://www.filteringinfo.org/
20	05	Resolution on the Use of Filtering Software in Libraries	http://www.ala.org/alaorg/oif/filt_res.html

Tabela 5.8 – Resultados do primeiro refinamento da terceira iteração de busca.

Nesse refinamento, a *precisão da busca* é de 57%. Todavia, *a precisão da relevância potencial* retornou aos 100%. A baixa precisão da busca continua sendo explicada pelo alto número de documentos retornados que não são relevantes ao contexto da busca.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

> information retrieval relevance query document feedback system user search text term relevant model research ir result collection database base performance

5.6 Quarta Iteração

Essa iteração foi efetuada utilizando-se o *Google*, o *Altavista* e o *Lycos* como provedores de informação. Novamente, espera-se que a maior abrangência da busca diminua sua precisão.

Os critérios de busca utilizados, foram:

[*relevance feedback information retrieval system*]

Os documentos localizados pelo *Fidus,* como resultado desta iteração, estão apresentados na Tabela 5.9:

#	☑	Título	URL
1	60	AN ANALYSIS OF AUDIO, VIDEO, AND IMAGE WEB QUERIES	http://jimjansen.tripod.com/academic/mm00.html
2	50	QUERY REFORMULATION AND RELEVANCE FEEDBACK ANALYSIS	http://jimjansen.tripod.com/academic/cais99.html
3	45	A Study and Analysis of User Queries on the Web	http://jimjansen.tripod.com/academic/pubs/ipm98/ipm98.html
4	40	Information retrieval realted Papers	http://chandrashekarb.tripod.com/PAPER1.HTML
5	40	How Users Search the World Wide Web:	http://jimjansen.tripod.com/academic/pubs/sigirforum98/forum98.html
6	35	Academic Articles on Web Queries, Web Searching, Information Retrieval , Software	http://jimjansen.tripod.com/academic/acad.html
7	35	About Library Associates, Inc.	http://www.angelfire.com/biz6/dhongens/our_company.htm
8	30	Up Coming Events	http://www.angelfire.com/biz6/dhongens/up_coming_events.htm
9	20	Knowledge Management Seminar - 17/8/99	http://www.angelfire.com/zine/qscl/seminars/seminar170899.htm
10	20	ELECTRONIC SUPERVISION SYSTEM	http://mugoma.tripod.com/archive/intoduction.html
11	20	ELECTRINIC SUPERVISION SYSTEM - DIPLOMA IN COMPUTER SCIENCE DISSERTATION	http://mugoma.tripod.com/academic_diploma/dissertation.html
12	15	GENERATION OF MIS REPORT FOR MOTHER DAIRY, GANDHINAGAR	http://www.angelfire.com/id/avi/Finrep.htm
13	05	Untitled	http://toprun-

			info.tripod.com/sitenovo/linkwebsites.htm
14	05	RIPL Services Page	http://www.angelfire.com/ri/radiant/services.html
15	05	RIPL Services Page	http://radiant_info.tripod.com/services.html
16	0	NO TITLE	http://www.angelfire.com/nd/Kamineni/HtmlDocumentLens.html
17	0	NO TITLE	http://michaelpeh.tripod.com/dba.htm
18	0	NO TITLE	http://www.angelfire.com/zine/splatterpattern/txt/part4.txt
19	0	MANAGEMENT ACCOUNTING	http://www.angelfire.com/or/ioriyagami/index.html
20	0	attach3	http://smsu76.tripod.com/attach3.html

Tabela 5.9 – Resultados da quarta iteração de busca.

A *precisão da busca*, nessa iteração, é de 33%, justificada pela expectativa acima apontada. A precisão da relevância potencial é de 100%, ressaltando a efetividade do *Fidus* quando a amostragem de informações julgadas é maior.

Após os passos 3 e 4, o *Fidus* determinou que os termos de maior peso para os documentos relevantes são:

> information retrieval query relevance document feedback system user search term text relevant ir research result number model collection database base

5.7 Comparação do Fidus com o WebMate

Os objetivos principais dos testes comparativos entre o *Fidus* e o WebMate foram três: (1) mostrar que, com poucas iterações (3, no máximo), o *Fidus* seria capaz de gerar termos precisos para os critérios de busca de documentos para uma determinada categoria de um perfil de usuário (com os 15 termos finalmente gerados, se todos os documentos recuperados fossem relevantes a precisão seria de 100%); (2) comparar a precisão do *Fidus* com a do *WebMate*, testando-os com um conjunto de duas ferramentas de busca genérica[17]; e (3) comparar a dependência do *Fidus* com a do *WebMate*, em relação a conjuntos de ferramentas de busca genérica.

Os testes comparativos foram feitos com o *WebMate*, pois esta é uma ferramenta bastante citada na literatura.

A metodologia utilizada para uma das baterias de teste foi a seguinte:

Para uma determinada categoria do perfil de um usuário, e para cada conjunto de ferramentas de busca genérica, {*Alta Vista, Google*, *Radix*} e {*Google*, *Radix*}[18]

1. Sessão no *Fidus*, com até 3 iterações, para determinar os critérios de busca de documentos relevantes;
2. Com os termos gerados após a última iteração, medir a precisão dos critérios de busca (precisão *Fidus*);
3. Submeter ao *WebMate* todos os documentos marcados como relevantes ao longo da sessão no *Fidus*;
4. Extrair os termos de maior peso gerados pelo *WebMate*.
5. Medir a precisão dos critérios de busca com os termos de maior peso (precisão *WebMate*);
6. Comparar a precisão *Fidus* com a precisão *WebMate*.

[17] Este critério excluiu o *Syskyll&Webert* e o *WebSail*.
[18] Algumas alterações foram feitas ao *WebMate* para adição do *Google* e *Radix* como ferramentas de busca.

Comparar a precisão *Fidus* (*WebMate*) para os dois conjuntos de ferramentas de busca genérica.

Para a categoria *Information Filtering*, os resultados dos testes estão na tabela 5.10.

Alta Vista + Google + Radix
Fidus (3 iterações) **Critérios iniciais de busca:** ***information filtering retrieval*** **Vetor final de termos:** ***information filtering user system document retrieval feedback relevance interest query research search model learn relevant*** **Precisão da Relevância Potencial: 91*,7%***
WebMate **Vetor final de termos:** ***information filtering user document system retrieval interest base method article model relevant relevance query profile*** **Precisão: 6*1,2%***
Google + Radix
Fidus (3 iterações) **Critérios iniciais de busca:** ***information filtering retrieval*** **Vetor final de termos:** ***information filtering user system retrieval document model learn method base relevant agent research network interest*** **Precisão: 88*,3%***
WebMate **Vetor final de termos:** ***information filtering user system document retrieval method model article base agent relevant new keyword*** **Precisão: 6*0,1%***

Tabela 5.10: Avaliação experimental do *Fidus*

Os testes mostraram que a precisão do *Fidus* foi maior que a do *WebMate*. Isto parece evidenciar a importância da participação mais efetiva do usuário na determinação dos perfis, como é o caso do *Fidus* (3 iterações, todas com a participação do usuário). O *WebMate* paga o preço de automatizar quase completamente seu processo de determinação de perfis (uma única iteração, com a participação do usuário). Os testes também confirmaram que, quanto maior o conjunto de ferramentas de busca genérica, melhor (o desempenho de {*Alta Vista, Google, Radix*} foi um pouco superior ao de {*Google + Radix*}, tanto para o *Fidus* como para o *WebMate*).

Nos testes com outras 7 categorias, e para vários conjuntos de ferramentas de busca genérica, a precisão do *Fidus* foi sempre superior à do *WebMate*, Em duas categorias, a diferença pró-*Fidus* foi bastante significativa.

5.8 Conclusão

Os testes efetuados com o *Fidus* demonstram que a *precisão da busca* é diretamente afetada pela precisão dos provedores de informação que integram a meta-busca. Quanto maior o número de provedores na meta-busca, maior a abrangência da busca e, consequentemente, menor a precisão global.

O filtro da coleção de informações localizadas pelo *Fidus,* com base na relevância potencial estimada, maximiza a precisão global do sistema. O grau de efetividade desse filtro depende da amostragem de informações julgadas, pois permite ao *Fidus* compor um vetor de termos mais consistente.

Nos testes comparativos entre o *Fidus* e o *WebMate*, diferente daqueles efetuados apenas com o *Fidus*, apenas as informações com alta relevância potencial foram apresentadas aos usuários. Esse filtro de informações por relevância potencial garante que o usuário vê no sistema uma precisão da busca extremamente alta e que, com poucas iterações, a efetividade do *Fidus* é assegurada.

Durante os testes efetuados com o *Fidus,* tornou-se claro que os principais fatores para garantir a eficácia da relevância potencial são o cálculo de similaridades e a manutenção do histórico de informações já analisadas pelo usuário. Esse histórico garante que apenas novas informações serão apresentadas e julgadas.

Os testes demonstram que não apenas os critérios de inclusão têm importância na precisão da busca. Os critérios de exclusão são igualmente relevantes, embora só detectados após uma nova iteração.

O mecanismo de validação de ponteiros do *Fidus* mostrou ser bastante vulnerável às inúmeras formas dos provedores de informação de indicar que uma página HTML não foi encontrada. Diversos ajustes foram feitos durante os testes, compondo oito situações para detectar que uma página não foi encontrada.

Novos testes foram efetuados para outros contextos, como “modelo objeto relacional” e “disseminação seletiva de informações”. Nesses testes, a precisão média da relevância potencial manteve-se acima dos 80% após a quarta iteração de busca com 20 documentos por iteração. Como nesse caso de testes apresentado, a efetividade de busca do *Fidus* é muito superior à efetividade de busca para as ferramentas de busca tradicionais.

Capítulo 6

Conclusões e Perspectivas

Sistemas de Informação Personalizada (SIPs) são um tema de pesquisa ainda incipiente, apesar de sua extrema importância e necessidade. O modelo de recuperação de documentos utilizado pelas ferramentas convencionais de busca de informações na Web, como Altavista, Google e Radix, não atende adequadamente às necessidades de um grande número de usuários - efetividade na busca de informações, manutenção de perfil que defina-os como consumidores de informação e a manutenção do histórico das consultas - que têm interesses específicos em determinadas áreas de pesquisa bibliográfica.

O requisito fundamental - efetividade na busca de informações - não é adequadamente resolvido pelas ferramentas de busca. Essas, normalmente, assumem políticas de busca genéricas, centradas na informação e seus atributos - números de acesso e ocorrências nas diversas páginas da Web -, sem considerar particularidades dos usuários.

Em conseqüência, o seguinte cenário indesejável é uma constante: um usuário solicita um pequeno conjunto de documentos (muitas vezes, um único documento) de um determinado assunto; ao invés disto, ele recebe uma montanha de documentos que pouco ou nada tem a ver com o que foi solicitado — com muita sorte, e depois algum tempo perdido, ele poderá encontrar seu(s) documento(s).

Nossa proposta de um SIP, *Fidus*, vem ao encontro da necessidade dos usuários-Web de acesso seletivo e preciso a informações que atendam a seus reais interesses. Seu principal requisito, que julgamos cumprido, é a interação com o usuário para avaliar e escolher critérios precisos de busca de documentos, segundo o perfil de consumidor de informação do usuário.

A avaliação experimental do *Fidus*, frente a conhecidos protótipos de pesquisa em SIPs, evidenciou a sua superioridade em termos da precisão das buscas a documentos de interesses particulares dos usuários. Essa superioridade é provida, em grande parte, pela implementação do *TF-Seno* como algoritmo de determinação de pesos. Esse algoritmo

mostra-se mais eficiente, para os requisitos do *Fidus*, que o algoritmo mais conhecido da literatura – TF-IDF.

A arquitetura de meta-busca utilizada para a construção do *Fidus* garante seu crescimento incremental através da construção de novos componentes especializados de busca. Assim, qualquer ferramenta de busca existente – *Altavista, Yahoo, Radix* - é provedor potencial de informações ao *Fidus*, garantindo-lhe abrangência virtualmente ilimitada na busca de informações armazenadas na Web, em contrapartida à maioria de ferramentas de busca conhecidas.

Em termos de perspectivas do trabalho, temos em vista o seguinte. No *Fidus*, os perfis dos usuários são estruturados por categoria de documentos (do nosso conhecimento, ele é o único que faz assim, todos os outros perfis sendo estruturas planas) — para cada categoria, são definidos os critérios de busca de documentos relevantes da mesma. As vantagens, para um usuário, de ter um perfil estruturado são evidentes: ele pode solicitar ao *Fidus* que selecione documentos relevantes para ele de uma categoria; em seguida, ele pode querer refinar a seleção por subcategoria(s) da categoria, depois por subcategoria(s) da(s) subcategoria(s), em seguida voltar para a categoria, etc., tudo isso de maneira muito confortável. Atualmente porém, a determinação dos critérios de busca de documentos relevantes das categorias não leva em conta a estrutura das categorias.

O que se faz necessário é investigar em que medida os critérios de busca de uma categoria podem ajudar na definição dos critérios de busca de suas subcategorias, ou vice-versa, dentro de um procedimento recursivo. Esta abordagem poderá vir a ser uma importante otimização do algoritmo de definição dos critérios de busca de documentos das categorias.

Esse trabalho nos permite identificar alguns desafios futuros no sentido de aperfeiçoar e estender o conceito de personalização de informações e de relevância potencial e efetiva da informação.

A cooperação dos usuários em grupos de pesquisa, mantendo interesses comuns por informações na mesma área de pesquisa bibliográfica, é um forte subsídio para avaliação de como interesses comuns são representados por usuários distintos. O método utilizado pelo *Fidus* para a construção de perfis, personalizando informações, pode evoluir com esse estudo para além da representação hierárquica de categorias. O suporte a ontologias para a

descrição e recuperação de informações é uma perspectiva bastante viável ao Fidus, permitindo-o categorizar informações com grande precisão.

Com a interação desse usuários com o *Fidus*, pode-se analisar o histórico das consultas dos usuários e da avaliação dos mesmos sobre as informações recuperadas. Desvios acentuados na análise global da relevância potencial pode indicar a necessidade do refinamento do algoritmo de cálculo de similaridades. Uma evolução visível sobre esse algoritmo é computar a análise da relevância efetiva dos diversos usuários, sobre uma determinada informação, no cômputo da relevância potencial dessa informação para outro usuário. Estabelecemos, com isso, o filtro cooperativo de informações.

Outros trabalhos ainda são necessários de modo a incluir outras características na solução proposta, inclusive para melhorar o atendimento a alguns requisitos levantados.

Em primeiro lugar, é preciso especificar um dicionário controlado de termos para o *Fidus*, possibilitando a representação de documentos e contextos como vetores de igual dimensão. Pode-se usar um dicionário de termos já pronto, compondo-o à arquitetura do *Fidus*.

Muitos algoritmos para o cálculo de similaridades - detalhados na literatura [ZHA 9?], [LIU 00] - trabalham com vetores representados assim. Com a adição de um dicionário de termos ao *Fidus* cria-se a possibilidade de troca de algoritmos de cálculo de similaridades e de simulação de resultados para avaliação de diferentes métodos para estimar a relevância potencial das informações recuperadas.

Em seguida, para permitir maior controle do usuário sobre o sistema e maior ênfase na personalização, o requisito de segurança precisa ser estendido.

A inclusão de múltiplas formas de identificação e autenticação do usuário, associando-as a ferramentas de busca específicas, maximiza a potencialidade de uma ferramenta de busca tornar-se provedora do *Fidus*. Desta forma, ferramentas que necessitam de autenticação, como a ferramenta de busca da ACM (www.acm.org), podem fazer parte do universo de busca do *Fidus*. Para isso, cada usuário precisa ter seu nome de usuário e senha cadastrado naquela ferramenta.

Finalmente, diversos requisitos de personalização precisam ser criados. O Fidus deve permitir ao usuário escolher:

1. O percentual mínimo de relevância potencial para a filtragem de informações apresentadas nos resultados. Assim, um percentual mínimo de 0% indica que o *Fidus* deve apresentar todas as informações recuperadas.

2. O tamanho da coleção de resultados, especificando quantas informações novas devem ser apresentadas a cada consulta ao Fidus.

3. Escolher os provedores que vão integrar a meta-busca.

Como um sistema extremamente flexível e extensível, tais alterações são facilmente incorporáveis ao *Fidus*.

Bibliografia

[ALM ??] ALMGREN, Magnus e BERGLUND Jenny, *Email Classification with TF-IDF in a Naive Bayesian Framework with Handcrafted Features,* Disponível em: http://www.stanford.edu/~almgren/project.html. [Capturado em 20 de nov de 2000].

[BAC 91] BACLACE, Paul E. *Personal information intake filtering.* Bellcore Information Filtering Workshop, Nov. 1991. Disponível: http://www.baclace.net [capturado em 25 de mar. 2001].

[BEL 92] BELKIN, Nicholas J., CROFT, Bruce W. *Information filtering and information retrieval: two sides of the same coin?* Communications of the ACM [online], vol. 35, n. 12, dez. 1992.

[BEL 93] BELKIN, Nicholas J. *Interaction with texts: information retrieval as information-seeking behavior*. In: Information retrieval '93. Von der Modellierung zur Anwendung. Konstanz: Universitaetsverlag Konstanz, p. 55-66. Disponível: ftp://scils.rutgers.edu/pub/belkin/papers/gi_ir93.ps [capturado em 3 abr. 2001].

[BEL 96] BELKIN, Nicholas J. *Intelligent information retrieval: whose Intelligence?* In: Proceedings of the Fifth International Symposium for Information Science [online], Konstanz Alemanha, Universtaetsverlag Konstanz, p. 25-31. Disponível: http://www.scils.rutgers.edu/tipster3/iirs.html [capturado em 3 abr. 2001].

[BEL 98] BELKIN, Nicholas J. *An overview of results from Rutgers investigations of interactive information retrieval.* In: 34th Annual Clinic on Library Applications of Data Processing [online], Champaign-Urbana USA, University of Illinois School of Library and Information Science, p. 45-62. Disponível: http://www.scils.rutgers.edu/tipster3/cladp97.html [capturado em 3 abr. 2001].

[BOO 99] BOOCH, G., RUMBAUGH, J., JACOBSON, I. *The unified modeling language user guide*, Addison-Wesley, 1999.

[CHE 00] CHEN, Zhixiang. et al. *Websail: from on-line learning to Web search.* Disponível: http://citeseer.nj.nec.com/chen00websail.html [capturado em 4 abr. 2001].

[CHE 98] CHEN, L., SYCARA, K. *WebMate: a personal agent for browsing and searching.* In: Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems [online], Minneapolis, p. 10-13, mai. 1998. Disponível: http://citeseer.nj.nec.com/chen98webmate.html [capturado em 3 abr. 2001].

[COM 95] Microsoft Corporation and Digital Equipment Corporation, *The Component Object Model Specification*, Draft Version 0.9, October 24, 1995

[CRE 95] CRESTANI, Fabio, van RIJSBERGEN, Cornelis J. *Information retrieval by logical imaging.* Journal of Documentation, v.51, n.1, p. 3-17, mar. 1995. Disponível: http://ir.dcs.gla.ac.uk/pubauthor/fabio.html [capturado em 3 abr. 2001].

[CRO 95] CROFT, W. Bruce. *What do people want from information retrieval?,* D-Lib Magazine [online], nov. 1995. Disponível: http://www.dlib.org/dlib/november95/11croft.html [capturado em 3 abr. 2001].

[FRE 99] FRENCH, James C., VILES, Charles L. *Personalized information environments: an architecture for customizable access to distributed digital libraries.* D-Lib Magazine [online], 1999. Disponível: http://www.dlib.org/dlib/june99/french/06french.html [capturado em 3 abr. 2001].

[FUC 98] FUCHS, Mathew. *Building an information system from the web.* In: Proceedings of the 31st Hawaii International Conference on System Sciences

(HICSS'98), 1998. IEEE Computer Society [online]. Disponível: http://citeseer.nj.nec.com/85183.html [capturado em 3 abr. 2001].

[GAM 95] GAMMA. et al. *Design Patterns: elements of reusable object-oriented software.* Addison-Wesley, 1995.

[GLO 01] GLOVER, Eric J. et al. *Improving category specific Web search by learning query modifications.* Symposium on applications and the Internet, SAINT 2001, San Diego, Califórnia. IEEE Computer Society [online] p. 23-31, jan. 2001.

[GOF 99] GOFFINET, Luc, NOIRHOMME-FRAITURE, Monique. *Automatic hypertext link generation based on similarity measures between documents.* Disponível: www.fundp.ac.be/~lgoffine/Hypertext/semantic_links.html [capturado em 3 abr. 2001].

[HAR 93] HARDY, Darren R., SCHWARTZ, Michael F. *Essence: a resource discovery system based on semantic file indexing.* In: Proceedings of the USENIX Winter Conference, San Diego, 1993. Disponível: http://citeseer.nj.nec.com/hardy93essence.html [capturado em 3 abr. 2001].

[HEA ??] HEARST, Marti A., PLAUNT, Christian, *Subtopic Structuring for Full-Length Document Access,* In: Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1993, pp. 59-68**.** Disponível: www.sims.berkeley.edu/~hearst/papers/subtopics-sigir93/sigir93.html. [Capturado em 20 de nov de 2000].

[HUA 9?] HUANG, Lan. *A survey on Web information retrieval techniques.* Disponível: http://citeseer.nj.nec.com/336617.html [capturado em 4 abr. 2001].

[KLI ??] KLINKENBERG, Ralf, RENZ, Ingrid. *Adaptative information filtering: learning drifting concepts*.

[KOB 99] KOBAYASHI, Mei, TAKEDA Koichi. *Information retrieval on the Web: selected topics*.

[KHO 99] KHOSHAFIAN, S, et al., *The Jasmine Object Database*, Morgan KaufMann, 1999.

[LAR 97] LARSEN, Ronald L. *Relaxing assumptions... stretching the vision.* D-Lib Magazine [online], Abril 1997. Disponível: http://www.dlib.org/dlib/april97/04larsen.html [capturado em 3 abr. 2001].

[LAW 00] LAWRENCE, Steve. *Context in Web search*. IEEE Data Engineering Bulletin, v. 23, n. 3, p. 25-33, 2000.

[LIU 00] LIU, King-Lup, MENG, Weiyi, YU, Clement. *Discovery of similarity computations of search engines*. Disponível: http://citeseer.nj.nec.com/267965.html [capturado em 4 abr. 2001].

[MEN 9?] MENG, Weiyi, YU, Clement, LIU, King-Lup. *Building eficient and efective metasearch engines.* Disponível: www.csam.iit.edu/~wsearch/p217-yu.pdf [capturado em 3 abr. 2001].

[PAZ 9?] PAZZANI, Michael, MURAMATSU, Jack, BILLSUS, Daniel. *Syskill & Webert: identifying interesting web sites.* Disponível: http://www.ics.uci.edu/~pazzani/Syskill.html [capturado em 3 abr. 2001].

[PEN 98] PENKER, Eriksson. *UML Toolkit.* John Wiley, 1998.

[POL 97] POLLOCK, Annabel, HOCKLEY, Andrew. *What's wrong with Internet searching.* D-Lib Magazine [online], mar. 1997. Disponível: http://www.dlib.org/dlib/march97/bt/03pollock.html [capturado em 3 abr. 2001].

[RUM 99] RUMBAUGH, J., BOOCH, G., JACOBSON, I. *The unified modeling language reference manual*, Addison-Wesley, 1999.

[SAL 83] SALTON, Gerard. *Introduction to modern information retrieval.* McGraw-Hill, 1983.

[SUM 98] SUMMER Jr., Robert G., Kiduk Yang, DEMPSEY, Bert J. *An interactive WWW search engine for user-defined collections.* In: Proceedings of the 3rd ACM International Conference on Digital Libraries [online], jun. 1998.

[TOR 98] TORRISSEN, Bjorn Christian. *Dewey goes surfing: agent-based information retrieval and classification support.* Tese (Doutorado em Informática) - Department of Computer and Information Science, Trondheim, Fev 1998. Disponível: http://www.pvv.ntnu.no/~bct/thesis/frontpage.html [capturado em 3 abr. 2001].

[ZHA 9?] ZHANG, Dell, DONG, Yisheng. *An efficient algorithm to rank web resources*. Disponível: http://www9.org/w9cdrom/251/251.html [capturado em 3 abr. 2001].

Apêndice A

Jasmine

A.1 Introdução

Jasmine é um SGBDOO, manipulável através da linguagem ODQL ("Object Database Query Language"). ODQL não é compatível com o padrão ODMG.

Ao contrário da linguagem ODMG-OQL, que é somente declarativa, ODQL tem também uma parte "*procedural*", permitindo a definição de variáveis, o controle de fluxo e laços de execução. Pode ser usada em combinação com linguagens hospedeiras, como Java e C++, para a implementação de aplicações ou construção de métodos.

No Jasmine, a definição do esquema do banco e o acesso aos objetos são feitos através da ODQL. O correspondente a ODL, da ODMG, é conseguido através de comandos para definição de classes, métodos, etc., demonstrados a posteriori.

Para o acesso aos objetos, a ODQL incorpora fracamente os aspectos declarativos da linguagem SQL. O seu comando ***Select***, que retorna sempre uma coleção (***Set, List, Bag*** ou ***Array***), não permite eliminação de duplicatas, subconsulta, ordenação, quantificadores, expressões com group by ou agregações, como a OQL. Para isto, são usados métodos das coleções, como ***average, max, min, sort***, etc.

Operações de alteração de dados são implementadas através de métodos.

A.2 Conceitos do Jasmine

A2.1 Biblioteca de Classes

O Jasmine fornece um conjunto de classes do sistema para tipos de dados complexos, como:

- Bag, Set, List e Array, com métodos para a manipulação de coleções.
- MMFile, que pemite a manipulação de dados multimídia. MMFile é superclasse de classes como Acrobat, Animation, Audio, Bitmap, Excel, Movie, Picture, PowerPoint, WinWord, etc.

A2.2 Propriedades de Classes e de Objetos

No Jasmine, as propriedades podem ser de classe ou de objeto - instância.

As propriedades de instância mantêm um valor individual para cada instância de uma classe e suas subclasses, enquanto as propriedades de classe mantêm um valor único para todas as instâncias da classe e subclasses. Propriedades de classe correspondem a propriedades ***static***, em Java.

Na seqüência, explicamos, através de vários exemplos, como definir uma classe-Jasmine. O primeiro exemplo trata somente de atributos de uma classe.

```
1.  defineClass Usuario super: Composite {
2.  class:
3.     String descricao default: "Classe de usuarios";
4.  instance:
5.     String nome default: NIL;
6.     Set <String> mail default: Set{};
7.     String login unique: mandatory: ;
8.     String senha default: NIL;
9.  };
```

Nos exemplos ilustrativos, as palavras reservadas do Jasmine são destacadas em itálico e azul.

A linha 1 define uma classe *Usuario*, subclasse de *Composite*.

Composite é semelhante à classe *Object*, do Java. É a classe base de todas as classes do usuário e define as operações básicas necessárias a uma classe ou uma instância, como *new – para* criar uma nova instância numa classe -, *delete – para* remover uma classe ou instância -, *print* - imprime as propriedades de um objeto -, etc.

De um modo geral, uma subclasse herda todas as propriedades de classe e de instância da superclasse.

As linhas 2 e 4 determinam o nível dos atributos.

A linha 3 define um atributo de classe com nome *descricao*, do tipo *String*, com um valor default. As linhas 5 a 8 definem vários atributos de instância.

A linha 6 define o atributo *mail*, que é uma coleção – *Set* - de strings. A classe *Set* define uma coleção não ordenada e sem elementos duplicados.

Na linha 7, a cláusula *unique* define que a propriedade *login* terá valor único entre todas as instâncias da classe. A cláusula *mandatory* define que toda instância da classe deve ter a propriedade *login* não nula.

Valores nulos, no Jasmine, são representados pela palavra reservada *NIL*.

O próximo exemplo demonstra o acesso a atributos**:**

```
1.  Usuario u;
2.  String s;
3.  ...
4.  s = u.nome;
5.  s.print();
```

A linha 1 define uma variável ***u*** do tipo *Usuário*, e a linha 2, a variável ***s*** do tipo *String*.

A linha 4 atribui à variável ***s*** o valor da propriedade *nome* da instância de *Usuário* apontada por ***u*.**

A linha 5 imprime o conteúdo da variável ***s***, através do método polimórfico *print*. Toda classe do Jasmine dispõe do método *print* para impressão de suas instâncias. As classes do usuário herdam este método de *Composite*.

A2.3 Métodos

Assim como para os atributos, pode-se definir métodos de classe e de instância. Os métodos de classe, assim como métodos declarados "*static*" em Java, operam sobre toda a classe, não necessitando de uma instância para sua execução. São, então, usados para a manipulação de coleções de objetos.

No exemplo seguinte, apresentamos a definição da classe *CategoriaPerfilavel* como subclasse de *CategoriaDeInformacao*.

```
1.    defineClass CategoriaPerfilavel super: CategoriaDeInformacao {
2.    class:
3.          CategoriaPerfilavel copia(CategoriaDeInformacao categoria);
4.          CategoriaPerfilavel novo(CategoriaDeInformacao categoria);
6.    instance:
7.          Integer prioridade default: 0;
8.          Void setPrioridade(Integer prioridade);
      };

10.   defineProcedure Void CategoriaPerfilavel::instance:setPrioridade(Integer
                                                          prioridade) {
11.     $self.prioridade = prioridade;
12.     $return;
      };

13.   defineProcedure CategoriaPerfilavel
            CategoriaPerfilavel::class:copia(CategoriaDeInformacao categoria) {
14.         $CategoriaPerfilavel cp;
            ...
15.         $cp = CategoriaPerfilavel.novo(categoria);
            ...
16.         $return (cp);
      };
```

As linhas 3 e 4 definem as assinaturas de métodos de classe, enquanto a linha 8 define método de instância.

Os métodos de classe, *copia* e *novo*, recebem um parâmetro do tipo *CategoriaDeInformacao* e retornam um objeto do tipo *CategoriaPerfilavel*. O método

copia faz uma cópia da categoria passada como parâmetro, retornando um objeto novo - criado com o método *novo* -, porém com todas as propriedades iguais às do parâmetro.

O método de instância *setPrioridade* manipula o atributo de instância *prioridade.* A palavra reservada *Void* indica que o método não retorna valor.

A linha 10 define o corpo do método de instância em ODQL, atribuindo prioridade a uma *categoria perfilável*. O **$** é usado em toda linha do método escrita em ODQL; a ausência do $ indica que a linha está escrita na linguagem C/C++.

De 13 a 16, temos o extrato do corpo do método de classe, *copia*.

O próximo exemplo demonstra o acesso a métodos:

```
1.    CategoriaDeInformacao ci;
2.    CategoriaPerfilavel cp;
3.    Set <CategoriaDeInformacao> conjCat;

4.    conjCat   =  select  CategoriaDeInformacao  from  CategoriaDeInformacao
alone
                        where CategoriaDeInformacao.nome == "OO";
5.    scan(conjCat, ci) {
6.          cp = CategoriaPerfilavel.copia(ci);
7.          cp.setPrioridade(10);
8.    };
```

A linha 3 declara uma variável-coleção - conjunto de objetos do tipo *CategoriaDeInformacao*.

A linha 4 ilustra o uso da cláusula *alone* em um *Select*. De um modo geral, ao ser executado um *Select* numa classe que possui subclasses, todas as instâncias concernentes da hierarquia - classe e suas subclasses - são recuperadas, efetuando-se a devida conversão de tipos. Esta conversão armazena, na coleção resultado, objetos do tipo da superclasse, eliminando propriedades e métodos especializados ou refinados. A restrição da seleção apenas à classe é feita através da cláusula *alone* do *Select*.

ODQL não possui o conceito de *extent* de classe como em OQL. O *Select*, em ODQL, é efetuado no nome da classe.

As linha 5 – 8 selecionam cada elemento da coleção através do comando *scan.* O comando *scan* permite que os elementos de uma coleção sejam recuperados, um a um, e então, manipulados. A interpretação algorítmica destas linhas é a seguinte:

1. Selecionar cada elemento da variável-coleção *conjCat*;

2. Atribuir cada elemento à variável *ci*.
3. Manipular a variável *ci*, que referencia uma *CategoriaDeInformacao*.

A2.4 Herança Múltipla

Todas as classes do Jasmine, incluindo classes do usuário, podem possuir várias superclasses. Herança múltipla surge naturalmente quando você tem duas classificações ortogonais dos mesmos objetos.

Ambigüidades em Herança Múltipla

Vamos definir *ambigüidade* em herança múltipla, e como resolver o potencial conflito, por meio de um exemplo.

```
1.    defineClass Mamifero super: Composite {
2.    instance:
3.          Integer peso default: NIL;
4.          Integer numeroDeMamas default: NIL;

5.          Integer getPeso();
6.          Integer getNumeroDeMamas();
      };

7.    defineClass Oviparo super: Composite {
8.    instance:
9.          Integer peso default: NIL;
10.         Integer tamanhoDoOvo default: NIL;

11.         Integer getPeso();
12.         Integer getTamanhoDoOvo();
      };

13.   defineClass Ornitorrinco super: Mamifero, Oviparo {
14.   instance:
15.         Integer peso default: NIL;

16.         Integer getPeso();
      };
```

As linhas 3 e 9 definem um atributo de mesmo nome, *peso*, que será herdado pela classe *Ornitorrinco*. Como isto gera uma ambigüidade, o atributo deve ser *refinado*, isto é, definido na subclasse. O mesmo acontece com os métodos das linhas 5 e 11, resultando que eles são refinados - linha 16.

A2.5 Integridade Referencial

O Jasmine não suporta o conceito de integridade referencial. Entretanto, ele pode ser facilmente implementado.

O exemplo seguinte apresenta o método de remoção de uma instância da classe *Perfil.* A classe *Perfil* possui uma propriedade-coleção *categoriaPerfilavel*, que representa um relacionamento de agregação com objetos da classe *CategoriaPerfilavel*. Logo, a remoção de uma instância da classe *Perfil* exige que cada elemento da coleção, referenciando uma *CategoriaPerfilavel*, seja também removido.

```
1.    defineProcedure Void Perfil::instance:delete() {
2.          $CategoriaPerfilavel cp;
3.          $Set<CategoriaPerfilavel> scp;
4.          $scp = self.getCategoriaPerfilavel();
5.          $if (scp.count() != 0) {
6.                $scan( scp, cp ) {
7.                      $cp.delete();
8.                };
9.          };
10.         $self.super::delete();
11.    $return;
       };
```

A linha 1 define o método de instância *delete*, que é uma sobrecarga do método *delete* da classe *Composite*.

Este método varre a coleção de objetos da classe *CategoriaPerfilavel* que são dependentes do *Perfil* a ser excluído, removendo-os - linhas 2 a 7. A linha 4 faz com que a variável-coleção *scp* referencie a coleção de todas as categorias do *perfil* corrente.

A linha 10 executa a chamada ao método de remoção da superclasse *Composite*, para remover a instância corrente. *Self* é a palavra reservada do Jasmine para se obter uma referência à instância corrente. *Super* é a referência à superclasse.

A.3 Implementação de Métodos no Jasmine

Com ODQL

setNome é um método de instância para atribuir valor à propriedade **nome,** da classe Usuário.

```
1.  defineProcedure Void Usuario::instance:setNome(String nome)  {
2.   $self.nome = nome;
3.   $return;
     };
```

A linha 2 atribui à propriedade nome da instância referenciada – self – o valor do parâmetro nome.

removeHierarquia é o método que remove as subcategorias de uma categoria excluída..

```
1. defineProcedure Void CategoriaDeInformacao::instance:removeHierarquia(Set
                        <CategoriaDeInformacao> colecaoCategorias)  {
2.      $CategoriaDeInformacao categoria;
3.      $Set <CategoriaDeInformacao> colecaoCategoriasInferiores;

4.      $scan(colecaoCategorias, categoria) {
5.              $if (categoria.getCategoriaInferior().count() != NIL) {
6.                      $colecaoCategoriasInferiores = categoria.getCategoriaInferior();
7.                      $self.removeHierarquia(colecaoCategoriasInferiores);
                }
8.              $categoria.delete();
        };
9.      $return;
};
```

Com ODQL e C/C++

As definições sem o símbolo “$” estão em C/C++.

O método *getOID* retorna o OID de uma instância da classe CategoriaPerfilavel. O retorno do OID está no formato: “NomeDaFamíliadeClasses:ClasseID:InstânciaID”

```
1. defineProcedure String CategoriaPerfilavel::instance:getOID () {
2.  $CategoriaPerfilavel cp <cp_val, cp_stat>;
3.  $String ret <ret_val, ret_stat>;
4.  char tmp[100];
```

```
5. ret_stat = ODB_STATVALID;
6. $cp = self;
7. sprintf(tmp,"%s:%ld:%ld",cp_val.cfname,cp_val.cno,cp_val.ino);
8. ret_val = tmp;
9. $return (ret);
};
```

A linha 2 define a variável ODQL ***cp*** do tipo *CategoriaPerfilavel* e, automaticamente, as variáveis C/C++ ***cp_val*** e ***cp_stat***.

O tipo de ***cp_pval*** é *ODB_OBJECT_ID*. Essa é uma estrutura definida com elementos que representam a classe, número da classe e número da instância, provendo um modo de acessar os identificadores do objeto nas linguagens hospedeiras.

As linhas 4, 5, 7 e 8 estão escritas na linguagem C/C++.

A.4 Manipulação de objetos com ODQL

O código seguinte insere, através do método *new*, definido e herdado de *Composite*, novas categorias de informação e, para cada uma, critérios de busca associados. No método *new*, colocam-se os nomes de todos os atributos mandatórios e seus respectivos valores.

```
1. CategoriaDeInformacao c1, c2;
2. c1 = CategoriaDeInformacao.new(nome:="Linguagens de Consulta para
   Documentos XML", descricao:="LCD-XML", criterioDeBusca := "query language
   document xml" ,categoriaInferior:=Set{});


   c2 = CategoriaDeInformacao.new(nome:="Linguagens de Consultas a XML",
   descricao:="LC-XML", criterioDeBusca := "query language xml" ,
   categoriaInferior:=Set{});
```

O código seguinte apresenta a seleção de objetos de uma classe. Serão selecionados todos os objetos da classe *CategoriaDeInformacao*, sem selecionar objetos das subclasses (cláusula *alone*). A coleção resultante será percorrida pelo método *scan*.

```
CategoriaDeInformacao ci;                    Definição de variáveis.
Set <CategoriaDeInformacao> lci;

lci = select c from CategoriaDeInformacao  alone c;        Seleção de objetos.

scan(lci, ci) {                              Iteração na coleção.
        ci.getNome().print();
};
```

O código seguinte apresenta a seleção de uma categoria perfilável específica e a atualização do valor de uma propriedade. A coleção será percorrida através de um *Iterador*. O *Iterador* no ODQL funciona como em outras linguagens OO.

```
CategoriaPerfilavel ci;
Set <CategoriaPerfilavel> sci;
Iterator <CategoriaPerfilavel> it;           Define it como sendo um iterador
                                             para coleções de CategoriaPerfilavel.

sci  = select c from CategoriaPerfilavel c where c.nome == "Jasmine";
it = sci.createIterator();                   Cria o iterador para a coleção sci.

while (it.advance()) {                       Percorre a coleção através do método
        ci = it.get();                       advance do Iterador.
        ci.prioridade = 15;                  A propriedade pode ser alterada diretamente
        ci.setPrioridade(15);                ou via um método.
        ci.print();
};
```

Apêndice B

Lista de StopWords

B.1 Introdução

Stopwords são termos considerados irrelevantes para a determinação ou identificação do contexto dos documentos, pois tais termos não são específicos o suficiente para representá-los. Artigos, preposições e demais termos inerentes à linguagem, além de termos que aparecem em muitos documentos, são exemplos de *stopwords*.

A construção da lista de *stopwords* pode ser manual, onde um especialista no sistema seleciona que palavras devem ser descartadas, ou automática. Nesse caso, o sistema deve monitorar quais os termos que aparecem em maior freqüência na maior quantidade dos documentos. Estes termos têm pouco poder de seletividade entre os documentos e, por isso, podem ser considerados como "*stopwords*".

B.2 Lista de *StopWords* do *Fidus*

A coleção de termos que compõem a lista de *stopwords* do *Fidus* foi elaborada, inicialmente, de forma manual. Em decorrência das diversas iterações de testes efetuadas – como apresentado no capítulo 5 – os termos menos significativos identificados com alto peso passaram a pertencer à essa lista.

A lista de *stopwords* do *Fidus* também contém símbolos, como sinais de pontuação, parênteses, etc., pois o *Fidus* considera como um *termo do documento* qualquer conjunto de caracteres contíguos separados por espaço.

Os termos ou símbolos mantidos nessa lista são:

-	tt	acm	"
pre	new	color	#

$
%
(
)
*
,
.
:
;
@
>
0
1
2
3
4
5
6
7
8
9
a
able
about
above
accord
according
accordingly
across
actually
address
after
afterward
afterwards
again
against
al
align
all
allow
almost
alone
along
already
also
alt
although
always
am
among
amongst
amp
an
and
another
any
anybody
anyhow
anyone
anything
anyway
anyways
anywhere
apart
appear
arc
are
aren
arial
around
as
aside
ask
at
available
away
awfully
b
back
bas
be
became
because
become
becomes
becoming
been
before
beforehand
began
behind
being
below
beside
besides
best
better
between
beyond
big
blockquote
body
book
border
both
bottom
br
but
by
c
came
can
cann
cannot
cause
causes
center
certain
click
co
colspan
com
come
comes
comment
concern
consequently
consider
contact
contain
copy
copyright
corp
corporate
correspond
could
course
currently
d
date
dc
dd
de
de
describe
despite
di
did
didn
do
does
doesn
don
done
down
downwards
dt
during
e
each
edu
eg
either
else
elsewhere
em
email
en
enough
error
especially
et
etc
even
ever
every
everybody
everyone
everything
everywhere
ex
example
except
f
face
far
feet
few

fewer
finally
find
following
font
foot
for
former
formerly
found
from
ft
further
furthermore
g
generally
get
gets
getting
gif
give
given
gives
go
goes
going
gone
got
gotten
greetings
gt
guest
h
h1
h2
h3
had
hadn
happen
hardly
has
hasn
have
haven
having
he
head
height
hello
helvetica
hence
henceforth
her
here
hereafter
hereby
herein
hereupon
hers
herself
hi
him
himself
his
hither
home
homepage
how
howbeit
however
hr
href
htm
html
html
http
i
ie
if
ignored
il
image
img
in
inasmuch
inc
include
included
includes
including
indeed
indicate
inner
insofar
instead
internet
into
inward
is
isn
it
its
itself
j
jpg
just
k
keep
keeps
kept
key
know
known
knows
l
last
later
latter
latterly
least
less
lest
let
li
like
liked
likely
link
little
ll
look
lt
ltd
m
made
mail
make
maken
many
may
maybe
me
mean
meanwhile
merely
meta
might
miss
more
moreover
most
mostly
move
much
must
my
myself
n
name
namely
nbsp
nd
near
nearly
need
needs
neither
net
never
nevertheless
news
next
nj
nl
no
nobody
non
none
nonetheless
noone
nor
not
nothing
now
nowhere
o
of
off

often
oh
ok
okay
on
once
one
ones
only
onto
open
or
other
others
otherwise
ought
our
ours
ourselves
out
outside
over
overall
own
p
page
pages
paper
part
particularly
parts
pdf
people
per
perhaps
place
please
plus
pm
possible
pp
previous
probably
provide
provides
pub
put
q
que
quite
quot
qv
r
rather
rd
rdf
re
regard
regardless
rethink
s
said
same
sans-serif
saw
say
script
sdi
see
seeing
seem
seemed
seeming
seemingly
seems
seen
self
selves
send
sent
set
several
shall
she
should
shouldn
sign
similar
since
site
sites
six
size
size
so
some
somebody
somehow
someone
something
sometime
sometimes
somewhat
somewhere
soon
sorry
src
still
such
sup
sure
t
take
taken
talk
tb
td
tel
tell
th
than
thank
thanks
thanx
that
thats
the
their
theirs
them
themselves
then
thence
thenceforth
there
thereafter
thereby
therefore
therein
theres
thereupon
these
they
this
those
though
three
through
throughout
thru
thus
time
tm
to
together
too
took
top
toward
towards
tr
tried
tries
try
trying
two
u
ul
un
under
unfortunately
unless
unlike
unlikely
until
untitl
untitle
unto
up
upon
url
us
use
used
useful
uses
using
usually

uucp
v
valign
value
various
ve
verdana
verdana
version
very
via
view
visit
viz
vol
vs
w
want
wants
was
wasn
way
we
web
webmaster
website
welcome
well
went
were
weren
what
whatever
when
whence
whenever
where
whereafter
whereas
whereby
wherein
whereupon
wherever
whether
which
while
whither
who
whoever
whole
whom
whomever
whose
why
wide
width
will
wish
with
within
without
wonder
would
wouldn
www
x
y
year
yes
yet
you
your
yours
yourself
yourselves

B.3 Conclusão

Embora a lista de *stopwords* contenha apenas termos em inglês, termos de qualquer idioma podem ser acrescentados para permitir a correta elaboração do vetor de termos para documentos escritos nesse idioma.