COMPAGENT: UMA FERRAMENTA PARA O APOIO À BUSCA E RECUPERAÇÃO DE INFORMAÇÕES ORIENTADAS A DOMÍNIO NA WEB

Marcelo Nascimento Costa

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Marta Lima de Queirós Mattoso, D.Sc.

Prof. Cláudia Maria Lima Werner, D.Sc.

Prof. Regina Braga Maciel Braga Villela, D.Sc.

Prof. Ana Maria Carvalho Moura, Dr.

RIO DE JANEIRO, RJ – BRASIL

DEZEMBRO DE 2002

COSTA, MARCELO NASCIMENTO

CompAgent: Uma ferramenta para o apoio a recuperação de informações orientadas a domínio na Web [Rio de Janeiro] 2002

XI, 130 p., 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2002)

Tese – Universidade Federal do Rio de Janeiro, COPPE

1. Recuperação de Informações na Web
2. Filtragem de Informações
3. Projeto Odyssey

I. COPPE/UFRJ II. Título (série) ii

aos meus pais

Agradecimentos

Aos meus pais, por todo apoio que eles me deram por toda a minha vida.

Às Professoras Cláudia Werner e Marta Mattoso, pela orientação constante, pela paciência em conduzir esse trabalho, estando sempre dispostas a me ajudar sempre que me foi preciso. Tenho uma grande admiração pela simplicidade delas para orientarem e se relacionarem com as pessoas, ressaltando ainda a imensa vocação acadêmica. Eu as considero um exemplo de profissional para a academia.

À Professora Regina Braga, que além de ter aceitado participar da minha banca, forneceu as idéias iniciais e acreditou nesse trabalho desde o seu princípio. Além de ter tido o prazer de poder compartilhar com a sua amizade.

À Professora Ana Maria Moura, por ter aceitado participar da minha banca.

Ao Professor Jano Souza, por viabilizar a minha vinda ao mestrado, acreditando no meu potencial para ingressar na vida acadêmica.

Ao amigo Robson Pinheiro, por trabalharmos juntos esses anos todos, sendo uma pessoa extremamente competente e com idéias bastantes práticas. Não é somente um colega da academia, mas sim um amigo que eu pude contar em todas as horas, desde os momentos profissionais, acadêmicos até os pessoais.

Aos amigos Alessandréia Oliveira, André Victor, Gustavo Pinto, Marcos Kalinowski, Paulo Kusano Ferrari e José Marcelo Filho que serão amigos para o resto da vida. Eu os considero como amigos de verdade, tanto para os momentos alegres quanto para os momentos difíceis, onde se acredita na verdadeira amizade.

Aos amigos Leonardo Murta, Gustavo Veronese, Márcio Barros, Alexandre Dantas, José Xavier, Marco Mangan, amigos do Projeto Odyssey, pelo constante apoio, pela amizade, pelos momentos divertidos no laboratório e a colaboração com várias idéias para a dissertação.

Aos meus familiares Antonio Nascimento, Márcia Nascimento, Paulo Sérgio Nascimento, Mauro Nascimento por todo apoio que me deram desde o primeiro dia que cheguei ao Rio de Janeiro, sempre me ajudando no que foi preciso.

Aos amigos de longa data Jaime Sabat Neto e Paulo André Silva, amigos desde a graduação, e que a vinda para o Rio de Janeiro serviu para fortalecer mais ainda a nossa amizade.

Aos amigos Carla Delgado, Catarina Rocha, Nicolas Ruberg, Gabriela Ruberg, Fernanda Baião, Leonardo Guerreiro, Rodrigo Salvador, Márcio Duran, Daniel Schneider, Sascha Kalinowski, Augusto Gomes, Gustavo Simões, Kelvin Reis, Gláucia Moreira, Helena Cintra, Juliana Nascimento muitos amigos que eu fiz durante o Mestrado, e que essa amizade ainda permanecerá por muitos anos.

Ao Cláudio José, José Roberto Dutra, Celso Tirré, Eduardo Muruci, Júlio Pires, Sérgio Ladany, Cláudio Galvão, companheiros de trabalho que sempre entenderam a importância do Mestrado, e que me deram suporte para que eu pudesse terminá-lo.

À Patrícia Leal, pela amizade, pelas correções na dissertação e desde o dia em que cheguei para fazer a entrevista até hoje sempre foi muito prestativa em tudo que precisei.

À Ana Paula Prata, por estar sempre disposta a ajudar em tudo que fosse preciso.

A CAPES, pelo apoio financeiro.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

COMPAGENT: UMA FERRAMENTA PARA O APOIO A BUSCA E RECUPERAÇÃO DE INFORMAÇOES ORIENTADAS A DOMÍNIO NA WEB

Marcelo Nascimento Costa

Dezembro/2002

Orientadora: Marta Lima de Queirós Mattoso
Cláudia Maria Lima Werner

Programa: Engenharia de Sistemas e Computação

A Engenharia do Domínio (DE) objetiva fornecer um grupo de componentes reutilizáveis dentro de um domínio específico. A Engenharia da Aplicação (EA) tem como objetivo construir aplicações pela reutilização de componentes resultantes de um processo de Engenharia de Domínio. Este trabalho apresenta uma ferramenta que permite a busca e recuperação inteligente de informações na Web, provendo aos usuários, envolvidos na modelagem de um domínio, uma vasta e relevante quantidade informações para apóia-los na realização dos processos de ED/EA.

A ferramenta utiliza diversas técnicas, como mediadores, filtragem colaborativa, recomendação colaborativa, modelagem de usuário, estereótipo e *relevance feedback*. Essas técnicas utilizadas em conjunto, na dissertação, resolvem os problemas mais comuns da recuperação de informação na Web. Estes problemas consistem em sobrecarga de informações provenientes da Web, falta de um contexto para a consulta do usuário e apresentação de informações consideradas não válidas ao usuário.

As propostas desta dissertação foram realizadas no contexto do Projeto Odyssey, desenvolvido na COPPE/UFRJ.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

COMPAGENT: A TOOL FOR DOMAIN ORIENTED INFORMATION SEARCH AND RETRIEVAL AT THE WEB SUPPORT

Marcelo Nascimento Costa

December/2002

Advisors: Marta Lima de Queirós Mattoso
Cláudia Maria Lima Werner

Department: Computer and Systems Engineering

Domain Engineering (DE) aims at providing a group of reusable components within a specific domain. Application Engineering (AE) aims at constructing applications by reusing resulting DE components. This thesis presents a tool, which enables intelligent information search and retrieval at the Web, providing domain engineers with a broad and relevant amount of information to support them in the accomplishment of DE/AE processes.

The tool uses several techniques, such as mediators, collaborative filtering, collaborative recommendation, user modeling, stereotypes and relevance feedback. These techniques when combined solve the most common problems found in information retrieval at the Web. These problems consist in information overload, absence of context for user's query and the presentation of not adapted information to the user´s needs.

The proposal was accomplished in the context of the Odyssey Project, developed at COPPE/UFRJ.

Índice

Índice de Figuras

Índice de Tabelas

Capítulo 1 - Introdução

1.1 Motivação

A informação possui uma importância estratégica no mundo atual. Entretanto, a quantidade de informações disponíveis atualmente é imensurável, pois esta foi acumulada durante milhares de anos. Desde o início da humanidade, teve-se a necessidade de se armazenar a informação e, em contrapartida, é necessário prover um meio para que o homem conseguisse recuperá-la. A recuperação da informação tem a finalidade de atender a necessidade que o ser humano possui de obter conhecimento para os mais diversos fins (Baeza-Yates e Ribeiro-Neto, 1999) (Salton, 1989) (Belkin, 1997). Esta é uma questão importante a ser tratada atualmente, pois está relacionada, exatamente, em atender uma necessidade que o homem tem de recuperar a informação para utilizá-la em diversos fins.

Um exemplo da importância da informação pode ser representado no caso em que se procura a solução para um determinado problema. O homem precisa recuperar o conhecimento já armazenado a respeito do assunto tratado, procurar se já existe solução para o problema e, caso exista, aplicá-la ao problema em questão (Belkin, 1997). Caso não exista, pode-se, a partir do conhecimento obtido, resolver o problema.

A recuperação da informação (RI) fornece subsídios para tentar atender a necessidade de informação (Belkin, 1997), o que torna a RI uma técnica cada vez mais importante em qualquer área do conhecimento. O processamento da informação recuperada pode gerar conhecimento a respeito de determinadas áreas e em algumas, atualmente, se torna uma vantagem competitiva, como no caso da gestão de uma empresa que possui diversos concorrentes e precisa atender, de acordo com os seus interesses, clientes com perfis diferentes.

Um sistema como o CRM (*Customer Relationship Management*) (Dilché, 2001), que trata do relacionamento com os clientes, tem como embasamento principal trabalhar com a questão do armazenamento de informações a respeito dos seus clientes como perfil de gastos, produtos utilizados, entre outros. Depois, esse conhecimento, armazenado pela ferramenta, pode ser recuperado e a organização, com a posse deste conhecimento

previamente armazenado, tem condições de se adaptar às necessidades específicas de cada cliente. Portanto, tende a prover uma melhor qualidade no atendimento aos seus clientes.

Outra área altamente influenciada pela RI são os Sistemas de Suporte a Decisão (SSD). Esses sistemas surgiram a partir da idéia do armazenamento de grandes volumes de dados para que, posteriormente, possa ser extraído conhecimento a partir da consolidação desses dados. O estudo de DataWarehouse trata dessa questão de armazenamento de grandes volumes de dados e o Data Mining explora as técnicas e algoritmos necessários para a extração de conhecimento a partir de um DataWarehouse criado. A extração desse conhecimento, que pode ser no formato de gráficos, planilhas, entre outros, apóia a tomada de decisões estratégicas pela camada gerencial das organizações. Como foi abordada brevemente acima, a informação tem uma função estratégica no mundo atual. Um bom aproveitamento das informações pode ser o diferencial para um profissional de qualquer área ou para uma organização.

A distribuição da informação é um fator importante a ser considerado. Inicialmente, as informações eram restritas a uma máquina isolada, depois foram distribuídas em uma rede local. Quanto maior a distribuição, maior a dificuldade na sua recuperação e, provavelmente, o volume dessas informações será maior. Cronologicamente, mais adiante, na distribuição de informações, teve o surgimento da World Wide Web (WWW), que, devido a sua crescente expansão, não se tem conhecimento exato do seu tamanho. Existem estimativas a respeito da ordem de grandeza da quantidade de documentos existentes, no entanto, não se tem o mínimo controle da estrutura destes.

A recuperação da informação é uma técnica útil para apoiar diversas áreas do conhecimento, inclusive para a área de Reutilização de Software e suas subáreas, como Engenharia de Domínio (ED) e de Aplicação (EA) (Braga, 2000) (Miller, 2000). Um processo de ED é dividido em diversas etapas (Braga, 2000). Para a realização dessas etapas de forma eficiente, é importante, dentre outros aspectos, a possibilidade de acesso a diversos repositórios de informações, com o intuito de ampliar o conhecimento do pessoal envolvido na análise de determinado domínio. Esses repositórios podem armazenar informações provenientes de diversos processos de ED realizados para o mesmo domínio, ou para algum domínio relacionado. Caso, nestes repositórios locais ou remotos, não exista

a quantidade de informações suficiente, é necessária a busca em outras fontes alternativas de informação.

Atualmente, a fonte de informação mais explorada tem sido a Web. Entretanto, devido à diversidade e ao volume de informações, há uma certa dificuldade em encontrar informações que sejam relevantes para o usuário (Baeza-Yates e Ribeiro-Neto, 1999). Deve-se considerar, também, que a Web possui uma grande quantidade de informações não validadas, pois qualquer um pode publicar suas informações em um site, sem ter aprovação prévia, validação da consistência ou, até mesmo, veracidade da informação disponibilizada (Costa, 2000).

A área de desenvolvimento de software baseado em componentes (DBC) é uma área promissora (Sametinger, 1997) (Heinenman e Council, 2001), que afetará a construção de aplicações em diversos domínios da computação, entre eles, encontra-se o comércio eletrônico. Em (Bichler et al, 1998), é feito um estudo a respeito da aplicabilidade do DBC no comércio eletrônico e também como este seria aplicado na construção de aplicações de outros domínios em geral. Uma fonte natural de procura de componentes a ser utilizada no contexto do DBC será a própria Web. A procura pode ser feita em sites que disponibilizam componentes *sharewares*[1] ou *freewares*[2]. Podem também ser procurados componentes em sites de desenvolvedores que desejam compartilhar seus componentes construídos para outros utilizarem, como em um esquema de software livre ou de colaboração de usuários de um grupo de desenvolvimento.

Como a complexidade do problema cresceu, apenas as técnicas tradicionais da recuperação da informação não conseguiam tratar o grande volume de dados, sendo necessária a união dessa área com outras áreas do conhecimento, formando um conjunto de técnicas interdisciplinares com o intuito de apoiar o usuário da forma mais eficiente possível. Foram consideradas diversas técnicas na dissertação como colaborações entre usuários, mediadores e além dessas foram consideradas ainda mineração de resultados a serviços de busca, modelagem de usuário e análise de julgamentos do usuário (Costa, 2000). Essas técnicas podem apoiar a busca por informações orientada a um domínio específico do conhecimento, como ED e EA (incluindo DBC), de uma forma mais efetiva.

---

[1] É o software que é obtido com permissão para pessoas redistribuírem as cópias, mas qualquer um que usa continuamente uma cópia tem que pagar uma licença de uso (FSF, 2002).

[2] Utilizado para pacotes de software que permitam redistribuição, mas nenhuma modificação (FSF, 2002).

As técnicas utilizadas nesta dissertação se tornaram temas importantes de pesquisa, considerando que existem diversos congressos que abordam exclusivamente cada técnica, como: o *User Modelling* (UM, 2003), que aborda a modelagem de usuário; o *Web Information Sofware Engineering* (WISE, 2002), trata da área de Engenharia de software para construção de sistemas na Web; o *Mining Enhanced Web Search* (MEWS, 2002), que trata da mineração de resultados de busca na Web. Autores como Benaki (1997), Lawrence (2000), Maes (2000) se preocupam com essas diversas áreas, que possuem muitos problemas ainda em aberto, até pelo pouco tempo do surgimento da Web e da utilização desta como fonte de informação a ser considerada na busca por informações.

Os usuários que participam de um processo de modelagem de um domínio ou utilizam Engenharia de Aplicação possuem a necessidade de informação, pois precisam buscar informações para realizarem suas atividades neste processo. Principalmente, porque a Engenharia de Aplicação/Engenharia de Domínio se baseiam, essencialmente, na busca por soluções já implementadas para7 determinados problemas.

1.2 Objetivos

A Web é um grande desafio atualmente para diversas áreas de pesquisa, pois é um vasto campo de repositório de informações. Dentro essas áreas existem estudos relacionados à semântica das informações na Web (W3C, 2002), apresentação dos metadados das páginas, modelagem de sistemas de informações para a Web, mineração de dados na Web (Sristava al., 2000) e recuperação de informação na Web (Baeza-Yates e Ribeiro-Neto, 1999).

Essa dissertação versa, especificamente, sobre a pesquisa relacionada à recuperação de informação na Web, procurando fornecer contribuições importantes para problemas específicos nesta área, como a busca de informações na Web para encontrar componentes úteis para reutilização num processo de DBC e, também, informações para apoiar a modelagem de domínio em um processo de Engenharia de Domínio.

Para efetivar a busca com o intuito de suprir a necessidade de informação no mundo Web, são necessárias ferramentas que superem, ou pelo menos atenuem, os diversos problemas existentes no mundo das informações publicadas na Web. Existem inúmeros

problemas, sendo que os principais são (Baeza-Yates e Ribeiro-Neto, 1999) (Kobayashi e Takeda, 2000) (Lawrence, 2000):

- **Overload de informações**: é o principal deles, consistindo na "sobrecarga" de informações, ou seja, quando o usuário precisa de alguma informação específica e a procura na Web retorna um excesso de informações sendo necessário filtrá-las manualmente. Este é um trabalho exaustivo, pois podem ser retornados milhares de páginas e, às vezes, não se chega a um resultado satisfatório.
- **Informações fora de Contexto**: As informações não possuem um contexto específico relacionado a cada usuário, pois, normalmente, não são armazenadas preferências a respeito do usuário. Como não se encontram em um contexto, podem ser genéricas demais, não atendendo a necessidade do usuário. Portanto, a personalização é um fator importante para o usuário.
- **Informações não-validadas**: A publicação de informação na Web é muito simples de se fazer. Atualmente, existem muitos sites onde se pode obter espaço e publicar informações gratuitamente, não existindo nenhum padrão de qualidade ou, por exemplo, uma agência reguladora que controle estas informações. Portanto, a validade de todas as informações publicadas não pode ser comprovada.
- **Falta de Colaboração entre usuários**: Não é comum serem encontrados serviços de busca ou, até mesmo, ferramentas que considerem o agrupamento de usuários com interesses semelhantes. Isto significa que uma informação que pode ser interessante para um usuário não tem como ser repassada para outro usuário com os mesmos interesses. Essa situação poderia ajudar um usuário no início das suas pesquisas a reconhecer informações que lhe sejam importantes.

Apenas a recuperação de informação com seus algoritmos pura e simplesmente não conseguiria resolver esses problemas. Seria necessário que fossem aliadas diversas técnicas de outras áreas, que ofereceriam uma personalização às consultas do usuário de uma forma colaborativa, ou seja, quanto mais ele ou usuários de um mesmo grupo fizessem consultas a Web, mais apuradas seriam as respostas para as consultas.

O objetivo da dissertação é prover uma ferramenta que apóie "inteligentemente" o usuário na recuperação de informações orientadas a um domínio de conhecimento e também na recuperação de componentes localizados remotamente, de forma a serem reutilizados. Além disso, procura contribuir com soluções e alternativas que possam resolver problemas existentes na recuperação de informação na Web, como *overload* de informações, informações fora de um contexto específico, informações não-validadas e a falta de colaboração entre os usuários.

O ambiente de desenvolvimento de software adotado para ilustrar a implementação da ferramenta CompAgent de apoio à recuperação de informações na Web orientada a domínio foi o Ambiente Odyssey (Werner et al., 2000), que visa prover uma infra-estrutura de suporte à reutilização baseada em modelos de domínio. O Ambiente Odyssey utiliza extensões de diagramas da UML para representar o conhecimento de um domínio, e permite que esses diagramas sejam reutilizados para facilitar a construção de aplicações. A CompAgent se insere no contexto do Odyssey como uma ferramenta interna que tem o intuito de prover um aumento do conhecimento para suas diversas categorias de usuários. O aumento do conhecimento facilita as diversas fases da modelagem do domínio e, também, a construção de aplicações em domínios já modelados anteriormente.

A área de recuperação na Web possui vários grupos de pesquisas com diversas ferramentas implementadas. Estas ferramentas possuem um caráter genérico, procurando abranger uma vasta gama de usuários com interesses diferentes. A CompAgent é uma ferramenta mais específica, que aborda o problema no contexto da recuperação de componentes e de informações para os usuários de um ambiente de desenvolvimento de software, no caso o Odyssey.

Na dissertação foram englobadas diversas técnicas de áreas, que são coordenadas para a obtenção dos seus objetivos. Algumas técnicas foram: filtragem e recomendação colaborativa de informações; mediadores em banco de dados heterogêneos; modelagem do usuário; análise do *feedback* do usuário para adaptação das bases de informação utilizadas pela ferramenta.

1.3 Organização do Trabalho

Este trabalho está organizado em 5 capítulos.

No capítulo 1 é a presente introdução, onde são apresentados a motivação, o objetivo e a organização do trabalho.

No capítulo 2 são apresentadas algumas abordagens sobre o estado atual da recuperação de informação, focando a questão da recuperação de informação na Web. Ao final são apresentadas e comparadas algumas ferramentas com objetivos similares ao da apresentada na dissertação.

O capítulo 3 aborda alguns problemas apresentados pelos serviços de busca atualmente. São descritas detalhadamente as arquiteturas de dois dos principais serviços busca Google e AltaVista, sendo feita ainda uma comparação entre esses dois serviços de busca.

No capítulo 4 é exibida a abordagem proposta, que consiste de uma ferramenta que utiliza conjuntamente técnicas interdisciplinares, de forma a se obter uma melhora na qualidade das informações obtidas nos serviços existentes de busca na Web.

No capítulo 5 é apresentada a implementação da ferramenta com os seus detalhes técnicos e características implementacionais. Ao final, é apresentada uma avaliação da utilização da ferramenta em um domínio específico do conhecimento.

O capítulo 6 apresenta as contribuições e limitações desse trabalho, assim como os trabalhos futuros.

Capítulo 2 - Recuperação de informação clássica e na Web

2.1 Introdução

Há milhares de anos a humanidade vem adquirindo conhecimento e armazenando-o para posterior recuperação. As primeiras formas de organização do conhecimento foram bibliotecas surgidas na Idade Antiga, cerca de 4000 anos atrás, e desde então, vêm sendo estudadas formas para classificação e recuperação desse conhecimento.

O computador tornou-se uma importante ferramenta no apoio à recuperação de informação, pois possui uma grande capacidade de armazenamento, classificação e recuperação de informação. Ele é capaz de classificar e recuperar a informação bem mais rápido que um ser humano, mas sua classificação não é tão perfeita como a realizada por um ser humano.

A Recuperação de Informação (RI) (Salton, 1983) é uma das áreas de pesquisa mais importantes dentro da ciência da computação, pois trata exatamente da representação, armazenamento e acesso às informações armazenadas em um meio magnético. Uma consideração importante é que a quantidade de informações existentes nas diversas áreas do conhecimento vem crescendo consideravelmente, e essas não podem ser perdidas pela humanidade.

O problema de RI pode ser caracterizado pela tradução da necessidade de informação em uma consulta que será processada pelo sistema de RI. A tradução produz um conjunto de palavras-chave (consulta), que representa a descrição da necessidade de informação. Dada essa consulta, o objetivo da RI é encontrar, numa coleção de documentos, um subconjunto de documentos que seja útil ou relevante para o usuário (Baeza-Yates e Ribeiro-Neto, 1999). Além disso, ela deve ser realizada da forma mais rápida possível.

Os dois pontos relacionados à quantidade e qualidade, abordados acima, são de difícil resolução, tornando-se, desse modo, tema de pesquisas por diversos anos. Muitos dos problemas concernentes a RI estavam em um estágio bem avançado de pesquisa, porém estavam restritos a coleções de informações estáticas e estruturadas. No entanto, o surgimento da Web e sua vertiginosa expansão mudaram os rumos da pesquisa na área.

Surgiram outras abordagens para pesquisa, principalmente devido a questões peculiares da Web, como o crescimento exponencial do seu conteúdo, que hoje representa aproximadamente um terabyte de informações (Baeza-Yates e Ribeiro-Neto, 1999), além da grande quantidade de dados multimídia (vídeo, imagem, sons, etc) armazenados. Outra característica importante da Web é o fato que ela pode ser vista como um banco de dados grande e não-estruturado, gerenciado por diversas pessoas, organizações e empresas.

Nesse capítulo será dada uma visão geral da área de recuperação de informação clássica, que trata da recuperação de informações em conjuntos bem definidos e limitados. São descritos métodos de classificação, modelos de recuperação e medidas de qualidade de recuperação relativas à informação clássica. Na seção 2.2, também será abordada a recuperação de informação, porém voltada para a Web, listando seus problemas, comparando com a recuperação de informação clássica e descrevendo suas características específicas, como as diversas formas de classificação de informações na Web. Ao final do capítulo, trataremos das diversas ferramentas, consideravelmente "inteligentes", que apóiam a recuperação de informação na Web.

2.2 Recuperação de Informação Clássica

O usuário de um sistema de RI possui uma necessidade de buscar informações para seu uso. Na literatura, essa necessidade é chamada de necessidade de informação, que consiste de itens de informação os quais o usuário está interessado e precisa recuperá-los.

Existem algumas diferenças entre RI e recuperação de dados. Como exemplo da primeira, temos uma busca de um texto em um documento, e da segunda, uma busca de registros em um banco de dados relacional. Enquanto a recuperação de dados trata da recuperação de objetos estruturados que satisfazem determinadas condições em uma expressão de consulta relacional, a RI não possui formalismo próprio, sendo comum o retorno de informações irrelevantes para o usuário. Entretanto, na recuperação de dados, se existir qualquer objeto retornado no resultado que não atende às condições da consulta, o objeto invalida toda a resposta.

Segundo Baeza-Yates e Ribeiro-Neto (1999), a principal diferença entre recuperação de dados e RI consiste no fato que a RI abrange texto normalmente não-

estruturado e com pouca semântica ou ambígua, enquanto, em um sistema de recuperação de dados, como um banco de dados relacional, os dados possuem estrutura bem definida.

Um sistema de RI pode ser definido como um sistema que, de acordo com uma consulta submetida pelo usuário, deve ser capaz, com a posse do conteúdo (ou representação do conteúdo) dos documentos, de encontrar o maior número possível de documentos relevantes e o menor número possível de irrelevantes. Esses documentos são apresentados e organizados por um grau de importância para o usuário. O sistema também deve ser capaz de obter algum julgamento do usuário em relação à relevância das respostas. O julgamento serve para melhorar a resposta para as próximas consultas deste ou de outro usuário do sistema.

A consulta é comparada com alguma forma de interpretação do documento, pois é praticamente inviável a comparação de uma consulta com o conteúdo inteiro de todos os documentos de uma coleção. Segundo Baeza-Yates e Ribeiro-Neto (1999), a interpretação do conteúdo envolve a extração de informação sintática e semântica dos documentos e a utilização dessas informações, posteriormente, para a decisão da relevância do documento para a consulta do usuário. Essas duas questões, extração e utilização das informações, representam dois dos problemas mais importantes de RI.

Segundo Baeza-Yates e Ribeiro-Neto (1999) e Krisnjberg (1979), a relevância de um documento no conjunto resposta ao usuário é um ponto central da IR, pois a sua função principal é exatamente lhe retornar somente os documentos relevantes para o usuário. Se determinado documento for importante para o usuário, o algoritmo de RI utilizado tem que ser capaz de encontrar uma característica no documento para que ele se enquadre no conjunto resposta de documentos importantes.

Como abordagem para a solução do problema de relevância, é utilizada a técnica de agrupamento (*clustering*), que consiste na identificação e separação de objetos por grupos (Baeza-Yates e Ribeiro-Neto, 1999). Essa técnica é importante para a classificação de objetos, não sendo utilizada apenas na RI, mas em áreas como Data Mining, entre outras.

Outra área importante de estudo da RI é a de indexação de documentos. Esta consiste na busca de formas de representação do conteúdo dos documentos que possam acelerar a busca de informação dentro dos mesmos documentos. Todo documento ao entrar

na coleção ou em certos períodos de tempo, sofre um processo de indexação para que se obtenha a representação do seu conteúdo.

Apenas a atribuição de termos não é suficiente para a indexação do documento, pois nem todos os termos em um documento possuem a mesma importância para o documento. Para resolver esse problema, são atribuídos pesos que distinguem a importância de cada termo. Esses termos e seus respectivos pesos ajudarão na decisão do grau de relevância do documento para a consulta do usuário. Existem duas abordagens: manual e a automática, sendo que esta se subdivide ainda em três métodos, a saber: estatísticos (Salton,1989), probabilísticos e métodos multitermos ou de frases. Mais informações podem ser obtidas em Baeza-Yates e Ribeiro-Neto (1999) e Salton (1989).

Muitos sistemas de RI clássica possuem diferentes formas de classificar um documento como relevante ou não, pois tal decisão de classificação é dependente do algoritmo de *ranking*, que tenta estabelecer uma ordenação simples dos documentos recuperados. Essa classificação da relevância de um documento é que prova a capacidade para um sistema de RI indicar se este deve ser agrupado ou não no conjunto de documentos importantes. Cada algoritmo opera de acordo com premissas básicas em relação à relevância de documento. Além da premissa do algoritmo de *ranking*, existem outras premissas que diferem os modelos de recuperação existentes no núcleo dos sistemas de recuperação, a saber:

1. Representação do documento e consultas;
2. Diferentes estratégias para avaliar a relevância do documento em resposta a uma consulta do usuário;
3. Métodos para classificar a formatação da consulta;
4. Métodos para classificar o *feedback* de relevância do usuário.

Existem modelos de RI que são considerados modelos clássicos, entre eles: baseados na teoria dos conjuntos, algébricos, probabilísticos e modelos híbridos. Em Baeza-Yates e Ribeiro-Neto (1999) e Salton (1989), são explicados esses modelos mais importantes, onde são também explicados alguns outros modelos mais atuais, alguns derivados dos clássicos.

2.3 Recuperação de Informação na Web

A recuperação de informação clássica é uma área de pesquisa bastante avançada, com a maioria de seus problemas já bem resolvidos e atendendo às necessidades de seus usuários. O escopo de pesquisa abrange diversas áreas, como arquitetura de sistemas de recuperação, interfaces de usuário, visualização de dados e diversas técnicas como classificação e categorização de documentos, filtragem de informação, entre outras.

O panorama das pesquisas mudou completamente após o surgimento da Web, no início dos anos 90. A Web tornou-se um grande repositório de informações do conhecimento humano, facilmente acessível para consultas, que permite, ainda, a um custo muito baixo, a publicação de páginas pessoais ou comerciais. Segundo a página do instituto de pesquisas NUA Internet Survey (NIS, 2002), em maio de 2002, existiam 580,78 milhões de usuários na Web, representando 9.57 % da população mundial. O Internet Software Consortium (ISC, 2002) indicou que existem 109,574,229 hospedeiros de páginas na Web, segundo pesquisa realizada em janeiro de 2001.

Esse crescimento vertiginoso fez surgir diversos problemas novos para a pesquisa, pois, além do tamanho acima mencionado, existe a heterogeneidade dos usuários da Web. O principal problema tornou-se a busca de informações. Esta é uma tarefa extremamente difícil, pois existem milhões de páginas, e apenas navegando pela estrutura da Web, é praticamente impossível o suprimento da necessidade de informação do usuário.

Segundo Baeza-Yates e Ribeiro-Neto (1999), o principal obstáculo é decorrente da falta de um modelo de dados bem-estruturado, fazendo com que a definição e estrutura de informação sejam de baixa qualidade. Isto dificulta o surgimento de uma linguagem de consulta que atenda aos diversos modelos de dados existentes nas páginas Web.

Existem atualmente algumas áreas de pesquisa exclusivamente para a resolução de problemas de RI para Web, ressaltando que as pesquisas são sempre realizadas sobre documentos Web. Algumas pesquisas tratam da indexação de documentos, interfaces de consulta e respostas para o usuário, algoritmos de classificação de buscas Web, recuperação multimídia e RI para realização de comércio eletrônico.

Na próxima subseção são avaliados os principais problemas e desafios, fornecendo um dimensionamento do problema de busca de informação na Web. Nessa dissertação,

trabalharemos especificamente com RI para a Web, não nos preocupando com a recuperação de informação clássica.

2.3.1 Problemas e desafios da RI-Web

A seguir, são apresentados os problemas da RI-Web:

A. **Volume grande de dados:** O tamanho da Web, como citado anteriormente, é de mais de cem milhões de servidores e, segundo Lin et al. (1999), cresce a uma média de 20 gigabytes por mês. Esse crescimento exponencial impõe questões de difícil tratamento. A principal delas diz respeito a como deve ser realizada a indexação desse grande volume de dados.

B. **Dados não-estruturados e redundantes:** Alguns pesquisadores consideram a Web como um conjunto de hipertextos distribuídos, sendo que, normalmente, hipertextos possuem alguma estrutura que organiza e adiciona consistência aos dados e hyperlinks. Entretanto, para alguns pesquisadores, na Web não é encontrada a estrutura de hipertextos em seu conteúdo, pois as páginas HTML não são totalmente estruturadas e, ainda, cada página possui uma organização própria. Por esse motivo, alguns pesquisadores chamam as páginas Web de semi-estruturadas. Outra característica da Web é a redundância. Como citado por Lin (1999) aproximadamente 30% das páginas Web são repetidas.

C. **Alta porcentagem de dados voláteis:** Devido ao dinamismo da Internet, novos computadores e dados são adicionados ou removidos rapidamente. Segundo alguns estudos feitos por Kahle (1996), 40% da Web muda mensalmente. Outro problema recorrente do dinamismo da Internet é o problema dos links corrompidos. Estes links podem estar apontando para páginas inexistentes, que foram removidas ou cujo domínio da página não existe mais.

D. **Dados heterogêneos:** A Web possui uma grande variedade de tipos e formatos de documentos, como figuras, arquivos de áudio, textos e scripts, entre outros. Além desse problema, existem páginas contendo textos nas mais diferentes línguas e ainda línguas com diferentes alfabetos, como chinês e japonês.

E. **Consultas mal formadas**: As consultas dos usuários são comumente simples e com pouca expressividade quanto à informação que o usuário está necessitando

retornar. Esta simplificação das consultas dificulta o processamento dos sistemas de RI-Web.

F. **Heterogeneidade dos usuários:** Cada usuário varia em grau de conhecimento, necessidades e expectativas em relação à resposta. Por isso, foram criadas métricas orientadas a usuários, como *recall* e *precision* (Baeza-Yates e Ribeiro-Neto, 1999), que em poucas palavras, significa a medida da habilidade do sistema em recuperar documentos relevantes, em detrimento da não-apresentação de documentos não-relevantes do usuário.

G. **Comportamento específico dos usuários:** A maioria dos usuários somente consulta a primeira página de resposta do serviço de busca (Huang, 1999), ignorando o restante das respostas retornadas. Ele também raramente refaz a consulta para obter uma resposta melhor. Para a recuperação na Web, praticamente, as pesquisas levam em consideração apenas o *recall* e *precision* dessa primeira página.

H. **Número de usuários:** O número de usuários que acessam simultaneamente o serviço de recuperação é consideravelmente maior que em um sistema de RI clássico.

Algumas considerações podem ser feitas em relação aos problemas comentados anteriormente. Alguns nunca serão resolvidos, como a diversidade de linguagens, outros agravados, como o volume de dados e heterogeneidade dos usuários. Outros problemas estão relacionados à interação do usuário com o sistema de recuperação.

O primeiro problema dessa interação é a capacidade do usuário de propor uma consulta apropriada para a resposta que ele deseja. Essa questão é importante, pois não existe uma regra para guiar o usuário na criação da sua consulta e, como existem milhares de documentos na Web, o usuário não tem como prever o conteúdo semântico do documento desejado. O segundo está relacionado a quais documentos indexar e agrupar para a base de documentos e, desses, selecionar alguns para mostrar ao usuário.

Há várias questões levantadas em Baeza-Yates e Ribeiro-Neto (1999) e Kobayashi e Takeda (2000) para tratar a resposta para o usuário. Algumas delas são: como manipular e mostrar uma grande quantidade de documentos; como classificar esses documentos; como selecionar documentos relevantes, dentre outras.

O desafio dessa área de pesquisa é tentar, a partir de uma consulta bem, ou mesmo mal, formulada, obter uma resposta adequada, gerenciável e relevante para o usuário. Para alcançar esses objetivos na recuperação de informação para Web, devem ser revistas diversas técnicas de recuperação de informação clássica, tais como indexação, agrupamento e algoritmos de classificação, que serão utilizadas nos documentos Web.

2.3.2 Indexação na Web

Em um sistema de RI clássica, a tarefa de busca de documentos é mais simples, pois todos os documentos da coleção são armazenados localmente ou remotamente em locais conhecidos previamente pelo sistema de RI. Na Web é praticamente impossível o armazenamento de todas as páginas, sendo extremamente lento acessar remotamente as páginas nos servidores Web. Por isso torna-se mais do que necessária a indexação das páginas Web, sendo que esta é uma tarefa extremamente complexa em relação aos problemas dos sistemas de recuperação clássicos.

Atribui-se a dificuldade na indexação das páginas à enorme quantidade de páginas Web, considerando, ainda, a espantosa proporção de crescimento da Web. Outros problemas encontrados são as freqüentes atualizações das páginas. Segundo Kobayashi e Takeda (2000), estima-se que uma página de texto permanece igual aproximadamente 75 dias, enquanto que Kahle (1997) estima que 40% da Web muda todo mês.

Os dados fornecidos por Lawrence (1999), consideram apenas alguns dos serviços de buscas principais e indicam que a indexação desses serviços abrange apenas uma pequena parte das páginas da Web. Ele cita ainda que nenhum serviço de busca abrange mais do que 16% do total de páginas da Web.

Alguns pesquisadores como Henzinger et al. (1999), perceberam que é impossível a indexação de todas as páginas da Web. Por isso, foram sugeridos métodos para avaliar a qualidade da indexação das páginas da Web. Inicialmente, a única avaliação era apenas relacionada à quantidade de páginas indexadas, mas depois se identificaram duas métricas importantes, como o *indegree*, uma análise de quantas páginas apontam para a página Web considerada; e o algoritmo *PageRank* (Brin e Page, 1998), que além de considerar o número de páginas que apontam para a página em questão, considera a qualidade dessas páginas, que consiste na importância das páginas apontadas para esta.

A invenção do espalhamento de palavras (*spamming*) gerou uma dificuldade adicional na tarefa de indexação. O espalhamento de palavras é uma técnica utilizada pelos publicadores de páginas Web para enganar os serviços de busca durante a indexação das páginas. Ele consiste no uso repetitivo de palavras ou textos "escondidos" inseridos nas páginas, tentando encaixá-las com um grau de importância relevante em diversas áreas, e retornando as páginas no topo das consultas dos usuários (Kobayashi e Takeda,1999).

2.3.3 Métodos de indexação na Web

Os métodos de indexação na Web se baseiam em fundamentos semelhantes aos da RI clássica. Entretanto, devido aos problemas citados na seção 2.3.2, a indexação se torna mais complexa, e influencia na criação de novas abordagens, como softwares robôs de indexação.

2.3.3.1 Indexação Manual

A indexação manual ou "humana", mesmo sendo atualmente inviável, ainda é utilizada por alguns serviços de busca, como o Yahoo (Yahoo, 2002) e o Cadê (Cade, 2002). Os dois serviços utilizam esta técnica devido a uma característica particular, na qual o próprio usuário cadastra sua página, inserindo a URL, indicando a categoria e descrevendo o seu conteúdo. Após o cadastro, eles verificam a consistência do endereço e se esta página deve ser inserida na sua base de índices.

2.3.3.2 Indexação baseada em Agentes

A segunda classe de métodos de indexação trata da indexação automática das páginas, ressaltando-se que atualmente tornou-se e futuramente se tornará a única solução viável para indexação. Para realizar essa indexação são utilizados agentes inteligentes, devido as suas principais características, tais como autonomia, mobilidade, entre outras. Essas características aos agentes a navegação por todas as páginas da Web e a respectiva indexação destas.

Agentes inteligentes possuem a seguinte definição, de acordo com (Wooldridge e Jennings, 1998): um agente é um sistema computacional encapsulado, situado em algum

ambiente, sendo capaz de realizar ações flexíveis e autônomas no ambiente, de forma a alcançar seus objetivos.

Deve ser considerado que as páginas a serem indexadas estão localizadas em outra máquina, ou seja, considerando o problema desse modo, devemos utilizar uma característica importante dos agentes: a que trata da sua mobilidade.

Segundo Brenner et al (1998), mobilidade é a capacidade de um agente em navegar dentro de uma rede local ou remota, movendo-se de um computador para outro. Essa característica, combinada com autonomia, permite a diminuição considerável do tráfego de informações na rede, pois o agente realiza o seu deslocamento até o computador-alvo da consulta, executa a tarefa e depois retorna as informações para o seu local de origem. O agente evita a sobrecarga do envio e recebimento de mensagens dos servidores.

Existem alguns nomes diferentes associados a esse tipo de agentes de indexação de documento. No contexto dessa dissertação, ele será chamado pelo seu nome mais genérico, indexadores automáticos. No entanto, em arquiteturas de serviços de busca, é freqüentemente chamado de vasculhadores (*crawlers*). Alguns dos mais importantes serviços de busca utilizam essa tecnologia, como o AltaVista, Lycos e o Google.

O funcionamento básico dos agentes consiste em vasculhar a rede à procura de páginas que são enviadas a um servidor, onde é realizada a indexação da página e, depois, armazenada no banco de dados de índice do serviço. As técnicas, algoritmos e a forma de varredura da Web, na qual os vasculhadores trabalham, serão apresentados no próximo capítulo, que versa sobre os serviços de busca.

Para facilitar a indexação das páginas pelos indexadores automáticos, são utilizados elementos associados a cada página. Esses elementos são conhecidos como metadados, definidos como descrições de recursos de informações, ou funcionam como apoio para o acesso ao recurso de informação (Cathro, 1997). Resumindo a definição, metadados são tratados como dados sobre dados.

No contexto de páginas Web, metadados podem ser tratados como arquivos invisíveis ligados a páginas Web ou como marcações que forneçam uma semântica maior aos dados contidos nas páginas Web. Essas informações estão inseridas no conteúdo das páginas, ressaltando-se que essas marcações não alteram a apresentação do documento em um visualizador.

Inicialmente, as formas mais comuns de metadados foram dados para a descrição de informações sobre documentos genéricos, a exemplo de arquivos textos, arquivos pdf e não-específicos a documentos Web. Essas informações (metadados) teriam a função de facilitar a recuperação mais rápida e precisa de documentos. Alguns metadados mais comuns incluem autor, data da publicação, tamanho do documento e o gênero do documento (Baeza-Yates e Ribeiro-Neto, 1999). Esse tipo de metadados, conforme citado anteriormente, pode ser tratado como descritivo. São metadados externos ao significado do documento, possuindo mais informações relacionadas à descrição do documento do que ao seu conteúdo.

Um padrão de metadados descritivo que tem uma boa aceitação e recomendado pelo W3C é o *Dublin Core* (Dublin Core, 2002). Este é composto por um conjunto contendo quinze elementos de metadados: título, criador, assunto, descrição, publicador, contribuidores, data, tipo, formato, identificador do recurso, fonte, linguagem, relação, abrangência e direitos à cópia *(copyright).*

Outra categoria de metadados existente engloba os metadados semânticos, que está mais relacionado ao assunto tratado pelo documento. Em outras palavras, refere-se a semântica do documento. Para a descrição semântica dos documentos, são utilizados conjuntos de palavras-chave relacionados ao conteúdo do documento. As palavras-chave costumam ser retiradas de um vocabulário de termos restritos ao assunto do documento. Esses termos podem formar um *tesauros* ou uma ontologia[3].

Um dos problemas principais das páginas Web, escritas na linguagem HTML, está relacionado à falta de características para a representação do seu conteúdo, de modo que facilite a indexação das páginas. Para tentar melhorar esses problemas, o World Wide Web Consortium (W3C, 2002), organismo que desenvolve tecnologias para a Web, incluiu, na versão 3 do HTML, uma marcação chamada de META, que permite aos autores do documento indicar termos que serão utilizados como informação de indexação do documento. Esse marcador causa alguns problemas, por exemplo, no momento da atualização individual de documentos e para o gerenciamento da coleção de documentos. Kobayashi e Takeda (2000) abordam esse problema mais profundamente.

[3] Um vocabulário de termos de um determinado domínio e a especificação dos relacionamentos entre eles (Wiederhold e Genesereth, 1997).

Outra técnica existente é a anotação, discutida por Nagao e Hasida (1998). Existem três exemplos de anotações mais comuns: anotação lingüística, comentários e anotações relacionadas à multimídia. Anotação linguística é a usada para resumo automático de páginas e recuperação automática de conteúdo. Alguns sites, como o do Yahoo e o Cadê, em que o próprio usuário escreve o resumo de suas páginas, utilizam esse tipo de metadados. Anotação multimídia é utilizada para incluir dados multimídia não-textuais, como imagem e som adicionados à página Web. Anotação comentário refere-se à inserção de comentários em documentos Web por uma pessoa diferente do autor. Além de ajudar a indexação e recuperação, esse tipo de anotação facilita a avaliação de documentos no *site*.

Houve, portanto, uma evolução na pesquisa na área de metadados: atualmente, procura-se fornecer cada vez mais semântica aos documentos Web. Assim, evoluiu-se para a área conhecida como Web Semântica (Moura, 2002).

2.3.4 Semântica da Web

As primeiras iniciativas estavam relacionadas a uma área de pesquisa da W3C conhecida como metadados da Web. A própria W3C evoluiu com as pesquisas e passou a chamar essa área de pesquisa como Web semântica (Moura, 2002).

A Web semântica é definida por Berners-Lee et al (2001) da seguinte forma: uma extensão da Web corrente, cujo objetivo principal é adicionar aos dados uma semântica bem definida, ajudando computadores e pessoas a cooperarem de uma forma mais fácil. Esse grupo de pesquisa trabalha para definir padrões, tecnologias e políticas para tornar os dados da Web disponíveis não apenas para visualização, mas também para automação, integração e reutilização dos dados entre várias aplicações. Isso pode ser considerado até mesmo para informações que tenham sido projetadas totalmente de forma independente.

Um dos padrões definidos pela W3C foi a XML (*eXtensive Markup Language)*, que é considerada por Widom (1999) uma revolução na representação de informações na Internet. A XML surgiu como um padrão para a representação de dados e troca de informações na Web. A idéia básica é muito simples: a inserção de marcadores nos elementos de dados identifica o significado dos dados, ao invés de apenas identificar o formato em que o dado deve ser apresentado. São ainda representados os relacionamentos

entre elementos de dados através do aninhamento de marcadores e referências entre eles. Na XML são representadas apenas informações de conteúdo das páginas, separando-as da apresentação. Os dados podem ser visualizados através de uma linguagem de visualização, a XSL (*eXtensible Stylesshet Language*). A XSL "lê" a página XML, formata e a apresenta para o usuário em um *browser* comum.

A XML poupa o trabalho, sujeito a erros, da retirada de informações das páginas HTML, pois XML é mais simples, facilmente analisável, auto descritiva e já existem diversos analisadores que realizam essa tarefa.

Um dos principais estudos na área de Web semântica é um padrão de metadados da Web, o RDF (*Resource Description Framework*) (Moura, 2002). O RDF tem como objetivo definir um mecanismo para a descrição de recursos Web independente do domínio e da semântica de qualquer aplicação construída no domínio. Sua ênfase está na criação de facilidades para habilitar o processo automático de recursos Web. Ele pode ser utilizado em diversas áreas da computação, como a de descoberta de recursos, que fornece melhor capacidade aos serviços de busca; e a área de agentes de software inteligentes, para facilitar o compartilhamento e troca de conhecimento.

Os metadados do RDF possuem um modelo que tenta representar a programação e o transporte de seus metadados de uma maneira que maximize a interoperabilidade de servidores e clientes Web desenvolvidos independentemente. A sintaxe do modelo RDF baseia-se inteiramente no padrão XML, pois um dos seus objetivos é a interoperabilidade, que pode ser obtida utilizando XML. Os dois modelos são complementares: RDF é um modelo de metadados para a descrição de recursos e a XML o complementa, servindo como padrão de suporte para a interoperabilidade. No W3C ressalta-se a possibilidade de surgir formas alternativas de se representar o mesmo modelo de dados do RDF.

A área de estudo de ontologias possui um papel importante na área de Web Semântica, pois procura fornecer uma maior semântica para páginas Web e também é fundamental para a integração de bancos de dados heterogêneos, entre outras características importantes. Atualmente, existem vários trabalhos e arquiteturas implementadas para gerenciar e integrar ontologias. Entre elas, podemos citar o OIL (OIL, 2002) e o SHOE (SHOE, 2002).

Nesse trabalho, apresentamos apenas uma visão rápida sobre o assunto. Mais informações são encontradas na W3C, que trata de semântica na Web com várias referências sobre o assunto.

2.4 Agentes Inteligentes na Web

Como foram citados na seção 2.3.3.2, os agentes inteligentes realizam uma importante função na execução de tarefas na Web. Para a dissertação, os agentes mais importantes são aqueles que apóiam o usuário na resolução de alguns problemas pungentes na utilização da Web, como por exemplo: descoberta, classificação e filtragem de informações, personalização de informações e realização automática de tarefas simples.

Os agentes procuram oferecer uma solução para organizar a desorganização das informações na Web. Esses agentes trabalham de forma colaborativa com o mesmo, mas, normalmente, não necessitam de interferência do usuário. Detêm conhecimento, métodos de resolução e dados relacionados ao problema, possuindo ainda a capacidade de personalização através do aprendizado e da adaptabilidade de acordo com o usuário.

Algumas informações são relevantes para a classificação da arquitetura de funcionamento dos agentes inteligentes para a recuperação de informação (Pretschner e Gauch,1999):

- Modelo do usuário: captura as informações sobre o grau de conhecimento do usuário, interesses e preferências.
- Algoritmo de aprendizado: indica como é feita a adaptabilidade do modelo de usuário de acordo com a utilização do sistema.
- Modelo de classificação: algoritmo utilizado para a classificação das informações para o usuário.
- Tipo de colaboração: indica se existe colaboração entre usuários, ou seja, se uma informação utilizada por um usuário pode ajudar a determinar informações importantes para outro usuário.

Foram analisados alguns representantes da categoria de agentes para a recuperação de informações na Web. Entre as ferramentas descritas estão incluídas diversas categorias,

a saber: agentes de apoio a metabusca, sistemas multi-agentes e serviços de recomendação de páginas. Ao final desta seção, é mostrada uma tabela comparando as ferramentas apresentadas.

2.4.1 Syskill & Werbert

O Syskill & Webert (Pazziani, 1998) é um agente de software utilizado para recomendar ao usuário páginas WWW que sejam do seu interesse. Indicando o interesse em determinadas páginas da Web, esse agente captará o perfil de tal interesse, o qual será utilizado depois para a classificação de páginas durante a navegação do usuário. Inicialmente é inserido um perfil dos interesses do usuário, e cada vez que este classifica uma página, seu perfil é revisto para adequá-lo a essas novas classificações. É importante frisar que cada usuário possui um perfil diferente para cada tópico de seu interesse.

A ferramenta Syskill & Webert identifica o interesse em páginas Web de duas formas: na primeira forma, o usuário indica um tópico e uma página-índice, categoria de página que contém diversos links sobre determinado assunto. A partir dessa página e de todas as outras em que o usuário navegar, o usuário terá controles adicionais para indicar interesse ou desinteresse em cada link da página sendo visitada. Na segunda forma, a ferramenta envia consultas ao serviço de busca Lycos (Lycos, 2002). Ela utiliza na consulta dois conjuntos de palavras: um com palavras comuns ao tópico (definem o tópico); outro com palavras que discriminam o tópico (distinguem páginas importantes dentro do tópico). As páginas retornadas pelo LYCOS são filtradas e recomendadas pela ferramenta com um grau de confiança no link, baseando-se no perfil do usuário.

No primeiro modo, o grau de interesse pelo link é indicado pelo próprio usuário. Esse grau possui três escalas de interesse: forte, média e baixa. Caso seja necessário acessar os outros links da página, o usuário pode pedir conselho para saber quais links explorar. Nesse caso, o Syskill & Webert realiza o cálculo da importância do link, indicando ao lado do link, com a imagem de uma "face sorridente", que o link é recomendado, ou seja, é considerado importante pela ferramenta e o usuário ainda não o visitou. Com um sinal de proibido, ao lado do link, a ferramenta avisa que, de acordo com o seu perfil, o usuário deve evitar a página, pois ela não possui informações importantes para ele. O restante dos

links possui uma classificação probabilística entre 0 e 1 para indicar o grau de importância da página.

A partir dos exemplos obtidos durante a navegação, é utilizado como técnica de aprendizado de máquina o algoritmo de classificação naïve Bayesiano, descrito por Pazziani (1998) como a melhor solução para o problema de aprendizado da ferramenta. Este algoritmo executa a classificação das respostas, no mínimo, tão bem quanto outros algoritmos, necessitando de menos recursos computacionais para a sua execução.

2.4.2 Fab

O Fab (Balabanovic e Shoam, 1997b) é descrito como uma implementação distribuída de um sistema de serviço de recomendação de documentos para Web. Ele foi criado na Universidade de Stanford, a partir do sistema LIRA (Balabanovic e Shoam, 1995), combinando dois métodos para recomendação de documentos existentes na Web: um baseado em conteúdo e outro baseado em colaboração.

O método baseado em conteúdo compara o perfil do usuário, composto de palavras-chave e seus respectivos pesos, com as palavras extraídas do documento, retornando-o caso ele seja considerado relevante para o usuário. Para ser considerado relevante, tem que haver um "casamento" entre as palavras-chave do usuário com as do documento. Este verifica a página e, caso esta interesse, retorna um feedback para o sistema, que atualiza o perfil do usuário com as palavras-chave do documento que ainda não existam no perfil.

O método baseado em colaboração realiza uma recomendação calcada em uma combinação de classificações retiradas de usuários que possuam um histórico de classificações de documentos semelhante ao do usuário. Esses usuários com um histórico semelhante são conhecidos como o conjunto de "vizinhos aproximados" do usuário.

Para manter o sistema baseado em recomendação e em conteúdo, o Fab incorpora as seguintes características (Balabanovic e Shoam, 1997b): mantém um perfil do usuário baseado na análise do conteúdo das páginas visitadas pelo mesmo e compara esse perfil com o de outros para tentar encontrar seus usuários similares. A segunda característica

consiste em fornecer ao usuário recomendações de itens, no caso dele ter indicado com um alto grau de confiabilidade itens com características similares, ou quando usuários similares indicaram outras páginas com características similares a esta, com alto grau de confiança.

O usuário realiza um *feedback* explícito: para cada página mostrada, ele indica a sua classificação numa escala de 7 pontos. A classificação do usuário é utilizada para atualizar o seu perfil e para o agente de coleção atualizar o perfil do tópico. Se obtiver uma classificação alta por parte do usuário, a página é enviada para os "vizinhos próximos", porém antes de enviar a recomendação para os outros usuários, esta é filtrada pelo agente de seleção do usuário.

2.4.3 Oyster

O OySTER (Mueller, 1999) é uma arquitetura multi-agente para a recuperação inteligente de informação da Web. Como técnica fundamental, são utilizados métodos de aprendizagem de máquina sobre a modelagem do usuário para filtrar resultados provenientes de serviços de busca da Web.

A arquitetura realiza uma meta-busca, chamando vários serviços de busca e realizando uma junção a partir dos resultados fornecidos pelos serviços. A junção consiste na filtragem e classificação dos links retornados pelos serviços de busca de acordo com o modelo do usuário.

São utilizados diversos agentes na arquitetura, os quais trabalham cooperativamente e concorrentemente para a execução das tarefas. Compondo a arquitetura, existe um agente-mestre que é usado para controlar e otimizar o plano de execução das tarefas. Ele interage com um servidor *blackboard* que é o responsável pelo armazenamento de todas as tarefas e o plano de execução para as mesmas.

O agente de interface é responsável pelo envio das consultas ao sistema multi-agente. Essas consultas são decompostas em planos, sendo que um plano de resposta também é criado e adicionando aos processos de filtragem e integração dos resultados.

Para a execução da meta-busca, agentes de busca são enviados para cada serviço de busca utilizado. Cada agente executa um *wrapper* específico para cada serviço, responsabilizando-se pela extração das páginas de respostas e gravação das informações disponibilizadas pelos serviços de busca. Essas informações extraídas das páginas são

gravadas em um banco de dados temporário. Elas consistem da URL, do prefixo do domínio da página, do título, de um trecho do documento, do grau de importância dado à página pelo serviço de busca e do serviço de busca que retornou a página.

Na arquitetura, existe uma classe de agentes conhecida como agentes *daemon*, responsável pela atualização do banco de dados de URL da arquitetura com as informações do banco de dados temporário e pela extração de informações adicionais sobre as informações retornadas pelos serviços de busca.

A modelagem do usuário tem um papel muito importante na arquitetura, sendo utilizada para aumentar a precisão nos processos de meta-buscas. O aumento da precisão é baseado em algumas medidas sintáticas e semânticas retiradas do modelo do usuário.

- As medidas sintáticas são:
  - Listas de palavras-chave e de categorias de documentos de interesse: estas são baseadas no resultado do processamento da *homepage* e do *bookmark* do usuário;
  - Cálculo do peso das palavras-chave pertencentes ao modelo do usuário em relação às meta informações contidas nos documentos HTML.

- As medidas semânticas são:
  - Classificação explícita de um documento durante a navegação do usuário: o usuário pode submeter uma URL como positiva ou negativa e tem a possibilidade de transformar a lista de resultados da meta-busca em um tópico específico do arquivo de marcações contendo os seus links mais importantes (*bookmark*). Durante o processo de transformação, o usuário pode esconder ou mostrar os nós do *bookmark*, indicando com um julgamento negativo os nós que ele deseja esconder e com um julgamento positivo os nós que ele quiser deixar visíveis;
  - Histórico das navegações do usuário: qualquer resultado considerado bom pelo usuário, e que tenha sido retornado pelo serviço de busca, é atualizado no modelo do usuário;

- Modelagem Colaborativa de usuário: os exemplos bons e ruins de usuários com o perfil similar são trocados, com o intuito de refinar os modelos dos usuários e as descrições de categoria dos documentos.

É importante frisar que o processamento dos resultados dos serviços de busca é feito off-line, sendo retornado para o usuário um e-mail indicando a URL onde pode encontrar a página HTML contendo o resultado da consulta.

2.4.4 WebWatcher

O Projeto WebWatcher (Joachims et al., 1997) é baseado em um agente que apóia o usuário em navegações realizadas em sites Web, aprendendo com as navegações de cada usuário pelos *hiperlinks* que compõem o site. O funcionamento do WebWatcher consiste das seguintes etapas: o usuário, ao iniciar a navegação, indica seus interesses (palavras-chave). A partir desse momento, o agente acompanha o usuário pelas navegações, indicando os itens de maior interesse na página e os links que seriam mais interessantes a serem seguidos. Cada vez que o usuário escolher um link sugerido pelo agente, significa que o usuário indicou interesse pelo link, atualizando o repositório de metadados. Quando um novo usuário acessar a página Web, será feita uma classificação de todos os *links* na página, baseando-se no repositório de metadados, sendo que a classificação é feita com base nos usuários que apresentaram os mesmos interesses.

2.4.5 Letizia

Letizia (Lieberman, 1999) é um projeto que utiliza dois conceitos de agentes: o agente de interface, que auxilia o usuário na operação de um sistema interativo, e o agente autônomo, que realiza ações sem a intervenção do usuário, ou seja, enquanto ele executa alguma outra tarefa ou está sem fazer nada. O objetivo desses agentes é apoiar o usuário nas navegações pela World Wide Web. O Letizia controla o comportamento do usuário e tenta antecipar, em tempo real, os itens que podem ser de interesse do usuário na página.

O Letizia grava as URLs escolhidas pelo usuário e lê as páginas acessadas, formando um perfil dos interesses do usuário. O perfil é construído a partir de um algoritmo

de recuperação de informação executado nas páginas acessadas pelo usuário e se baseia na freqüência de ocorrência das palavras nessas páginas.

O Letizia então faz uma busca em largura a partir dos *links* da página corrente, verificando, e indicando ao usuário, de acordo com o seu perfil, os *links* que lhe são mais interessantes. São utilizadas algumas heurísticas para analisar o comportamento do usuário, como por exemplo: verificar se o mesmo salvou uma referência para o documento; se ele seguiu o *link* indicado; e ainda o número de *links* que um usuário seguiu em uma determinada página.

2.4.6 Inquirus 2

O Inquirus 2 (Glover et al., 1999), disponibilizado pelo NEC *Research Institute* (Inquirus, 2002), faz recomendações de páginas Web baseadas nas preferências do usuário e possui uma interface dinâmica para lhe mostrar os resultados filtrados. Ele envia consultas do usuário para diversos serviços de busca da Internet e combina os resultados baseando-se em um nível de relevância para cada documento.

Deve ser ressaltado que, para cada preferência individual do usuário, é utilizada uma estratégia de busca diferente. Essa estratégia de busca é definida a partir da especificação do usuário sobre a categoria de informação que necessita recuperar. Alguns exemplos de categoria são: artigos de pesquisa ou referência a eles; páginas WWW pessoais; páginas de organizações; eventos correntes ou novos e introdução sobre determinado assunto. Cada usuário pode ter seu próprio conjunto de categorias de informações e suas respectivas estratégias, ou pode usar estratégias genéricas das categorias, comuns a todos.

A estratégia de busca pode variar de três modos diferentes, explicados a seguir:

- Seleção da fonte de consulta: em (Glover et al., 1999), é afirmado que a escolha de várias fontes de informações melhora a precisão dos resultados em detrimento a apenas a obtenção de um número maior de resultados. Atualmente, o Inquirus 2 utiliza um conjunto fixo de fontes de consulta (serviços de busca) para cada categoria de informação possível de escolha pelo usuário.

- Modificação da consulta: Consiste na alteração da consulta enviada pelo usuário, normalmente, com a adição de palavras-chave extras ou restrições não relacionadas ao tópico. Para cada categoria de informação e serviço de busca, são adicionados conjuntos de palavras diferentes. Aumentando a precisão dos resultados podem ser aceleradas as consultas, pois poucas páginas deverão ser recuperadas e filtradas.
- Classificação dos resultados personalizada: o Inquirus não utiliza a classificação fornecida pelos serviços de buscas, porém trata do problema recuperando a página e aplicando sua própria função de classificação. Essa função de classificação utiliza a teoria da utilidade (Brown et al.,1999). Trata-se de um método que se baseia em todas as preferências e fatores para a tomada de decisão, avaliando a importância de cada fator como modelo para avaliação dos resultados. Deve ser considerado que a política de ordenação baseia-se no valor assinalado por essa teoria. Portanto, cada categoria de informação selecionada pelo usuário possui uma função-utilidade relacionada, permitindo a cada usuário ter os resultados mais importantes apresentados primeiro.

O Inquirus 2 possui ainda algumas características importantes: na sua arquitetura, possui um recuperador e um analisador de páginas para garantir-lhes uma classificação consistente, ou seja, ele recupera a página sempre com a sua última versão, eliminando links "mortos" ou páginas inexistentes. Além disso, utiliza a página HTML completa para melhorar a busca e encontrar o contexto da palavra-chave no documento retornado.

Em (Glover et al., 1999), é afirmado que essas diferentes estratégias utilizadas para cada busca realizada fornecem uma arquitetura que retorna resultados importantes para as mais diferentes necessidades do usuário. São estudadas ainda formas para se recuperar o feedback do usuário, melhorar as funções-utilidade, melhorar a estratégia para as fontes a serem escolhidas e modificações nas consultas.

2.5 Comparação das Ferramentas

Para a comparação entre as ferramentas, foi montada uma tabela (Tabela 2.1) ressaltando as principais características que um sistema de recuperação na Web deve possuir. Na tabela, é descrito, resumidamente, para cada ferramenta, qual é a sua funcionalidade.

Tabela 2.1 – Comparação entre as ferramentas

Ferramenta	Usuários-Alvo	Tipo de Recomendação	Tipo de Busca	Tipo de Arquitetura	Forma de Colaboração
Syskill & Webert	Usuários navegando por páginas Web	Links WWW	Predição de links	Agentes Individual	Individual
Fab	Usuários de um serviço de recomendação de documentos	Links WWW	Predição de Links	Agentes Individual	Colaborativo
Oyster	Usuários buscando documentos WWW	Documentos WWW	Metabusca	Sistema Multi-Agente	Colaborativo
WebWatcher	Usuários procurando links importantes	Links de navegação entre páginas WWW	Predição de links	Agentes Individual	Colaborativo
Letizia	Usuários procurando links importantes	Links de navegação entre páginas WWW	Predição de links	Sistema Multi-Agente	Individual
Inquirus 2	Usuários buscando documentos WWW	Documentos WWW	Metabusca	Agentes Individual	Colaborativo

A colaboração é um diferencial importante encontrada em uma ferramenta, pois um usuário, principalmente iniciante, pode reutilizar a experiência de outros usuários que já navegaram pela ferramenta. Algumas ferramentas implementam essa característica, como Fab, Oyster e WebWatcher. Numa colaboração individual, a ferramenta apenas se adapta

ao usuário, isoladamente, sem levar em consideração as preferências de outros usuários similares. Algumas vezes essas experiências anteriores são interessantes para outros usuários.

O modelo utilizado para o algoritmo de aprendizado também pode ser um diferencial, pois é muito importante a forma como a ferramenta se adequa ao comportamento de um ou de um conjunto de usuários. A ferramenta deve evitar a obtenção explícita de *feedback* em relação às respostas retornadas ao usuário, tentando ser o mais transparente possível para este. O Letizia e WebWatcher procuram atender essa funcionalidade de transparência no *feedback* do usuário. Pela documentação obtida, as outras ferramentas implementam uma forma explícita para obter a aprovação do usuário em relação a resposta retornada por ele.

As ferramentas capazes de realizar metabusca podem ser mais eficientes que outras ferramentas que realizam apenas uma busca simples, ou somente realizam a predição de links. A metabusca fornece à ferramenta um espaço maior de pesquisa para encontrar um documento importante do usuário, enquanto, que uma ferramenta consultando apenas uma fonte, terá, provavelmente, uma quantidade bem mais limitada de informações. Como já foi abordado por Kerschberg (2002), existe a questão do custo/benefício para a realização de uma metabusca. Uma metabusca pode levar um tempo considerável de processamento, no entanto, existe o benefício de ser obtida uma resposta mais adequada às necessidades do usuário.

As ferramentas, que possuem como tipo de recomendação documentos WWW, poderiam ser mais bem aplicadas ao problema abordado na dissertação, pois seriam capazes de filtrar os documentos e indicar a importância destes para o usuário. No entanto, as ferramentas apresentadas não teriam a capacidade de encontrar informações específicas de um determinado domínio para atender o usuário, e também não possuiriam características de compartilhamento de informações entre os usuários de um mesmo domínio.

Esses problemas ocorrem devido às ferramentas não possuírem o conceito de armazenamento de informações a respeito de usuários por domínio, possuindo apenas uma visão genérica por usuário. Caso essas ferramentas fossem adaptadas e, levando em consideração que elas apresentam algoritmos eficientes, poderiam se encaixar numa visão orientada a recuperação de informações por domínio.

As ferramentas relacionadas na tabela 2.1 são abrangentes, atingindo diversas categorias de usuários. No entanto, retornam diversos tipos de informações, dentre estas, podem ser encontrados componentes de qualquer tipo, como modelo de dados, documentos de requisitos e componentes de software reutilizáveis. Diferentemente do problema abordado nesta dissertação, que consiste em implementar características específicas que sirvam para apoiar os processos de ED/EA. A metabusca realizada pela ferramenta serve para a obtenção de componentes específicos a esses dois processos. Ela pode ser feita através da conexão a um serviço de busca para a procura de informações genéricas sobre o domínio, e a um sistema criado especificamente para a publicação e recuperação de artefatos de software. Das ferramentas procuradas na literatura, não foi encontrada nenhuma que realizasse uma procura tão específica para recuperar informações orientadas ao processo de ED/EA.

Em geral, a obtenção de uma métrica para comparar as ferramentas é bastante complicada por diversos motivos, a saber: cada ferramenta tem uma forma diferente de armazenar os interesses dos seus usuários; mecanismos de aprendizado são variados, adaptando de forma distinta o perfil do usuário, sendo que existem casos em que não é feita a adaptação desse perfil; a colaboração não segue um padrão, portanto, cada ferramenta implementa sua colaboração, quando esta existe, de maneira particular; as ferramentas possuem público alvo e, até mesmo objetivos distintos. Enquanto algumas indicam páginas e links importantes para os usuários durante a navegação, outras realizam metabusca em serviços de busca para indicar documentos importantes. Numa avaliação das ferramentas, o que deve ser levado em consideração é realmente o público-alvo a qual se destina a ferramenta, e como ela utiliza suas técnicas para atingi-lo.

Na prática, as ferramentas poderiam ser comparadas através das métricas tradicionais de *recall* e *precision* (Salton, 1989). Deve ser observado que numa primeira consulta, uma ferramenta poderia apresentar melhor retorno que a outra, mas com a constante utilização das ferramentas e as diferentes estratégias de abordagem, outra ferramenta poderia se adaptar melhor ao usuário e, em uma nova consulta, apresentar um resultado melhor.

2.6 Conclusões

Existem ainda muitas questões abertas de pesquisa para a resolução de problemas na Web. A mais importante trata da recuperação de informações relevantes ao usuário, pois nenhum dos atuais serviços de busca incorporou técnicas de modelagem do usuário ou técnicas de *relevance feedback*. Aplicar essas técnicas a uma lista de documentos pode melhorar a qualidade da resposta para o usuário. Agentes inteligentes de apoio à recuperação e filtragem de informação indicam ser uma área promissora de pesquisa para melhorar a eficiência da recuperação de informações.

Uma outra área também promissora é a Web semântica, que procura fornecer mais semântica e um modelo de dados para as páginas Web. A união da semântica com o modelo de dados pode facilitar o funcionamento das linguagens de consulta para Web, como WebOQL (Arocena e Mendelzon, 1998), e de acordo com pesquisas atuais, incluir consultas baseadas em conceitos e processamento de linguagem natural.

Uma área em franca expansão na RI-Web é voltada para o comércio eletrônico, principalmente para previsão e simulação de cotação de preços para vendas pela Internet. Os principais representantes dessas áreas são os "robôs", que ajudam os consumidores a comprar, chamados de *shopbots* (Kobayashi e Takeda, 2000). Além da comparação de preços, vários *shopbots* atuais são capazes de responder por um conjunto de outras tarefas, tais como comparação de características de produtos, revisões de produtos por usuários, opções de entrega e garantia de informação (Clark, 2000).

Capítulo 3: Ferramentas de Busca na Web

3.1 Introdução

A Internet tem crescido rapidamente desde a sua criação em dezembro de 1969 (Lawrence e Giles, 1998). A quantidade de informações armazenadas, atualmente, é muito grande, tendo em vista que no início eram apenas informações de cunho científico, pois eram restritas ao meio acadêmico. Hoje em dia, ela é completamente aberta à comunidade em geral, servindo como um meio para a comercialização, divulgação de produtos e serviços e para a publicação de páginas com informações pessoais.

A explosão da Internet aconteceu mesmo com o surgimento da WWW. Esta pode ser vista como o principal repositório de dados da Internet, sendo que muitos desses dados são não-textuais, como vídeos e sons. Essa quantidade de dados cresce exponencialmente, estando os dados distribuídos por milhões de *sites* e em formatos cada vez mais heterogêneos.

Desde o início da Internet, existe uma preocupação com a recuperação dessa grande quantidade de informação disponível. A primeira ferramenta foi o WAIS (*Wide Área Information Service*), que permitia ao usuário a busca de palavras-chave sobre um conjunto de 500 índices. Esses abordavam uma grande variedade de tópicos, como ciência da computação e partículas físicas, entre outros.

A Web é o principal serviço utilizado na Internet. Ela surgiu, de fato, com a criação de uma interface gráfica, o *Mosaic*, que permitia a navegação em uma estrutura de hipertexto. Ela favoreceu a substituição do WAIS como ferramenta de busca para a Internet, pois este era incapaz de atender às novas necessidades de busca, principalmente pelo fato de possuir uma interface textual.

Desde o início da Web, foram criados diversos serviços de busca ao seu conteúdo. Os primeiros projetos foram o Lycos, desenvolvido na *Carnegie-Mellon University*, e o AltaVista, desenvolvido nos laboratórios de pesquisa de *Digital Equipment Corporation* (DEC) (Kobayashi e Takeda, 1999). Os projetos evoluíram, criando ferramentas públicas de busca e que estão disponíveis atualmente.

Neste capítulo, é dada uma visão geral dos problemas, soluções e tendências na tecnologia dos serviços de busca, bem como tratada a arquitetura dos dois principais serviços de busca na atualidade, a saber: AltaVista e Google, sendo realizada uma comparação entre os dois. Por fim, abordar-se-a as demais ferramentas utilizadas na Web para busca de informações.

3.2 Serviços de Busca

Atualmente, há diversos serviços de busca distribuídos pela Web que procuram atender à necessidade de informação da maioria dos usuários da Web. Segundo o GVU (GVU, 1998), *Graphics, Visualization, and Usability Center of Georgia Institute*, em uma pesquisa feita com usuários em outubro de 1998, 85% dos usuários Web encontram informações úteis através de serviços de busca e 88% as encontram através da navegação por *hiperlinks* em outras páginas. Chegou-se à conclusão que os serviços de busca são, praticamente, tão utilizados quanto a navegação pelas páginas da Web. Essa pesquisa demonstrou, portanto, a importância dos serviços de busca.

Como os números de páginas na Web crescem rapidamente, é inevitável a utilização de serviços de busca. O *World Wide Web Worm*, pioneiro nesses serviços, acusou uma média de mil e quinhentas consultas por dia em abril de 1994, enquanto que o AltaVista acusou ter manipulado duzentos milhões de consultas por mês em setembro de 2001 (SearchEngineWatch, 2001).

Apesar de possuírem milhões de usuários utilizando os seus recursos, os serviços de busca ainda necessitam de muita pesquisa. Diversos trabalhos como (Huberman e Lukose, 1997), (Lawrence e Giles, 1998), (GVU, 1998), (Huberman et al, 1998), (Brin e Page, 1998) e (Sam, 2001) citam problemas encontrados por usuários na sua utilização. Dentre esses problemas estão:

A - Qualidade da busca: Não é sempre que os resultados mostrados pelos serviços de busca satisfazem as necessidades de informação do usuário. Em uma lista de resultados retornados, sempre são encontradas informações desnecessárias, ou seja, *links* que não interessam ao usuário. Em alguns casos, não são obtidas informações relevantes.

B - Velocidade de recuperação: existe uma insatisfação por parte dos usuários com a velocidade de retorno das respostas às suas consultas e com o acesso lento aos *sites*

dos serviços de busca. Algumas vezes os usuários desistem de realizar a busca devido a lentidão do serviço.

C - Links errados: os *sites* Web sofrem diversas mudanças, como por exemplo: o surgimento e desaparecimento de domínios, mudança de endereços de páginas ou desaparecimento de página. Devido a esses problemas, tornou-se comum os serviços de busca retornarem *links* para páginas inexistentes. Para evitá-los, é necessário uma constante atualização dos índices dos serviços busca, tratando a eliminação desses *links* errados.

D - Problemas de interação: Existem alguns problemas de interação entre o usuário e a interface do serviço de busca. Os usuários não entendem como utilizar a lógica booleana, portanto, não podem realizar buscas avançadas. Além disso, não sabem como fornecer uma seqüência de palavras para a busca e nem como iniciá-la. Algumas vezes, eles não estão cientes dos requisitos para entrada de uma consulta, cometendo erros tais como considerar que a entrada de uma consulta no serviço de busca não é sensível ao contexto.

Uma questão importante dos serviços de busca está relacionada a sua abrangência, ou seja, ao número de páginas indexadas armazenadas no índice. Em 1997, os responsáveis pelos serviços de busca afirmaram que possuíam a Web praticamente indexada. Essa afirmação não é confirmada pela pesquisa realizada por Lawrence e Giles (1998), a qual indicou que o máximo que um serviço de busca conseguia indexar eram 24 percentuais.

Atualmente, a abrangência das páginas indexadas cresceu extraordinariamente. Segundo dados fornecidos pelo SearchEngine Watch (SearchEngineWatch, 2001), *site* que fornece relatórios estatísticos e diversas informações sobre os serviços de busca, o serviço de busca Google já ultrapassou a barreira de um bilhão de páginas indexadas, como mostra a tabela 3.1. Esta tabela foi publicada em agosto de 2001.

Tabela 3.1 – Dados sobre o índice dos principais serviços de busca

Serviço de Busca	Páginas Indexadas (em mihões)
Google	1,387
FAST	625
AltaVista	550
Inktomi	500
NorthernLight	380
Excite	350

Apesar da quantidade de páginas indexadas, a Web não foi totalmente indexada, pois existem alguns tipos de páginas que não podem sofrer indexação. Alguns motivos são: as páginas podem estar escondidas por trás de formulários de busca; podem estar incluídas em alguma política de exclusão de acesso a robôs, impedindo o acesso a determinadas páginas, ou podem necessitar de algum tipo de autenticação (Lawrence e Giles, 1998).

Outra questão importante é como as arquiteturas de serviços de busca podem resolver o problema da expansão do número de documentos da Web e fazer a atualização das suas bases-índices e de dados de uma forma eficiente.

3.2.1 Arquiteturas dos Serviços de Busca

A maioria dos serviços de busca possui a arquitetura de índices criados por agentes vasculhadores, mostrada na figura 3.1. Foi percebida uma grande dificuldade na descrição dos serviços de busca em virtude destes serem ferramentas comerciais e por isso não terem seus algoritmos de classificação e nem de indexação das páginas disponibilizados. Um outro motivo percebido e que os serviços procuram esconder é a fomra como são indexadas as páginas para usuários que desejam fazer *spamming*, conforme descrito na seção 2.3.2, das suas páginas.

A falta de entendimento do funcionamento leva alguns usuários a uma certa frustração, pois utilizam uma ferramenta de busca de informações sem saber como ela funciona internamente. O AltaVista, por exemplo, fornece pouco material a respeito de seu funcionamento, porém o Google, uma ferramenta desenvolvida em um projeto acadêmico, fornece publicamente a sua arquitetura e os seus algoritmos de funcionamento.

3.2.1.1 Arquitetura Indexadora -Vasculhamento

Na figura 3.1, é mostrada a arquitetura *crawler*-indexadora (Baeza-Yates e Ribeiro-Neto, 1999). Nessa arquitetura são utilizados os *crawlers* ("vasculhadores"), citados na seção 2.3.2.2. Existem algumas técnicas que são utilizadas pelos *crawlers* com a finalidade de buscar páginas na Web. Estas páginas são indexadas pelo módulo Indexador pertencente a arquitetura do serviço de busca. A técnica mais comum de "vasculhamento" consiste em

iniciar a busca por um conjunto de URLs e, a partir dessas, extrair outras URLs recursivamente com a técnica de busca em largura ou em profundidade.

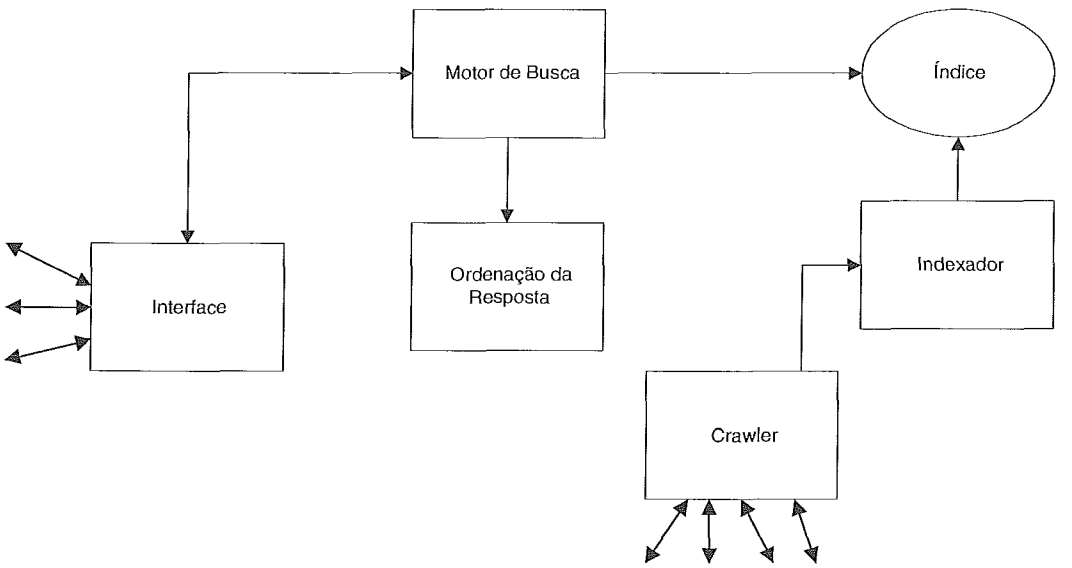


Figura 3.1- Arquitetura Indexadora-Vasculhamento

Na busca em largura, o agente percorre todas as páginas ligadas pela página corrente e realiza o processo de busca continuamente. O alcance dessa busca antige mais páginas em largura, ou seja, as páginas mais externas dos *sites*, e não consegue atingir as mais internas. Na busca em profundidade, é seguida a ligação da primeira página para as suas páginas internas, até não existirem páginas a serem seguidas. Ela oferece o acesso a páginas mais internas, todavia não é tão abrangente.

Essas técnicas funcionam com um agente de software, mas quando existem vários agentes, o controle é mais difícil, pois ocorre a probabilidade destes visitarem a mesma página. Uma estratégia, normalmente utilizada para resolver esse problema é dividir a Web por código de país ou por domínios da Internet. Cada agente percorre essas páginas, divididas por áreas, exaustivamente (Baeza-Yates e Ribeiro-Neto, 1999).

Um outro componente importante é o índice do serviço de busca. A maioria dos índices utiliza a técnica de lista invertida de arquivos. Uma lista invertida de arquivos pode ser definida como uma lista de palavras organizadas (vocabulário), em que cada palavra tem um conjunto de ponteiros para as páginas onde elas ocorrem (Baeza-Yates e Ribeiro-Neto, 1999). Para cada página Web armazenada é mantida uma visão lógica do texto, que é

constituída com uma descrição curta de cada página, contendo informações como: data de criação, tamanho, título e as primeiras linhas da página ou o cabeçalho.

Simplificadamente, uma consulta é respondida realizando-se uma busca binária no conjunto de palavras ordenado na lista invertida. Se a busca possui mais de uma palavra, os resultados têm que ser combinados para gerar a resposta final. Numa técnica estendida, denominada de lista invertida completa (Baeza-Yates e Ribeiro-Neto, 1999), cada palavra possui um ponteiro indicando a sua posição no texto. Essa técnica ocupa muito espaço e aumenta consideravelmente o tamanho dos índices, mas permite que os serviços de busca respondam à busca por frases ou pela proximidade das palavras. Eles conseguem verificar se cada palavra da consulta está próxima entre si na mesma página.

Na composição da arquitetura, existe um módulo destinado à utilização do serviço de busca pelo usuário, fornecendo-lhe uma interface para esse, que é preenchida e enviando a sua consulta para o servidor (do serviço) de busca. O conjunto de palavras recebido da consulta é passado para o motor de busca, que consulta o índice. Este retorna um conjunto de páginas e a freqüência com que cada palavra aparece nas páginas. Esse conjunto de páginas é passado para o algoritmo de classificação das respostas. O algoritmo ordena as páginas e as retorna para o cliente. Deve ser ressaltado que esse processo ocorre em paralelo ao processo de busca e indexação das páginas, constituindo duas preocupações diferentes em um projeto de serviço de busca.

3.2.2 Serviço de Busca AltaVista

O AltaVista utiliza a arquitetura indexadora-vasculhador (figura 3.1): os *crawlers* enviam as requisições para os *sites* Web e recebem as páginas para serem indexadas no módulo indexador[4]. Segundo foi apresentado em SESC (2001), o AltaVista possui algumas heurísticas para tratar a indexação de uma página. Elas são descritas a seguir:

- **Dependência de localização do texto**: textos localizados no início tem um peso maior que um texto no rodapé da página;

[4] Algumas informações desta seção têm por base um relatório escrito por Dirk (2001), não proveniente de fontes oficiais e que foi baseado nos arquivos de ajuda de uma versão usuário do AltaVista, portanto, as informações podem não representar uma realidade.

- **Título da página**: palavras-chave localizadas no título da página possuem um grau de importância diferenciado das palavras-chave localizadas no resto do texto;
- **Marcação META**: considera as palavras-chave contidas na marcação META das páginas HTML, sendo que estas são inseridas pelo próprio projetista da página. Essa heurística é tratada com cuidado, pois é comum os projetistas de páginas inserirem palavras que não refletem o conteúdo da página. Esta marcação pode constituir uma fonte de *spamming*;
- **Popularidade da página**: a página é considerada popular caso seja apontada por *links* vindo de páginas consideradas importantes;
- ***Links* artificiais**: *links* que apontam para páginas inexistentes são considerados *spams*.

O AltaVista trata a indexação da seguinte forma: ele considera o texto inteiro das páginas Web, tratando cada página e cada artigo da USENET (Usenet, 2002) como uma seqüência de palavras. Uma palavra nesse contexto significa qualquer conjunto de letras e dígitos delimitados por pontuação ou qualquer outro sinal não-alfabético; ele utiliza, ainda, o esquema de *stopwords*[5], que, no caso do AltaVista, funciona dinamicamente, ou seja, a lista de *stopwords* não precisa ser cadastrada manualmente. A cada documento indexado, é contado o número de vezes que o termo aparece no mesmo, sendo este valor armazenado no índice. Ao atingir um valor designado antecipadamente, o termo, antes considerado indexável, se torna um *stopword*. Isto significa que ele não terá mais valor para a indexação.

Os termos no índice do AltaVista são distribuídos respeitando a lei Zipf de distribuição de valores, que tem sido utilizada para caracterizar a distribuição de palavras em uma linguagem natural, como na língua inglesa. A lei utiliza algumas premissas: uma linguagem tem poucas palavras que são usadas muito freqüentemente, um grande número de palavras que são usadas constantemente e tem outras tantas quase não usadas.

[5] *Stopwords* são palavras de ligação, auxiliares do tipo artigo, preposição, conjunção, entre outros, não possuindo valor para indexação. Exemplos de *stopwords* são: e, em, para, de, o, a.

As *stopwords* representam a categoria, mencionada anteriormente. Desta forma não são significativas para consultas e não há a necessidade de armazenamento dessas palavras. As palavras que ocorrem raramente nos documentos podem se tornar interessantes para distinguir documentos existentes na base e podem melhorar a resposta para o usuário (Dirk, 2001).

Outra questão de indexação tratada pelo AltaVista está relacionada à duplicidade de informações, ou seja, páginas Web que possuem a mesma informação, tais como documentação de software e notícias. Para detectar duplicatas, existe o algoritmo de detecção que, durante a indexação, compara conjuntos de oito palavras indexadas dos documentos. Quaisquer dois documentos, que possuírem conjuntos de palavras aproximados, são considerados duplicatas. A decisão de qual cópia manter é difícil de ser tomada. Para a documentação de software, a versão mais nova é provavelmente a mais atualizada. Porém no caso de roubo ou cópia de *site*, a versão mais antiga é comumente a verdadeira.

Ao ser enviada ao AltaVista, uma consulta tem os seus termos ou palavras-chave comparados a uma lista de itens compostos pré-computados. Esses itens compostos são palavras que podem ocorrer com maior probabilidade juntas nos documentos, como Rio de Janeiro, São Paulo, etc... Esse passo é necessário para verificar se a consulta se encaixa nessa categoria de palavras. Caso se encaixe, serão procurados na base de índices documentos que tenham esses itens compostos. O próximo passo é a retirada de *stopwords* da consulta, pois essa categoria de palavras ocorre com tamanha freqüência que ela tem pouca importância para diferenciar os documentos na base que atendam a consulta, ou seja, não conseguem limitar a abrangência da resposta da consulta.

Após o tratamento da consulta submetida pelo usuário ela é enviada ao processador de consulta que a submete à base de índice e obtém um conjunto de *links*, que são as respostas relativas à consulta do usuário. Esses *links* necessitam de uma classificação antes de serem retornados, pois é interessante para os usuários receberem os *links* mais importantes primeiro.

Segundo o arquivo de ajuda anexado ao programa de AltaVista pessoal, o AltaVista segue alguns critérios básicos, descritos na seção 3.2.12, para a classificação da importância dos *hits*. Além desses citados na consulta, um outro critério considera se todas as palavras

ou frases especificadas na consulta existem em um documento. Por exemplo, no caso de uma consulta com três palavras, o documento que contiver todas as três palavras especificadas na consulta será classificado em uma posição melhor que documentos somente com uma ou duas palavras; outro critério é o de proximidade. Este considera que no caso das múltiplas palavras ou frases de uma consulta ocorrerem em uma posição aproximada em um documento, este documento é importante para a resposta.

Os *links* com seu grau de importância são ordenados e retornados para o usuário em páginas HTML. Cada página possui um conjunto de *links* a ser escolhido pelo usuário. Esses *links* não são todos retornados para o usuário em uma página. Normalmente são retornados em um conjunto de *links* com o tamanho definido pelo usuário.

3.2.3 Serviço de Busca Google

O Google é um projeto acadêmico em andamento na Universidade de Stanford, constituindo um serviço de busca que também utiliza a arquitetura *crawler-indexadora*. Seu nome deriva da palavra inglesa googol, que significa 10^{100} na língua inglesa. Por ser um projeto acadêmico e por enquanto sem interesses comerciais, seus algoritmos principais são divulgados detalhadamente. Segundo seus projetistas, Steve Lawrence e Page Brin, um dos principais motivos de sua criação foi a necessidade de divulgação de como funciona internamente um serviço de busca para a comunidade em geral, pois todos os outros serviços de busca comerciais funcionam como uma "caixa-preta".

O Algoritmo *PageRank* (Brin e Page, 1998) utiliza a estrutura de links da Web para fornecer uma qualidade de classificação para cada página Web, chamada de Classificação da página. Essa estrutura forma um grafo de links representando a ligação entre as páginas da Web. No grafo de *links*, as páginas são os nós do grafo e os *links* entre as páginas são consideradas as arestas do grafo. Brin e Page (1998) consideram que o grafo de links é um importante recurso que não tem sido costumeiramente usado por outros serviços de busca. A partir do grafo, foram criados mapas de busca contendo cerca de 518 milhões de hyperlinks entre páginas da Web. O mapa permite o cálculo rápido da medida do PageRank, que verifica a importância de citações para determinada página feitas por outras páginas. Segundo Brin e Page (1998), o algoritmo é capaz de classificar 26 milhões de

páginas em poucas horas. O grau de Classificação corresponde a uma medida subjetiva pela qual as pessoas consideram o quão importante uma página é na Web.

O cálculo simplificado do PageRank (Page et al., 1998) pode ser representado pela seguinte equação (3.1) :

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_u} \qquad \text{(Equação 3.1)}$$

em que *u* representa determinada página Web para qual está sendo calculado o seu valor de classificação. R(u) representa o valor obtido de classificação para a página u. B_u representa o conjunto de páginas que apontam para a página u. *R(v)* representa o valor do PageRank obtido para a página *v* que é apontada pela página *u* sendo calculada. N_u representa o número de links para qual u aponta. O termo c é utilizado para a normalização do vetor, para que ele atinja sempre um valor constante. O PageRank forma uma distribuição de probabilidade sobre as páginas da Web. Então, a soma de todas as páginas calculada na Web será constante. Mais detalhes a respeito da forma de cálculo e suas restrições matemáticas podem ser obtidas em (Page et al., 1998).

Os autores indicam uma justificativa intuitiva para a qualidade do algoritmo PageRank. Uma página terá alto valor de PageRank, se existirem diversas páginas apontando para ela, ou se existirem algumas páginas que apontam para ela e possuem um alto valor de PageRank. Essas justificativas podem ser levadas em consideração na prática, pois quando uma página é considerada importante, é citada por várias páginas pela Web, significando que é válido visitar a página. Outro caso é quando um *site* com um grau de importância grande, como o Yahoo, possui citação para a página. A fórmula do PageRank manipula esses casos recursivamente, conseguindo propagar os pesos pela estrutura de *links* na Web.

Uma outra característica é o Texto Âncora. O Google trata os textos dos *links* de uma maneira especial, ao invés de associar o texto do *link* com o URL onde está a página, ele aponta para a página que representa o endereço físico para onde o *link* realmente aponta. Esta característica oferece algumas vantagens: as âncoras freqüentemente fornecem

descrições mais apuradas das páginas Web que as próprias páginas Web e âncoras poderiam existir para documentos que não poderiam ser indexados por um serviço de busca baseado em texto.

3.2.3.1 Arquitetura do Sistema

Uma visão geral da arquitetura do Google é apresentada na figura 3.2. No Google, a busca por páginas para a indexação é feita com a utilização de vários agentes vasculhadores distribuídos. Ela funciona do seguinte modo: do servidor URL são enviadas listas de URLs que são obtidas pelos agentes de recuperação. As páginas Web recuperadas são enviadas para o servidor de armazenamento, que as comprime e armazena em um repositório separado. Cada página Web, no repositório, possui um número ID associado, chamado de docID, que é assinalado sempre que uma nova URL é analisada em uma página Web. Esse ID tem a função de identificar a URL dentro da base de urls mantida pelo Google.

A função de indexação é executada pelo indexador e pelo classificador. O indexador executa várias funções, incluindo a leitura do repositório, a descompressão e análise dos documentos. Cada documento analisado é convertido em um conjunto de ocorrências de palavras chamado de acerto (*hit*). Cada acerto grava a palavra, posição da palavra no documento, uma aproximação do tamanho da fonte da palavra, e o tipo de caixa da palavra. Cada acerto possui um wordID que o identifica no índice.

O indexador distribui esses acertos em um conjunto de estrutura de dados, chamados de "barris", criando um índice direto parcialmente organizado. A outra função importante executada pelo indexador consiste na análise de todos os *links* em cada página Web e no armazenamento de informações importantes sobre cada *link* no repositório para âncoras. O repositório de âncoras contém informações importantes para determinar de e para qual página aponta cada *link* e seu texto relacionado.

No próximo passo, o determinador de URL (figura 3.2) lê o arquivo contendo as âncoras e converte as URLs, relativas às páginas, em URLs absolutos e, neste momento, cada URL absoluto recebe o seu docID. Nesse módulo, o texto da âncora é armazenado em um índice direto que possui ainda outro campo contendo a identificação do documento (docID) para o qual este aponta.

Na arquitetura existe um banco de dados que armazena os *links* extraídos da Web. Ele é formado por pares de docIDs. Um docID identifica a página onde o *link* está e o outro docID indica o endereço para o qual esse *link* aponta. Esse banco de dados de *links* é utilizado, principalmente, para calcular o *PageRank* para todos os documentos indexados pelo Google.

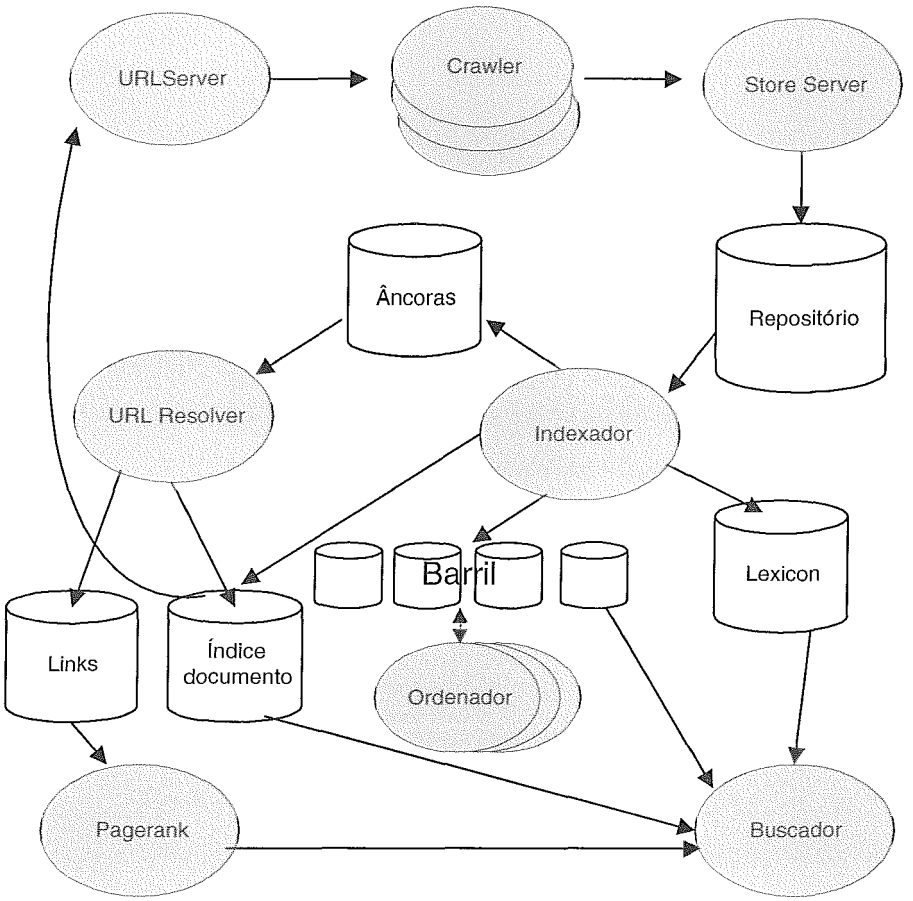


Figura 3.2 – Arquitetura de alto nível do Google

O módulo do ordenador de documentos (figura 3.2) trata os "barris", que são organizados pelo docID, recuperando as palavras pelo wordID inseridos pelo indexador para gerar o índice invertido. O ordenador também gera uma lista de wordID e deslocamentos das palavras dentro dos documentos em um índice invertido.

Existe ainda um módulo chamado de DumpLexicon que pega esta lista invertida junto com o *lexicon*[6] das palavras indexadas, gerado pelo indexador, e gera um novo

[6] Dicionário de palavras gerado pelo indexador de documentos do Google.

lexicon a ser usado pelo buscador. O módulo buscador é executado em um servidor Web e usa o lexicon construído pelo DumpLexicon junto com o índice invertido e o PageRank para processar as consultas.

3.2.3.3 Agente Vasculhador do Google

Um outro módulo importante aborda a recuperação de páginas na Web. Para tratar com milhões de páginas, o Google apresenta um sistema de vasculhamento distribuído. Segundo informações oficiais (SearchEngineWatch, 2001), o Google vasculha as páginas a cada 28 dias, mas leva um tempo adicional de processamento para a página ser armazenada no índice. O Google indexa *sites* pelo seu grau de importância, visitando mais frequentemente os *sites* que obtiveram um valor alto no cálculo do PageRank das páginas.

Um problema importante na recuperação de páginas consiste na busca por DNS dos *sites* Web a serem recuperados. Cada uma das centenas de conexões tem que passar por diversos estados: procurar o DNS do Web *site*; conectar ao computador remoto; enviar a requisição e receber a resposta. Esses fatores transformam o agente em um importante e complexo componente na arquitetura. O agente utiliza entrada/saída assíncrona para gerenciar os eventos e um certo número de filas para realizar buscas às páginas pelos diversos estados. Uma observação importante trata da indisponibilidade de uma página, pois no caso do vasculhador não encontrá-la, ela será apagada do índice do Google.

3.2.3.4 Análise das páginas Web

Um ponto fundamental a ser descrito detalhadamente é a análise das páginas recolhidas pelo *crawler*, a fim de fazer sua indexação. Para a realização da tarefa, o Google utiliza um analisador léxico próprio, robusto o suficiente para tratar com os diversos problemas encontrados para se indexar páginas em toda a Web. Depois do documento ser analisado, ele é armazenado nas estruturas de dados dos "barris". Cada palavra do documento é convertida em um wordID usando o lexicon, que consiste em uma tabela *hash* em memória contendo todas as palavras já indexadas. Após ser encontrado o wordID da palavra, as ocorrências da palavra no documento são traduzidas em uma lista de *hits* e estes são escritos na lista de indexação direta de documentos.

A próxima etapa trata da ordenação dos *hits* encontrados para a geração de um índice invertido. O ordenador pega cada elemento da lista de indexação direta e os organiza

por wordID para produzir uma lista invertida por título e *hits* de âncora e um índice para texto completo invertido. Esse processo ocorre em um barril por vez. Mais informações sobre este assunto são encontradas em (Brin e Page, 1998).

3.2.3.5 Processamento da consulta do usuário

Um ponto primordial é a forma como o Google processa uma consulta enviada pelo usuário e, principalmente, como funciona o seu algoritmo de classificação dos *links* a serem retornados ao usuário. Inicialmente, era imposto um limite máximo de 40000 *links* retornados. Quando esse valor era alcançado, a resposta era automaticamente retornada. Atualmente, não existe mais essa limitação.

Os passos básicos realizados pelo algoritmo são descritos abaixo (Brin e Page, 1998):

1. A consulta é analisada;
2. As palavras da consulta são convertidas em wordID, utilizando o lexicon;
3. Cada palavra da consulta é procurada no início das listas de documentos no barril com estrutura mais simples;
4. Verifica-se a lista de documentos até encontrar um documento que atenda a todos os termos da busca;
5. Caso o documento exista, é calculado o seu *rank* para a consulta;
6. Se o algoritmo estiver pesquisando a estrutura de barril mais simplificada e a lista de documentos tiver terminado, a busca é realizada pelo início da lista de documentos no barril com informações mais completas, para cada palavra da consulta;
7. Enquanto não tiver terminado a lista de documentos, continua a execução do passo 4;
8. Ordenar os *links* retornados de acordo com o algoritmo de ranking. Depois são retornados os dez primeiros para o usuário.

Para efetuar a classificação dos *links* da forma mais eficiente possível, o Google possui uma quantidade maior de informações sobre as páginas que os outros serviços de busca. Cada lista de *hits* inclui a posição do *hit*, a fonte e a informação relacionada à caixa

da palavra no documento. Além dessas informações, são contabilizados os *hits* do texto da âncora e o PageRank do documento. Segundo Brin e Page (1998), o algoritmo foi projetado para que nenhum desses fatores tenha peso maior que o outro no seu processamento.

No caso de uma consulta com apenas uma palavra-chave, o Google procura na lista invertida de *hits* dos documentos a palavra-chave. O Google possui vários tipos de classificação para cada *hit*. Esses tipos são: título, âncora, fonte de texto grande, fonte de texto pequena. Cada tipo possui seu respectivo peso e esses pesos dos tipos formam um vetor indexado pelo tipo. Esse vetor servirá para calcular a importância do documento para a consulta.

O algoritmo contabiliza o número de *hits* encontrado para cada tipo na lista de *hits*, como por exemplo, o número de documentos em que a palavra-chave foi encontrada no título e assim por diante, sendo que, para cada *hit*, é atribuído um peso. Este peso equivale ao número de vezes em que o *hit* aparece como sendo desse tipo. O algoritmo realiza o produto vetorial do vetor de pesos dos *hits* com o vetor dos pesos dos tipos (o valor dos elementos do vetor é conhecido previamente) para calcular um valor final para o documento. Por fim, o valor final do documento é combinado com o valor calculado para esse documento no algoritmo de *PageRank*, resultando em um valor final para o documento.

No caso de consultas do usuário que possuem mais de uma palavra, listas de *hits* devem ser pesquisadas em uma única vez, para que *hits* que ocorram juntos ou próximos em um documento tenham um peso maior que os *hits* que aparecem longe um do outro no documento. Para cada conjunto de *hits* encontrado, um grau de proximidade é calculado. Essa proximidade é calculada com base na distância dos *hits* no documento, ou na âncora do texto. Observa-se que a sua posição é armazenada nos *hits*, possibilitando a sua posterior recuperação para ser realizado o cálculo da proximidade entre as palavras. É atribuído um grau de proximidade entre o conjunto de palavras da consulta, que varia desde "junto" até "não tão próximo" no documento.

A contagem de valores não é feita apenas para cada tipo de *hit*, mas para cada tipo e proximidade. Cada par de tipo e proximidade tem um peso atribuído, sendo armazenado em um vetor. É criado um vetor contendo o número de vezes que os termos aparecem, sendo calculado um produto vetorial desse vetor com o vetor de tipo e proximidade, gerando um

valor que será utilizado para a classificação do documento para a consulta. Do mesmo modo que a classificação simples, esse valor é combinado com o valor obtido pela página no algoritmo de PageRank, resultando em um valor final para cada documento. Para os dois casos, o valor obtido para cada documento servirá para a ordenação da resposta. Após essa ordenação as respostas são retornadas ao usuário.

3.3 Serviços de MetaBusca

Um serviço de metabusca é realizado por um servidor Web que possui um programa residente que recebe uma consulta do usuário e a envia a diversos serviços de busca, diretórios Web e outros bancos de dados na Web. A sua principal função é fornecer ao usuário a capacidade de enviar a mesma consulta para diversos serviços de busca ao mesmo tempo. Ele é capaz de ordenar os resultados por diferentes atributos como *host* da página, palavras-chave, data e popularidade. Esse serviço co0leciona as respostas das fontes, realiza a ordenação pelos atributos citados acima e retorna um resultado unificado para o usuário.

Duas vantagens são apresentadas por Hower e Dreiling (1997): os serviços de metabusca podem consultar serviços de busca desconhecidos pelo usuário e aceleram o processo de busca, pois enviam e processam as consultas em paralelo. Na tabela 3.2, são mostrados alguns serviços de metabusca, seus endereços URL e o número de fontes utilizadas (Baeza-Yates e Ribeiro-Neto, 1999).

Tabela 3.2 – Serviços de metabusca

Serviço de MetaBusca	URL	Fontes Utilizadas
Cypber 411	www.cyber411.com	14
DogPile	www.dogpile.com	25
Highway61	www.highway61.com	5
Inferenc Find	www.infind.com	6
SavvySearch	www.search.com	Mais que 13
MetaCrawler	www.metacrawler.com	7
MetaFind	www.metafind.com	7
MetaMiner	miner.uol.com.br	13
MetaSearch	www.metasearch.com	N/D

Segundo Lam (2001), um serviço de metabusca é mais informativo que um simples serviço de busca, pois retorna um resultado mais abrangente que apenas um serviço de busca. Embora o número de resultados retornados do serviço de metabusca possa ser ajustado, esse número é ainda limitado pela eficiência de se fornecer uma resposta rápida. Essa limitação leva à possibilidade de que, em uma metabusca, o resultado final apresentado não atenda às necessidades do usuário, pois a página procurada pode estar no conjunto de páginas não retornadas pelo serviço. Apesar do Google apresentar um alto grau de indexação (tabela 3.1), ele não consegue abranger toda a Web e por isso, os serviços de metabusca podem melhorar os resultados dos serviços comuns de busca.

No trabalho apresentado por Lawrence e Giles (1998), é realizado um estudo detalhado abordando a abrangência dos índices dos serviços de busca. Ao final do estudo, é indicada a utilização de um serviço de metabusca para melhorar os resultados das consultas dos usuários. Os autores chegaram à conclusão de que a realização de uma metabusca com os seis serviços de busca (HotBot, AltaVista, Nothern Light, Excite, InfoSeek e Lycos), pesquisados no trabalho, poderia aumentar em até 3.5 vezes o alcance de uma busca na Web. Uma maior abrangência dos índices consultados pode ser importante no caso de buscas mais específicas, como um determinado artigo ou informações localizadas em uma página pessoal de um determinado pesquisador.

Os serviços de metabusca enfrentam alguns problemas para a classificação das respostas dos serviços de busca. O principal está relacionado à impossibilidade de se conhecer como eles funcionam internamente. Esse conhecimento facilitaria a definição de qual serviço utilizar para consultas mais específicas. Outro problema está relacionado ao desempenho. O tempo de uma metabusca pode ser superior ao de uma busca comum, tornando-se um impedimento para o seu uso, no caso de se estar interessado em velocidade nas respostas às consultas.

3.4 Diretórios Web

Os Diretórios Web funcionam de forma semelhante a um catálogo telefônico, dividindo os endereços Web por assuntos. Eles também são conhecidos como catálogos, páginas amarelas, ou diretórios de assuntos. Os diretórios são taxonomias hierárquicas que

classificam o conhecimento humano, formando categorias onde são enquadradas as páginas Web.

Na maioria dos casos, o processo de indexação feito pelos diretórios Web é diferente dos serviços de busca. Nos diretórios Web, a página é enviada pelo usuário, onde é revisada e se for aceita é classificada em uma ou mais categorias da hierarquia. Essa é uma desvantagem desse tipo de classificação, pois o esforço despendido para a classificação é muito grande.

O mais popular diretório Web é o Yahoo, com aproximadamente um milhão e quinhentas mil páginas classificadas, que permite a busca por palavras-chave e retorna as páginas que atendem à consulta, classificadas por categoria. Alguns diretórios Web são mostrados na tabela 3.3.

Tabela 3.3 – Principais diretórios Web – (SearchEngine Watch ,2001)

Diretório Web	Editores	Categorias	Links
Open Directory	36,000	361,000	2.6 milhões
LookSmart	200	200,000	2.5 milhões
Yahoo	Mais de 100	N/D	1.5 até 1.8 milhões
AskJeeves	150	N/D	128 milhões

As categorias de classificação dos serviços de diretórios Web são as seguintes:

- Editores – número de pessoas envolvidas na indexação do diretório. O Open Directory é um esforço mundial para a classificação de páginas Web;
- Categorias – número de categorias do diretório;
- Links – mostra o número de URLs únicas existentes no diretório. No serviço de diretório Yahoo, há a possibilidade da repetição de um link, ou seja, ele pode aparecer em mais de uma categoria. Por isso, é feito um somatório aproximado de todos os links existentes no serviço.

Há ainda *sites* Web focados em determinados domínios do conhecimento, como negócios, notícias e os mais importantes que são os sites de busca bibliográfica ou biblioteca digital. Um dos *sites* mais conhecidos de busca bibliográfica é o ResearchIndex,

que tem como objetivo a busca por artigos e trabalhos científicos na área da computação. Ele utiliza diversos esquemas de colaboração de interesses entre usuários e algoritmos de classificação de documentos por citação em outros documentos (Bollacker et al., 1998).

3.5 Comparação dos serviços de busca

Nas seções 3.2.1.3 e 3.2.1.4 foram descritos os dois principais serviços de buscas atualmente: o AltaVista e o Google. A comparação desses serviços possui relevada importância para a arquitetura proposta neste trabalho, pois o serviço de busca é um componente fundamental na implementação da mesma. Nesta seção, é feita uma comparação, citando e detalhando algumas características consideradas importantes para o trabalho em questão. Essas características se tornaram determinantes na escolha do serviço de busca.

A primeira e principal vantagem do Google é por ele ser produto de um projeto acadêmico. Essa vantagem reside no fato da principal função do seu projeto, continuamente desenvolvido na Stanford University, ser mostrar para o mundo acadêmico e, até para usuários comuns, um protótipo de um serviço de busca, mas que tenha o mesmo desempenho e eficiência de um serviço de busca comercial. Por esses motivos, o Google fornece abertamente descrições detalhadas do funcionamento dos seus principais algoritmos, como algoritmo de indexação, de classificação de links e dos agentes que realizam a busca por páginas Web para serem indexadas.

O Google não possui, por enquanto, interesses comerciais, ou seja, não precisa fazer marketing em suas páginas para a manutenção do *site*. O principal tipo de *marketing* realizado na Web são os *banners*[7] existentes em quase todas as páginas residentes em sites comerciais, incluindo sites de serviços de busca. No caso da implementação desta dissertação, em que são feitas consultas remotamente, isto significa que os usuários não acessam a página de busca e de resposta diretamente. Portanto, não podem visualizar e acessar as páginas indicadas pelos *banners*, burlando a propaganda das páginas, ou seja, a principal receita de um serviço de busca. Como não existem essas restrições comerciais no

[7] Faixas inseridas em páginas Web funcionando como um outdoor para um determinado site ou produto.

Google, este pode ser utilizado por ferramentas que realizam mineração a partir de resultados de busca.

Como mostrado na tabela 3.1, o Google possui aproximadamente 1.687 bilhões de páginas indexadas, mais que o dobro que o AltaVista, terceiro lugar na classificação dos maiores bancos de índices dos serviços de busca. Essa abrangência é importante para a obtenção da métrica de *recall*, ou seja, o serviço de busca pode procurar o maior número de documentos relevantes entre todos os relevantes na Web.

O Google possui ainda algumas outras vantagens, a exemplo do algoritmo *PageRank*, que é bastante poderoso na classificação da importância das páginas e também outras heurísticas utilizadas na classificação, tais como: considerar a proximidade entre palavras nas buscas por frases; considerar o tamanho das fontes dentro das páginas.

O *site* search iQ (SEARCHIQ, 2001) indica, em ordem de classificação, o Google, com IQ[8] de 140, e o AltaVista, como IQ 130, como os melhores serviços de busca disponíveis atualmente. Inicialmente, o AltaVista era o mais popular e o melhor serviço de busca. Atualmente, o poder de indexação e classificação de páginas do Google o tornaram o serviço de busca de texto mais importante na Web.

3.6 Conclusões

A pesquisa científica relacionada aos serviços de busca avança rapidamente. Todavia a maior parte dessa pesquisa está concentrada na área comercial. Conseqüentemente, a maior parte dos algoritmos utilizados pelos serviços de busca não é conhecida e nem é divulgada. O principal motivo está relacionado ao fato de que, se o núcleo dos serviços de busca comerciais forem divulgados, eles poderão ser copiados e criados novos serviços de busca concorrentes. Por esse sigilo nos algoritmos, torna-se bastante difícil realizar a análise ou comparação entre os serviços de busca.

Entretanto, há um único serviço de busca que divulga amplamente seus algoritmos, o Google, permitindo a sua análise e um estudo mais aprofundado da sua estrutura de funcionamento e de suas características.

No artigo apresentado por Lawrence e Giles (1999), os autores prevêem que os serviços de busca tentarão, nos próximos anos, indexar ao máximo a parte pública da Web.

[8] Índice calculado pelo site como quociente de respostas importantes retornadas.

Isso será possível pelo barateamento do custo de armazenamento, velocidade atual das máquinas e, principalmente, a melhora na tecnologia de indexação de páginas.

Uma outra categoria existente, e que deverá se expandir, é a de serviços de busca específicos por determinados domínios. Eles são eficientes por algumas características peculiares à sua categoria de serviço de busca. Uma delas trata da restrição do domínio, pois é menor a quantidade de páginas a indexar, tornando os serviços de busca mais abrangentes dentro dessa área específica. Além disso, os serviços orientados a determinadas áreas do conhecimento são capazes de atualizar com menor latência os índices da sua base, pelo mesmo motivo apresentado acima. Um exemplo desses serviços é o ResearchIndex (Bollacker et al., 1998), especializado em artigos de ciência de computação, considerado uma biblioteca digital. Ele utiliza um algoritmo com filosofia semelhante ao do Google, classificando a importância do artigo científico pela quantidade de citações que o artigo possui em outros artigos.

Os diretórios Web também apresentam uma boa expectativa, preenchendo o nicho da busca por informações populares, como páginas sobre uma cidade, sobre turismo, etc...Eles são muito úteis, nesse caso, por causa dos poucos resultados de relevância retornados ao usuário, facilitando a escolha do link por parte deste.

Uma categoria também importante são os serviços de metabusca. Segundo Lawrence e Giles (1999), os serviços de busca procuram retornar as respostas para o usuário da forma mais rápida possível, sem a preocupação de perder tempo na filtragem da resposta para o usuário. Os serviços de metabusca podem realizar esse processamento de filtragem na máquina do cliente, filtrando o resultado das consultas e tentando encaixar, com o apoio do usuário, a consulta em um determinado contexto.

Algumas características, se melhoradas, podem tornar os serviços de metabusca mais poderosos: a capacidade de combinar os resultados dos diversos serviços de busca, como também a capacidade de evoluir com as constantes mudanças dos serviços de busca utilizados.

A área de metabusca é um pouco diferente da área de serviço de busca. Naquela categoria, existe uma quantidade maior de ferramentas que surgiram no meio acadêmico, e, normalmente, fornecem artigos científicos explicando seu funcionamento. Entre as ferramentas podemos citar o SavvySearch (Howe e Dreilinger,1999) e o Inquirus 2,

conforme citado na seção 2.4.6, que foi explicado na seção de trabalhos relacionados por ter diversos avanços na área de metabusca e indexação de páginas.

Nosso trabalho se encaixa nesta categoria de metabusca, pois depende de um serviço de busca e de chamar uma arquitetura específica para busca de componentes. A ferramenta, proposta nesta dissertação, segue uma tendência dos serviços de metabusca de realizar um processamento na máquina do cliente para melhorar as respostas vindas dos vários serviços de busca, apresentando uma qualidade melhor na resposta ao usuário.

Capítulo 4 - Arquitetura CompAgent

4.1 Introdução

A Engenharia de Domínio (ED), uma subárea de pesquisa da Engenharia de Software, corresponde ao processo que estuda um domínio de conhecimento com perspectiva para a construção de componentes, propondo um processo completo para o desenvolvimento de componentes reutilizáveis (Arango, 1988).

Um domínio é definido em SEI (2000) como um termo utilizado para denotar ou agrupar um conjunto de sistemas, ou áreas funcionais dentro de sistemas que exibam funcionalidade similar. O processo de ED possui os seguintes objetivos (SEI, 2000):

- encontrar os requisitos comuns e variantes dos sistemas no domínio;
- projetar uma arquitetura genérica que organiza os requisitos em componentes;
- comprar ou construir componentes reutilizáveis de terceiros que se encaixem na arquitetura e satisfaçam os requisitos.

Todos os componentes gerados através de diversos processos de engenharia de domínio são submetidos à gerência de componentes, que envolve atividades como a aquisição, aceitação, classificação, catalogação e certificação dos componentes (Braga, 1999). Nessa fase de gerência, eles são armazenados em uma biblioteca de componentes. Esse processo de geração de componentes para reutilização é conhecido como desenvolvimento *para* reutilização, que é considerada a atividade fim do processo de ED.

Posteriormente, esses componentes serão recuperados para reutilização. Essa fase, denominada como engenharia de aplicação (EA) (Miller, 2000), é tratada como desenvolvimento *com* reutilização, onde os usuários (Engenheiros da Aplicação) buscam e recuperam os componentes armazenados na biblioteca para serem utilizados no processo de construção do software.

Um processo de ED possui as seguintes etapas (Braga, 2000): análise do domínio, projeto do domínio e implementação do domínio. Nessas etapas é muito importante a

obtenção de informações para a efetivação de cada etapa da forma mais completa e abrangente possível.

A etapa mais suscetível à recuperação de informação é a de análise de domínio, pois é nesta etapa que são feitas as definições dos principais conceitos do domínio e criação de um modelo de características desse, que consiste em um modelo que captura as suas abstrações. Para realizar essas definições e abstrações, necessita-se de uma grande quantidade de informações, que podem ser extraídas de diversas fontes como livros, padronizações, aplicações existentes e especialistas dessa área do conhecimento. Portanto, esta etapa é a mais importante no contexto desta dissertação, pois nela se faz necessário o suporte de um sistema/mecanismo para a recuperação de informação na Web proposta.

Outra fase da reutilização de software que também tem um grau de importância para o contexto da dissertação, é o de Engenharia de Aplicação (EA). Nesta fase, o usuário necessita localizar e recuperar componentes para a construção da sua aplicação a partir de um domínio já modelado anteriormente. A ferramenta apresentada na dissertação também fornece a funcionalidade de localização, filtragem e recuperação de informações para o processo de EA.

Recentemente, a EA tem utilizado a técnica de desenvolvimento baseado em componentes (DBC) (Braga & Werner, 2000), que é uma área de conhecimento que vem sendo estudada como uma forma de utilizar componentes para o desenvolvimento de software. A definição de DBC é a base para a construção de aplicações na área de EA. Segundo Sametinger (1997), o DBC surgiu como uma inovação para o contexto do desenvolvimento de software, objetivando a "quebra" dos blocos monolíticos existentes na maioria dos produtos de software em componentes interoperáveis. Desse modo, reduz a complexidade no desenvolvimento, assim como os custos, através da utilização de componentes que, a princípio, seriam adequados para serem utilizados em outras aplicações.

Segundo Braga (2000), o DBC tem como principal objetivo o desenvolvimento de aplicações por partes já existentes. Ainda a respeito do DBC, Jacobson et al. (1997) afirmam que esta tecnologia reduz a complexidade, assim como os custos do desenvolvimento de software, através da reutilização de componentes testados

exaustivamente. Um estudo mais completo sobre DBC e componentes pode ser encontrado em (Braga, 2000) e (Braga et al., 2000).

A Internet, principalmente a Web, considerada como o maior repositório de informações existente, é uma fonte natural para a busca de informação para a realização do processo de engenharia de domínio e também para a busca de componentes para realizar o processo de EA (Costa, 2000). Entretanto, como exposto nos capítulos anteriores e também por diversos autores, como (Baeza-Yates e Ribeiro-Neto, 1999), a recuperação na Web apresenta diversos problemas e desafios que ainda estão em aberto. Alguns desses problemas estão diretamente relacionados à incalculável quantidade de informações armazenadas na Web.

O problema de recuperação de informações na Web, em especial o problema de busca de componentes na Web e em biblioteca de componentes, constitui uma área de pesquisa complexa com diversos desafios a serem superados. Esses desafios envolvem questões de Banco de Dados (recuperação de informação na Web) e Engenharia de Software (Engenharia de Domínio, Engenharia de Aplicação).

Neste capítulo, é apresentada uma visão geral do mecanismo implementado e os requisitos necessários para o apoio à realização do processo de engenharia de domínio/engenheiro de aplicação. Na próxima seção são mostrados tais requisitos para um mecanismo de apoio ao usuário e, na seção 4.3, a ferramenta CompAgent implementada para atender esses requisitos. Nas seções seguintes serão descritos as técnicas utilizadas e os respectivos módulos que as utilizam.

4.2 Requisitos para o apoio à recuperação de informação em ED/EA

A questão fundamental da dissertação aborda o apoio ao usuário na busca e recuperação de informações em um ambiente de suporte aos processos de ED e EA. O usuário que, dependendo do contexto, pode ser o engenheiro de domínio ou engenheiro de aplicação, sente-se "perdido" quando percebe a necessidade de encontrar componentes ou informações importantes para a continuação das atividades do processo de ED ou de EA. Essa busca por componentes envolve a consulta a diferentes formas de descrições

heterogêneas de componentes dentro de um vasto espaço de busca como a Web, tornando a procura extremamente difícil. Os componentes podem ser descritos por arquivos XML, arquivos texto, tags HTML, entre outros.

Existem duas abordagens para a busca de componentes: a primeira consiste na busca na Web, baseada no conjunto de métodos publicados (interface) dos componentes pelos seus responsáveis; e a segunda na busca sobre uma biblioteca de componentes local, normalmente gerada por um processo de ED, que fornece uma descrição semântica dos componentes e do seu armazenamento.

A abordagem de busca na Web é a mais interessante para a dissertação, pois nesta se tem a possibilidade de obtenção da maior quantidade possível de componentes e que, se passado por um bom processo de filtragem, poderá ser obtida uma boa qualidade nestes componentes. Observando que existem alguns problemas típicos das próprias características da Web e que podem ser considerados como requisitos para uma ferramenta genérica de apoio a busca de informações (Lawrence, 2000) (Gudivada et al.,1997), entre eles podemos citar:

- **Respostas livres de contexto**: as respostas retornadas às consultas do usuário são muito genéricas, pois são baseadas em buscas realizadas sobre uma ou várias palavras-chave enviadas pelo usuário. As consultas, por sua vez, não estão associadas a nenhum contexto específico, e por isso são retornadas livremente pelos serviços de busca. Portanto, na maioria das vezes, os componentes não se adequam às necessidades dos usuários, pois são retornados em muita quantidade, mas não estão ligados ao domínio (contexto) sendo modelado;

- **Semântica dos componentes:** as semânticas publicadas na Web pelos desenvolvedores dos componentes são muito simplificadas quanto na sua representação. O componente precisa representar, na sua interface publicada, a sua semântica e também o domínio, ou domínios através do(s) qual(is) o componente melhor se integra. A semântica do componente, estando bem definida, o torna mais acessível às consultas do usuário, mesmo que estas não sejam específicas a um determinado domínio de aplicação;

- **Preferências dos usuários**: nenhum serviço de busca tem a preocupação de guardar informações a respeito do usuário que o está acessando. O principal motivo é a enorme quantidade de usuários que o acessam, inviabilizando o armazenamento de

informações a respeito de todos. Uma solução para tal problema poderia ser a criação de *cookies*[9]. Entretanto, um serviço de busca não tem interesse em armazenar tais informações detalhadas, pois já possui uma arquitetura muito complexa, como mostrado no capítulo 3, dificultando a inclusão de mais uma funcionalidade. No entanto, o armazenamento de informações seria importante para prover um contexto para a sua consulta e, por conseqüência, prover a personalização dos componentes apresentados ao usuário.

- **Busca orientada:** o usuário, principalmente ao iniciar a análise de um domínio, não possui referências a respeito sobre o que seria interessante para consultar e obter informações a respeito do domínio. Caso ele decida consultar um serviço de busca, poderá não obter imediatamente (nas primeiras páginas de resposta) informações que possam ser consideradas importantes. Portanto, algumas vezes, faz-se necessário a orientação, por parte de algum especialista ou responsável pelo domínio, de fontes que sejam importantes para o entendimento do domínio. Essas fontes, além da importância das suas informações, já foram validadas por algum usuário com um conhecimento mais profundo a respeito desse domínio.

- **Colaboração entre usuários:** alguns usuários podem considerar que certos componentes são importantes para outros usuários do mesmo domínio, com perfil semelhante ao deles. Essa forma de colaboração é importante para o usuário no momento da escolha de qual componente pode ser interessante para a realização do processo de EA. O mesmo pode ocorrer caso o usuário considere importante para outros usuários páginas HTML com informações para auxiliar no processo de ED.

A abordagem baseada em uma biblioteca de componentes não apresenta algumas das desvantagens citadas anteriormente, como, por exemplo, as respostas livres de contexto e problemas na semântica dos componentes. Entretanto, existe uma limitação grande em relação a ser restrita à apenas componentes previamente e localmente armazenados. Às vezes, essa quantidade de componentes pode ser insuficiente para a construção das aplicações. Todavia, podem existir componentes já desenvolvidos por terceiros e que estão disponíveis para *download*. Porém a biblioteca em geral está restrita aos desenvolvidos localmente. Percebe-se, portanto, a necessidade de uma tecnologia associada às bibliotecas

[9] Arquivo criado pelo site na Web, na máquina do usuário, contendo informações a seu respeito.

de componentes que agregue essa funcionalidade de compartilhamento de componentes remotos, permitindo que estes sejam recuperados remotamente e que possam servir para a realização do processo de ED ou EA.

O processo de ED, devido ao seu grau de complexidade, necessita de ferramentas que o apóiem durante toda a sua realização, desde a análise até a implementação de modelos do domínio. A este conjunto de ferramentas se atribui o termo de "ambiente de suporte ao processo de engenharia de domínio". Os usuários desses ambientes ainda necessitam de um mecanismo, de preferência interna, que os apóiem na busca de informações para o processo de ED e na busca de componentes para o processo de EA.

O sistema de busca de informações tem que apresentar diversos requisitos que atendam à necessidade das suas diversas classes de usuários tais como analistas de domínios, projetistas de domínio e engenheiros de aplicação. O principal deles está relacionado à capacidade de realizar buscas tanto para a Web como para uma biblioteca de componentes. Mas além de satisfazer esse requisito principal, ela deve ser capaz de tratar os outros problemas relacionados às duas abordagens principais, conforme apresentados anteriormente.

Através da proposta de desenvolvimento de uma ferramenta apresentada nesse trabalho, procura-se superar esses problemas e atender aos diversos requisitos identificados, utilizando-se técnicas diferentes, que existem em diversas áreas de pesquisa da ciência da computação. Algumas técnicas como agentes (Maes, 2000), mediadores (Wiederhold, 1992), modelagem de usuário (Rich E., 1983), filtragem colaborativa de informações (Ord e Kim, 1998) e *relevance feedback* (Paepcke et al., 1999), são utilizados na especificação da CompAgent (Figura 4.1) um sistema de apoio à busca e recuperação de informações na Web e componentes publicados na Web. Na próxima seção serão descritas em detalhes as técnicas envolvidas e como elas são abordadas na dissertação.

4.3 Visão Geral do funcionamento da ferramenta CompAgent

A ferramenta CompAgent é descrita por um conjunto de módulos de software e um conjunto de usuários que a utilizam. Uma visão geral da ferramenta é mostrada na Figura 4.1. Como mostrado nesta figura, os seguintes módulos a compõem (Costa, 2000): Modelagem de Usuário; Agente de Busca; Filtragem; Recomendação Colaborativa; Agente

de Julgamento. Esses módulos possuem um forte acoplamento para a realização de suas tarefas e estão agrupados nas seguintes camadas: interface; processamento interno; módulos externos. Estas camadas se comunicam entre si (figura 4.1).

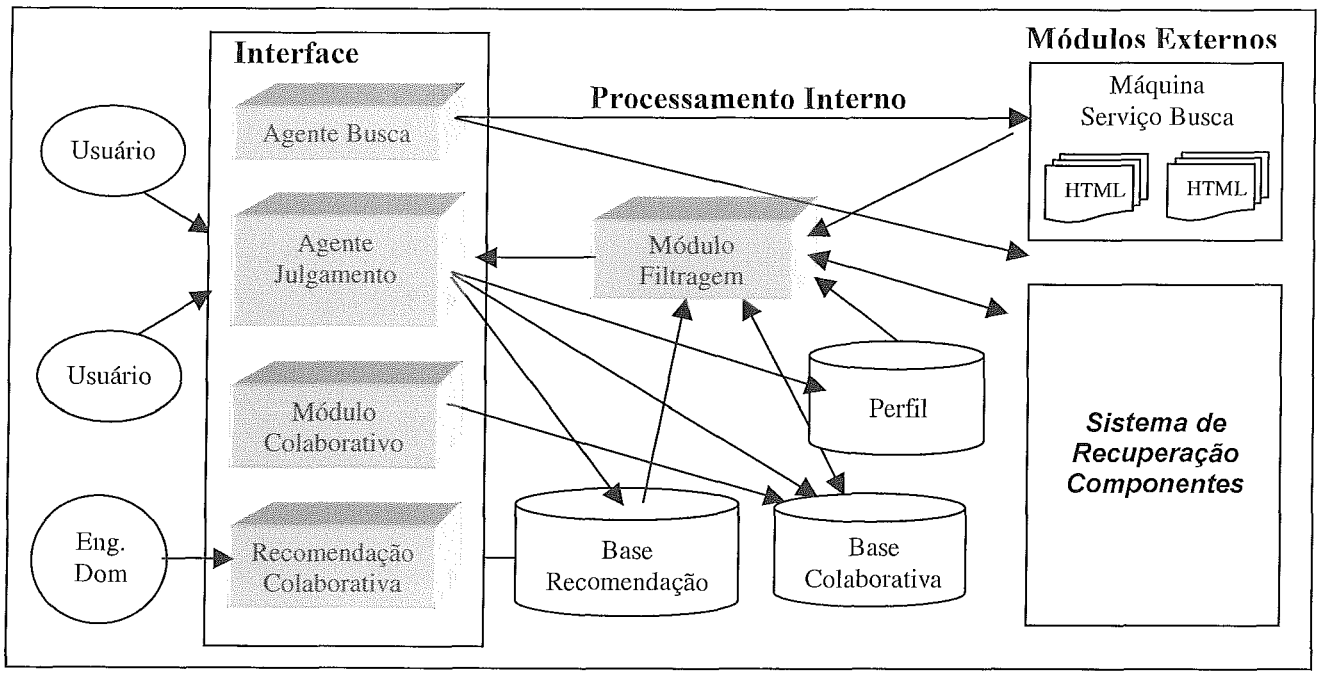


Figura 4.1 – Visão geral da CompAgent

A forma como é feita a comunicação entre os módulos é mostrada esquematicamente na figura 4.2, que apresenta uma visão mais detalhada sobre o funcionamento da ferramenta. Existem, basicamente, dois tipos de interação: a primeira ocorre entre o usuário e os módulos de software (mais especificamente os módulos de interface) e a segunda, internamente, entre os módulos de software.

A interação ocorre da seguinte forma: quando o usuário acessa pela primeira vez o ambiente de modelagem de ED/EA, ele deve inserir uma senha que servirá para os próximos acessos.

Na primeira vez que o usuário vai modelar (trabalhar) o domínio criado no ambiente, ele deve responder a um questionário com informações a respeito de seu grau de conhecimento sobre o domínio, objetivos na modelagem do domínio, contextos que o interessam e, o mais importante para a ferramenta, as palavras-chave que lhe interessam no domínio e seu respectivo grau de importância. Esse questionário formará o perfil do usuário

e servirá para, a partir desse, encontrar os usuários com interesses similares e os inserir num mesmo estereótipo.

O usuário, ao necessitar de alguma informação ou algum componente na Web, aciona a busca de informações na Web, preenchendo nos campos do formulário a principal característica que se deseja no componente, o número de links que espera ser retornado e o nível da filtragem que o sistema deve executar sobre os links retornados. Então o agente de busca se conecta a um serviço de busca, recupera as páginas e também se conecta a um sistema de publicação e recuperação de componentes, o ComPublish (Souza, 2002), explicado detalhadamente na seção 4.4.5.1, para buscar e recuperar os componentes que atendem à consulta.

De acordo com a descrição dos serviços de busca Google e AltaVista, feita no capítulo 3, junto com uma comparação, entre estes na seção 3.6, concluiu-se que o Google seria a escolha mais adequada a ser utilizada na dissertação, como um servidor de consulta a documentos Web.

Após a recuperação das páginas pelo serviço de busca Google, o agente de busca realiza uma análise das páginas HTML, retirando informações como o título do link, a descrição do link e, caso exista, uma descrição manual do link feita pelo usuário que inseriu o link na base de links do Google. Cada componente retornado pelo sistema Compublish é representado por um documento XML, do qual também é feita uma análise, sendo extraídos atributos como nome, categoria, data, autor, fase de desenvolvimento, linguagem, descrição, entre outros.

Essas informações, obtidas do Google e do ComPublish, são passadas para o agente de filtragem que as utilizará nas suas heurísticas. Algumas das heurísticas são: comparar todas as palavras extraídas das páginas retornadas com o perfil do usuário corrente, para poder encontrar os links mais importantes dentro do contexto do usuário; acessar a base de filtragem colaborativa e encontrar links ou componentes que já tenham sido selecionados por usuários similares; também é acessada a base de recomendação colaborativa, onde são procurados links e componentes inseridos pelo engenheiro do domínio. Mais detalhes do processo de filtragem são mostrados no capítulo 5, inclusive com um exemplo do funcionamento da arquitetura.

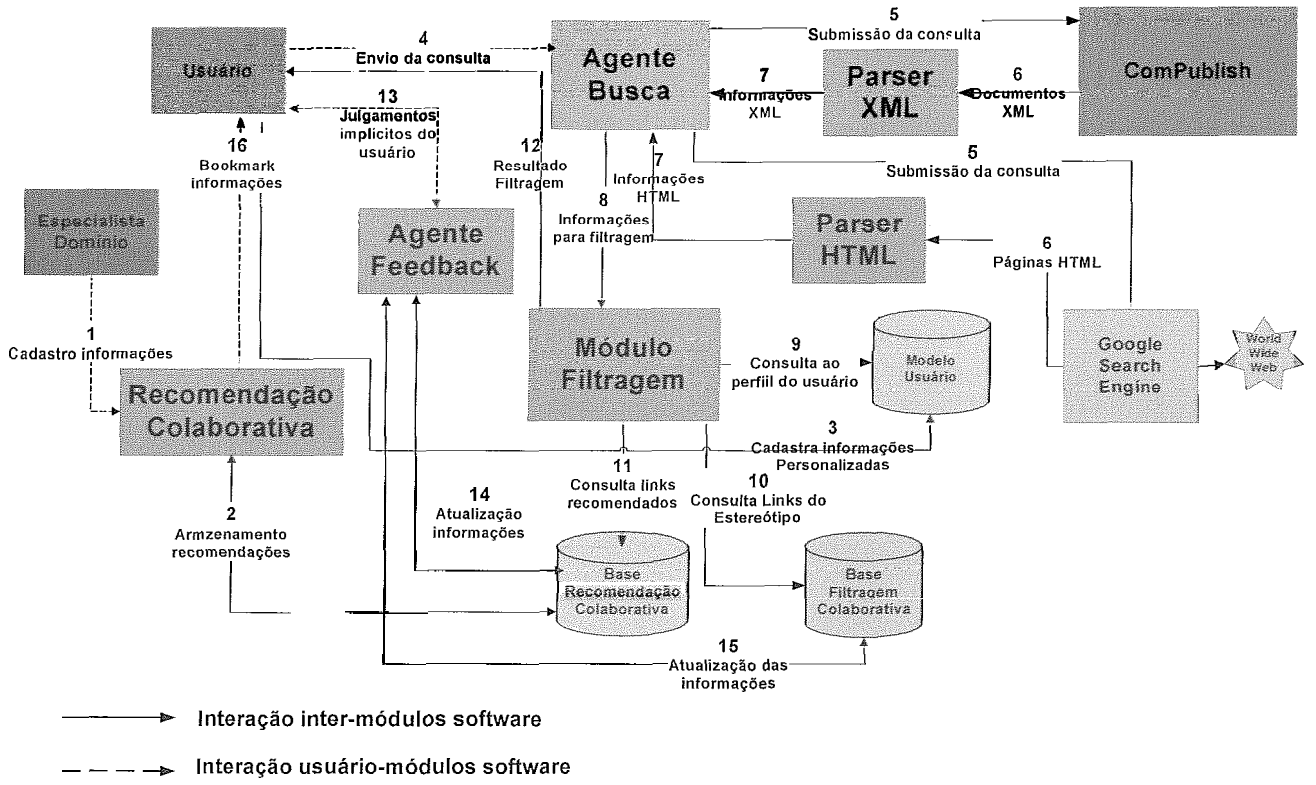


Figura 4.2 - Interação entre usuário e os módulos de software e entre os módulos de software.

Todos os links selecionados durante o processamento das heurísticaspassam ainda por um cálculo, que indica o valor de classificação de cada link para a consulta do usuário. O cálculo é baseando em dois atributos existentes em todas as bases, para cada link. O primeiro é o grau de importância do link, que trata do valor de importância do link dentro do domínio, variando de zero a um. O segundo trata do peso do link, onde são consideradas todas as vezes que o link já foi selecionado por usuários com perfil semelhante. Um terceiro fator considerado é a importância de cada heurística, como por exemplo, se o link encontrado estiver na base de recomendação colaborativa, o link terá peso 4, ou seja, o valor de classificação já calculado para os dois atributos será multiplicado por quatro. Esse peso foi escolhido empiricamente, pois se considera que se ele foi indicado por um usuário mais experiente, ele deve ser consideravelmente mais importante para o usuário comum. Essa multiplicação por peso quatro aumentará sua importância entre o conjunto de links a serem filtrados.

Existe ainda um quarto fator que consiste na posição, calculada pelo Google, do link nas páginas retornadas. A posição retornada do Google é importante, pois o próprio possui um algoritmo interno para a atribuição de importância das páginas na Web. Então, atribui-se um certo grau de confiança à resposta retornada pelo algoritmo do serviço de busca. Esta posição é somada à classificação do link já obtida anteriormente. Um processo semelhante é adotado para os componentes retornados pelo sistema ComPublish, sendo que a posição retornada pelo servidor de consulta não é considerada, pois o ComPublish não retorna seus componentes de forma ordenada, tornando-se indiferente a posição do componente na lista.

Os links são classificados, de acordo com o seu valor de classificação anteriormente calculado, e depois filtrados, com o nível de filtragem inserido pelo usuário durante a submissão da consulta. Após a filtragem, os links são retornados em um *browser* para o usuário. Nesse momento o comportamento do usuário é monitorado. Cada link que o usuário selecionar, ou seja, clicar no link para visualizá-lo, é automaticamente inserido na base de filtragem colaborativa. Caso o link já exista na base colaborativa, o seu peso é acrescido de uma unidade e, caso não exista, ele é inserido na base com as informações extraídas das páginas retornadas pelo Google. Caso o usuário que esteja visualizando seja o engenheiro de domínio, a ferramenta age de maneira diferente, inserindo o link também na base de recomendação colaborativa. Vale ressaltar que o único que possui direito de inserir nessa base é o usuário com papel de engenheiro de domínio.

Uma outra forma de interação ocorre entre o usuário e o módulo de recomendação colaborativa. O usuário, no papel de engenheiro do domínio, possui a permissão de cadastrar os links que considerar importantes para o usuário comum, ou até para os outros engenheiros do domínio do projeto. Ele pode cadastrar informações de dois tipos: páginas HTML contendo informações para o usuário; componentes armazenados no sistema ComPublish importantes para o processo de ED/EA. Em ambos os casos são fornecidas informações como: título, URL, descrição, tipo do link e o grau de importância.

O usuário modelando o domínio também pode interagir diretamente com o módulo. Ele pode consultar um *bookmark* contendo todos os links inseridos pelo(s) engenheiro(s) de domínio e, a partir deste, começar a navegação por informações importantes para ele ou, então, começar a compor sua biblioteca de componentes para a construção da aplicação. Ao gerar o bookmark, ele pode escolher se deseja ver somente os links HTML, ou os

componentes ou, ainda uma lista com os dois. Desse bookmark também é retirado um retorno implícito do usuário, que servirá para alimentar a base de filtragem colaborativa e a própria recomendação colaborativa.

4.4 Técnicas utilizadas na Arquitetura CompAgent

A OdysseySearch, apresentada por Braga (2000), é uma ferramenta implementada no contexto de um ambiente de ED para apoio ao usuário na recuperação de informações. Ela é constituída por um sistema de múltiplos agentes que possuem as seguintes funções:

- realizar o armazenamento e a busca por informações de um dado domínio, de tal forma que esse domínio seja relevante para a aplicação sendo especificada pelo engenheiro de software;
- buscar por informações em domínios correlatos ao domínio em questão;
- ser personalizada, ou seja, adquirindo informações a respeito do interesse de seus usuários e adaptando suas ações de acordo com esses interesses;
- prover persistência, salvando seu estado de forma que possa guardar as últimas ações dos usuários para melhorar as próximas.

A ferramenta CompAgent, proposta nesta dissertação, refina a OdysseySearch, tendo como principal extensão a capacidade de acessar informações espalhadas pela Web e componentes remotamente, mas mantendo a mesma filosofia da primeira de buscar e apresentar apenas as informações consideradas relevantes para o usuário.

As subseções a seguir apresentam as técnicas utilizadas pela ferramenta proposta.

4.4.1 Modelo do usuário

Para a obtenção de informações úteis a partir de um grande conjunto de informações, como a Web, é fundamental ou praticamente inevitável que se conheça alguns dados prévios sobre os interesses dos requisitantes da consulta. Esses dados ajudam na personalização do vasto conhecimento existente na Web, ou seja, na adequação desse conhecimento de acordo com as necessidades do usuário. A modelagem do usuário, estudada inicialmente por Rich, E. (1983), tem a responsabilidade de armazenar e atualizar

esses dados considerados cruciais para cada usuário. Devido à característica de personalização apresentada, a modelagem do usuário torna-se uma peça fundamental dentro de um sistema de filtragem de informações, sendo uma das heurísticas utilizadas mais importantes.

Segundo Kim et al. (2000), atualmente os sistemas de filtragem adotam duas abordagens para a tarefa de modelagem do usuário: modelagem explícita e modelagem implícita. Na modelagem explícita, o perfil do usuário é construído explicitamente, com informações fornecidas por este. A modelagem implícita, por outro lado, explora o retorno do usuário em relação a documentos considerados importantes ou não e, a partir desse retorno, é construído ou adaptado o seu perfil.

A modelagem de usuário abrange o perfil do usuário e os estereótipos utilizados nessa dissertação. No contexto do nosso trabalho, o perfil armazena diversas informações importantes para a ferramenta conhecer os interesses e apoiar o usuário na realização do processo de Engenharia de Domínio. Essas informações são: nome do usuário, objetivo dentro do ambiente, nível de conhecimento a respeito do domínio, categorias de domínio, palavras-chave que o interessam no domínio junto com a sua importância.

As informações são cadastradas manualmente. Portanto, na ferramenta, é utilizada, inicialmente, a abordagem de modelagem explícita. Contudo, posteriormente, são feitas adaptações implícitas no perfil do usuário de acordo com suas intervenções, quando lhe são apresentadas informações. Desse modo, podemos afirmar que, na modelagem de usuário utilizada nesta dissertação, é construída uma abordagem híbrida.

Uma informação importante existente no perfil do usuário trata de qual estereótipo ele pertence. O estereótipo foi definido por Rich (1983) como grupos de usuários que compartilham os mesmos interesses de acordo com um conjunto de critérios. No contexto da dissertação, os usuários de cada grupo que pertencem à mesma categoria no desenvolvimento do domínio e também possuem o mesmo grau de conhecimento em relação ao domínio em questão, provavelmente terão interesses nas mesmas informações e componentes dentro do processo de Engenharia de Domínio.

Na CompAgent, a partir do cadastro das informações do usuário contendo o seu questionário, o perfil desse é montado, sendo automaticamente encaixado em um estereótipo. O estereótipo e o perfil são consultados pela ferramenta durante toda a

utilização do usuário. No contexto da CompAgent são armazenadas algumas informações importantes em cada estereótipo, recuperadas a cada consulta feita à Web pelo usuário. Essas informações são: o conjunto de links já visitados pelos usuários pertencentes ao seu grupo, o grau de importância dado para cada link visitado pelo usuário com seu respectivo peso, que significa o número de vezes que os usuários pertencentes ao mesmo estereótipo já visitaram o link.

O estereótipo é utilizado da seguinte maneira: quando uma consulta é submetida por um usuário, o estereótipo associado ao usuário é encontrado, a partir do qual é feita uma pesquisa na sua base de links. Caso seja localizado um link igual aos retornados pelo serviço de busca, são levados em consideração o grau de importância e o peso deste link na base do estereótipo. Esses valores são utilizados para efeito de cálculo do valor de classificação do link pelo algoritmo de filtragem.

O estereótipo é muito importante no apoio à efetivação da filtragem colaborativa pela ferramenta. Em toda a execução da filtragem colaborativa são consultadas as informações relativas ao usuário e seu respectivo estereótipo. Os links armazenados no estereótipo também são utilizados como uma recomendação implícita pela ferramenta, pois podem indicar, para usuários similares, links que os usuários consideraram importantes e que devem ser levados em consideração nas futuras consultas à Internet.

4.4.2 Filtragem Colaborativa de Informações

A filtragem colaborativa também é chamada de filtragem social (Shardanand e Maes, 1995) ou baseada em ação (Paepecke et al., 1999). Alguns autores, como Delgado e Ishii (1999), consideram que essa técnica deveria ser denominada sistemas de recomendação, pois são arquiteturas que funcionam como recomendadores de informações, indicando um grau de importância da recomendação para cada informação apresentada.

A questão básica dessa técnica consiste em não levar em consideração o conteúdo do documento, ou seja, não se preocupando em entender ou analisar o texto do documento para executar a sua filtragem. Ao invés disso, a filtragem colaborativa se baseia na colaboração entre usuários, ou seja, nas respostas dos usuários a determinadas consultas, ou os documentos que os usuários consideraram importantes. Normalmente, os sistemas de

filtragem preferem levar em consideração, nos seus algoritmos de filtragem, os usuários similares entre si, isto é, que apresentem preferências similares utilizando o sistema.

Uma grande vantagem da filtragem colaborativa é conseguida pelos usuários quando começam a utilizar o sistema, pois sem essa técnica, nas primeiras vezes que estes a utilizassem, esta possuiria apenas o perfil do usuário e não teria informações completas para executar a filtragem eficazmente. Assim, a ferramenta realizaria a filtragem apenas utilizando as palavras-chave e seus respectivos pesos. Com a filtragem colaborativa, desde o início da utilização da ferramenta, este já possuirá alguns links para os documentos importantes, já visitados pelos usuários similares, utilizando-os como referência para as primeiras e também para futuras consultas do usuário.

A filtragem colaborativa armazena informações a respeito das preferências dos usuários, podendo ser utilizadas futuramente em outras consultas pelo sistema de filtragem de informações. Essas preferências são baseadas no julgamento do usuário em relação ao valor de um documento. Os julgamentos podem ser obtidos diretamente dos usuários de três formas: derivados do contexto; com base nas observações das ações; ou de alguma classificação explícita fornecida pelo usuário. Essas formas de julgamento produzem duas abordagens, explicadas a seguir, que são: julgamento implícito e julgamento explícito.

O julgamento explícito acontece quando as classificações são colecionadas diretamente do usuário que utiliza o sistema de filtragem de informações. Cada usuário avalia explicitamente os documentos e coleções apresentadas, ou seja, indica se os documentos são relevantes ou não para sua consulta.

Essa forma de julgamento foi muito utilizada nos primeiros sistemas de RI. A interação entre a consulta do usuário e o julgamento explícito da resposta por parte do usuário originou o termo *relevance feedback*, amplamente difundido na literatura. A técnica era comumente utilizada em sistemas que possuem a técnica de refinamento de consultas. O refinamento de consulta funciona do seguinte modo: após o retorno da consulta, o usuário indica explicitamente o que ele considerou importante; então o sistema de recuperação modifica a consulta com a finalidade de serem retornados mais documentos interessantes (Paepcke et al., 1999).

São utilizadas, em geral, três abordagens diferentes para o julgamento explícito (Paepcke et al., 1999):

- filtragem baseada em retorno do usuário: nesta categoria são considerados os julgamentos realizados por um usuário, ou por um grupo de usuários, e que serão considerados para outros usuários com perfil semelhante.
- filtragem baseada em expressões: esse tipo de filtragem é baseado na construção de uma expressão pelo próprio usuário que irá filtrar as informações. Sistemas de mensagens de correio eletrônico utilizam este tipo de filtro. Exemplo de uma expressão seria o usuário declarar que todo correio eletrônico recebido com um remetente diferente dos existentes na sua lista de endereços deve ser ignorado.
- filtros sintetizadores: são similares aos filtros baseados em expressões. No entanto, utiliza uma abordagem com menos intervenção por parte do usuário. Esse apenas define apenas a tarefa que deseja realizar como por exemplo, filtrar os correios eletrônicos recebidos.

Segundo Kim et al. (2000), embora a implementação do julgamento explícito seja mais fácil, a carga cognitiva elevada, associada às avaliações explícitas de utilização de documentos individuais, serviriam como um desestímulo para a utilização de algumas aplicações. Desse modo, podem ser limitadas as oportunidades para aprendizagem de perfil e, assim sendo, diminui a eficiência do sistema de filtragem em geral. Outro motivo, que pesa contra o julgamento explícito está relacionado ao fato de algumas vezes os usuários não serem capazes de fornecer um julgamento correto, ou seja, possuem dúvida em relação à qualidade do documento que lhe foi retornado. O usuário realiza o julgamento através da inserção de qualquer tipo de informação, preocupando-se apenas em retornar alguma informação para a interface de julgamento, mesmo que a informação seja inútil.

O grande problema encontrado no julgamento explícito, está relacionado à intervenção explícita do usuário quando é necessário o retorno de julgamentos para o sistema. Normalmente, o usuário necessita indicar a qualidade da resposta documento por documento, tornando bastante desgastante e desinteressante para si esta abordagem. Para superar esses problemas, foi criada uma abordagem que procura obter informações implicitamente, ou seja, sem o conhecimento do usuário, procurando deixar o mais

transparente possível a sua utilização no sistema de filtragem. O objetivo principal do julgamento implícito consiste em ter o sistema como observador das atividades do usuário e inferir opiniões dos usuários sobre documentos implicitamente a partir de observações. A divisão dessa categoria é baseada em duas subcategorias (Papecke et al., 1999):

- Julgamento baseado no comportamento de um grupo de usuários: Nessa abordagem, são armazenados históricos de acessos, gerando estatísticas a respeito dos acessos. Os documentos mais acessados são considerados os de maior valor para os usuários e também para os usuários com interesses similares.
- Julgamento isolado do comportamento individual do usuário: A grande vantagem dessa abordagem em comparação com a anterior é que essa não necessita de um histórico de acessos e julgamentos prévios relativos a esses acessos. Nessa abordagem, cada acesso do usuário é considerado individualmente, sendo seu perfil e acessos de usuários similares comparados aos documentos encontrados pelo sistema. As filtragens dos documentos são realizadas com base nas informações do usuário e de seus similares.

Todos os julgamentos dos sistemas de filtragem são feitos a partir dos documentos que o usuário acessa e de observações do seu comportamento em relação aos documentos selecionados para ele. Em Nichols (1997), foi apresentado uma lista de prováveis tipos de comportamentos de usuários que poderiam ser utilizados como fontes para retorno implícito, tais como compra, avaliação, uso repetido, impressão/armazenamento, exclusão, referência, resposta, marcação, verificação/leitura, associação e consulta.

No comportamento de referências pelo usuário, estão representadas todas aquelas instâncias onde um item de informação se relaciona a outro item. Estes relacionamentos podem incluir citações acadêmicas tradicionais como também ligações entre páginas Web. O comportamento de referência é o mais importante, pois através da chamada ao Google, a proposta apresentada nesta dissertação se utiliza indiretamente das referências entre as páginas na Web.

Em um artigo complementar, Ord e Kim (1998) identificaram três categorias gerais de observações do comportamento do usuário: verificações, retenções e referências. Na tabela 4.1, são mostrados os comportamentos divididos por categorias.

Tabela 4.1 – Comportamentos observados para retorno implícito (adaptado de (Ord e Kim, 1998)).

Categoria	Comportamento Observado
Verificação	Seleção Duração Modo de edição Repetição Compra (Objeto)
Retenção	Salvar uma referência ou salvar um objeto • Com ou sem anotação • Com ou sem organização Impressão Remoção
Referência	Objeto para Objeto (redirecionamento, resposta) Parte para Objeto (link de hipertexto,citação) Objeto para Parte (citação)

A categoria mais importante da ferramenta CompAgent implementada é a de verificação e mais precisamente, do comportamento observado na seleção do documento. Segundo Kim et al. (1998), a seleção de objetos para uma verificação mais detalhada pode fornecer a primeira pista sobre os interesses dos usuários.

Durante a interação do usuário com a ferramenta CompAgent, é apresentada, em um browser, uma lista de links para documentos e componentes considerados importantes pelo módulo de filtragem. Então, é capturado implicitamente o julgamento do usuário através do comportamento de seleção dos documentos. Esse comportamento, dentro do nosso contexto, consiste no usuário selecionar (clicar) o link do documento para visualizá-lo, indicando que ele teve interesse em ler/armazenar o documento. Nesse caso, são recolhidas informações específicas do comportamento do usuário que são agregadas à base de filtragem colaborativa.

Em termos de julgamento do documento, é capturado o grau de importância do link e tenta-se inseri-lo na base colaborativa. Caso esse link já exista na base de filtragem colaborativa, seu peso é incrementado dentro da base e, caso não exista, o link é inserido na base e seu peso é considerado como uma unidade. O grau de importância e o peso são incorporados ao link na base de filtragem colaborativa. Portanto, em futuras consultas, se esse link (inserido) estiver contido em algum conjunto para ser filtrado, o grau de importância e o peso a serem utilizados são recuperados a partir de informações relativas

aos links armazenados na base colaborativa. Uma importância do estereótipo é verificada nessa característica da CompAgent, pois os retornos positivos relativos a um ou vários documentos são armazenados no estereótipo do usuário a serem utilizados colaborativamente por outros usuários.

O exposto acima demonstra que a CompAgent aborda a categoria de julgamento isolado do comportamento individual do usuário, pois o usuário é considerado individualmente, com o seu próprio perfil, e bem como os interesses de usuários similares. Além disso, o que o usuário julgar importante também vai servir para os próximos usuários similares.

4.4.3 Recomendação Colaborativa

A Recomendação Colaborativa é uma técnica baseada na colaboração de links entre usuários. Essa colaboração cria uma base de links, que são inseridos por um usuário que possui um profundo conhecimento a respeito do domínio em questão. Esses links são compartilhados com outros usuários que não detêm esse alto grau de especialização. A base de colaboração deve ajudar o usuário, principalmente no início, a executar mais facilmente a tarefa de busca de conhecimento a respeito do domínio.

A recomendação colaborativa tem como objetivo fornecer um suporte a buscas dos usuários na Web, de forma que eles tenham um guia inicial do que deve ser importante. Isso pode evitar que naveguem aleatoriamente pela Web, procurando informações para o processo de análise de domínio ou procurando componentes para suas aplicações. Antes de ser utilizada a recomendação colaborativa, é criado um *bookmark* para armazenar links importantes a respeito do domínio, que poderá ser consultado pelos diversos usuários do domínio, e sendo atualizado explicitamente ou implicitamente pelo seu responsável.

Nesta técnica, existe uma figura principal que funciona como se fosse um superusuário da ferramenta. Normalmente, o especialista do domínio é o responsável por esse papel, pois é o profissional do projeto que possui o maior conhecimento a respeito do domínio que está sendo construído. Ele é responsável pela orientação inicial dos outros participantes do projeto para busca de informações importantes.

A recomendação colaborativa pode ser feita de forma híbrida, ou seja, explicitamente ou implicitamente:

- Explicitamente: O usuário cadastra, em qualquer etapa do processo de Engenharia de Domínio ou de Aplicação, o link que deve ser seguido pelos outros usuários (a qualquer momento significa que pode ser no início, de preferência, ou durante o processo de Engenharia de domínio). Esse cadastro não é composto somente por links, mas também por componentes que podem ser úteis para a Engenharia de Aplicação. O cadastro dos componentes ou links se constitui de informações como: nome, descrição, tipo do link (componentes ou páginas HTML) e classificação do item, que é um grau de importância dado pelo engenheiro de domínio que está cadastrando a informação.
- Implicitamente: Ao submeter uma consulta o usuário, no papel de engenheiro de domínio, recebe uma lista que contém links para componentes ou informações. Cada link selecionado funciona como um retorno positivo em relação ao link escolhido. Esse, por sua vez, será adicionado à base de recomendação do domínio, com a classificação do item sendo fornecida a partir do grau de importância dado pelo algoritmo de filtragem. Caso seja um componente, este será adicionado à categoria de componentes e o mesmo procedimento será usado para os links de páginas HTML. Desta forma, a ferramenta vai alterando dinamicamente a base colaborativa. Caso seja um usuário comum, o peso do link é atualizado, ou seja, acrescido de uma unidade, ressaltando que ele não tem direito de inserir o link na base.

Com a atualização realizada implicitamente, a ferramenta é capaz de adaptar dinamicamente a base colaborativa, procurando ajustar as necessidades dos usuários com o mínimo de sobrecarga possível ao seu responsável.

4.4.4 Mediadores

Uma solução para a integração das informações distribuídas e heterogêneas, amplamente descrita na literatura, é a utilização de uma arquitetura de mediação. Essa arquitetura é composta por mediadores, que, de acordo com Wiederhold (1992), são módulos que incorporam camadas de serviços de mediação, conectando bases de dados distribuídas e heterogêneas (produtores) a sistemas de informação (consumidores).

A camada de mediação possui uma função fundamental na arquitetura de mediadores, pois permite a visão integrada das bases de dados, tornando o acesso às

informações transparente. A CompAgent trabalha como cliente de uma implementação dessa categoria de arquitetura de bancos de dados heterogêneos. A CompAgent envia consultas para a arquitetura e recupera os componentes a serem filtrados para, posteriormente, mostrá-los ao usuário.

Baseada na arquitetura de mediadores criada por Wiederhold (1992) foi construída a arquitetura HIMPAR (Pires, 1997). O conceito de mediadores, criado inicialmente por Wiederhold, evoluiu e foi estendido para um novo conceito, o de mediação inteligente, cujo principal objetivo é a aquisição de conhecimento sobre determinado domínio de aplicação e a posterior manipulação desse conhecimento para a recuperação de informações.

Essa abordagem de mediação inteligente apresenta diversas vantagens, como uma definição mais precisa dos modelos utilizados por cada domínio, melhora na recuperação de informações, pois são analisadas apenas as relacionadas àquele domínio. Para a adequação a esse novo conceito, a arquitetura HIMPAR sofreu uma extensão descrita em (BRAGA et al., 1999). A arquitetura, após ser estendida, foi incorporada ao ambiente Odyssey, sendo denominada de *Odyssey Mediation Layer* (OML). Na OML é utilizado o conceito básico de mediação inteligente, considerado por alguns autores como ontologias de domínio. Essa divisão por domínio facilita a visão e a recuperação dos componentes específicos por domínio, devido a cada componente ser representado por termos da ontologia do domínio a qual ele pertence.

Uma definição dada por Wiederhold (1997) estabelece que, no contexto da mediação, ontologia pode ser considerada como um vocabulário de termos e a especificação dos relacionamentos entre eles. Para cada termo, deve ser criada uma definição que consiste de uma descrição informal do termo. Deve ser feita também uma especificação dos relacionamentos entre os termos do domínio ou de outros domínios, formando uma rede semântica. Na OML são utilizadas essas definições e relacionamentos para a especificação da ontologia sobre um determinado domínio, sendo que cada mediador está associado a uma ontologia do domínio.

A HIMPAR e a OML evoluíram para uma arquitetura em comum, a arquitetura implementada pelo sistema ComPublish (Souza, 2002), devido a novas necessidades relacionadas à integração de dados estruturados e semi-estruturados como XML e também relacionadas à busca, integração e publicação de componentes na Internet. A partir dessas

necessidades, concebeu-se essa nova arquitetura implementada em (Souza, 2002) e utilizada diretamente pela ferramenta CompAgent. A ferramenta ComPublish é apresentada na próxima subseção.

4.4.5.1 Sistema de recuperação de componentes

A ferramenta proposta nesta dissertação funciona como cliente de uma arquitetura de recuperação de componentes, sendo que, para esta implementação, ela é cliente da ComPublish. O sistema ComPublish, mostrado na figura 4.3, visa a publicação de componentes de software e seus artefatos relacionados, tais como modelos, diagramas, código-fonte e outros documentos, na Internet de forma que outros engenheiros de domínio, analistas de domínios, engenheiros de aplicação compartilhem os componentes armazenados na sua base. Além da publicação de componentes, o ComPublish fornece uma visão uniforme dos componentes que pertencem ao mesmo domínio de aplicação. Na camada de mediação, são encontrados metadados que descrevem os repositórios dos componentes, representando informações sobre esses, como: o domínio, a semântica deles, a arquitetura e as interfaces.

A camada de mediação tem como objetivo funcionar como um componente acoplado ao ambiente Odyssey, mas funcionando também de forma independente. Ela utiliza as mesmas tecnologias da OML, explicadas anteriormente, tais como mediadores por domínios de aplicação e suas respectivas ontologias, compondo as ontologias de domínio. Essa camada, um elemento do ComPublish, promove a integração de domínio para traduzir requisições de componentes através das ontologias (Oliveira, 2002).

Portanto, os principais serviços do ComPublish (Souza et al., 2001) são: descrever componentes baseados na informação de domínio; integrar a descrição com outras semânticas de outros componentes publicados do mesmo domínio; fornecer mecanismos de busca sobre os componentes publicados; e armazenar e recuperar componentes de software.

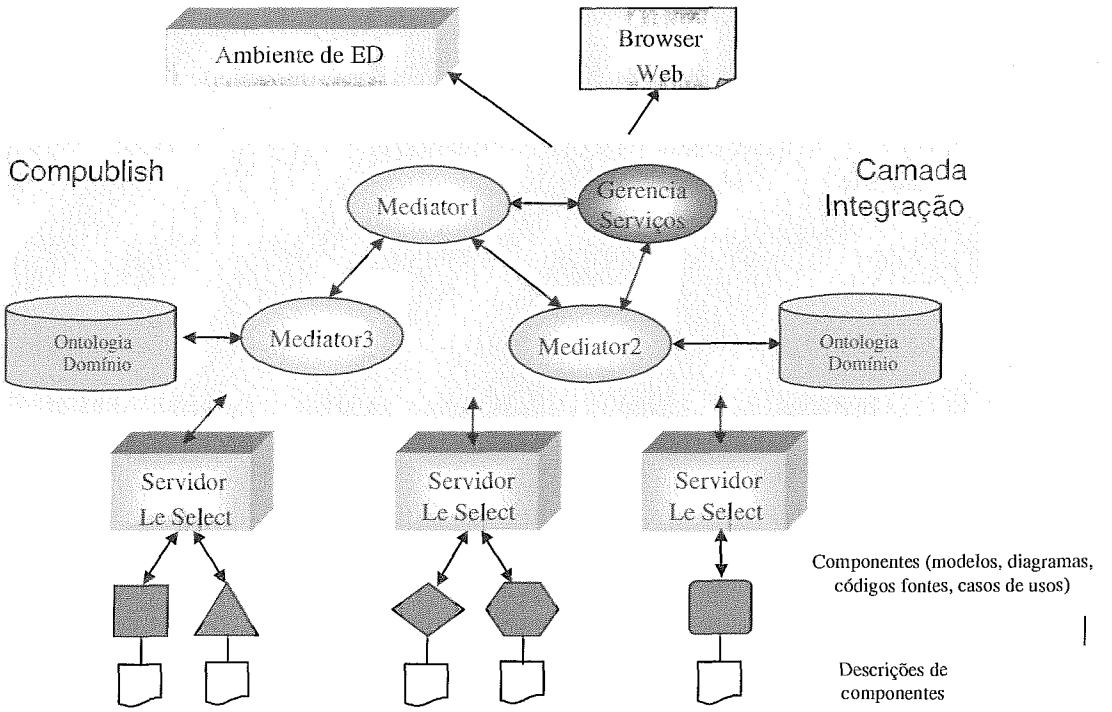


Figura 4.3 – Visão geral do Sistema Compublish e Le Select.

Para publicar e acessar um componente em um site remoto através do ComPublish, o proprietário do componente deve instalar o sistema LeSelect (LeSelect, 2002). LeSelect fornece facilidades para a publicação de dados e programas na Web, além de suportar a publicação de metadados associados aos dados publicados. O ComPublish associa documentos XML para representar cada descrição de componentes de software. Assim sendo, para todos os componentes a serem publicados, o publicador tem que criar um documento XML descritivo para associá-lo ao documento.

O ComPublish é chamado pela CompAgent com a finalidade de fornecer um conjunto de componentes como resposta, de acordo com as informações enviadas pela CompAgent. Essas informações consistem da palavra-chave inserida pelo usuário do domínio sendo pesquisado e das características do componente que devem possuir a palavra-chave. A partir da definição de qual domínio será consultado, o ComPublish define qual ontologia e mediador serão utilizados na consulta.

Os componentes que atendem aos requisitos da consulta são então retornados pelo sistema ComPublish, de modo a sofrer um processo de filtragem e classificação semelhante

ao utilizado para o retorno de links do serviço de busca Google. A descrição dos componentes, retornados pelo ComPublish, vem no formato XML, sendo necessária à realização de uma análise dos documentos XML para uma posterior etapa de filtragem e classificação. É importante frisar que esses componentes retornados podem ser tanto publicados localmente, ou seja, armazenados nas bases de dados gerenciados pelo ComPublish, como publicados remotamente em algum servidor LeSelect.

Os componentes são apresentados, em um browser, ao usuário. Caso ele escolha algum deles, o usuário o traz para a sua máquina, clicando no link relacionado a este componente. Desse modo, ele é adicionado na base de filtragem colaborativa. Na seção 5.2.6 do capítulo 5, é detalhada a implementação do ComPublish.

4.5 Conclusões

Na seção 4.2 foram descritos diversos requisitos necessários para atender os usuários de ambiente de suporte aos processos de engenharia de domínio e de aplicação. Esses requisitos estão relacionados a busca e recuperação por informações e componentes na Web, ou por componentes publicados, por outros desenvolvedores, localmente ou remotamente. Os requisitos de respostas livres de contexto, preferências dos usuários, semântica de componentes, busca orientada, colaboração entre usuários foram incorporados na CompAgent, através das diversas técnicas apresentadas na seção 4.4.

A inserção de um contexto na consulta do usuário é obtida através de todas as técnicas utilizadas pela CompAgent. Dessas, a mais importante, em termos internos à arquitetura, é a modelagem de usuário, pois cada usuário possui um perfil diferente para cada domínio. Através do seu perfil, é possível moldar o domínio de acordo com seus interesses, definindo um contexto específico para cada domínio.

Um segundo objetivo atendido na CompAgent está relacionado ao armazenamento das preferências dos usuários. Estas informações são persistidas em um banco de dados para posterior recuperação, utilização e adaptação pela ferramenta. Essa característica permite um avanço em relação a uma busca aleatória realizada em um serviço de busca, procurando prover uma personalização à busca de informações e também para a busca dos componentes desejados em uma biblioteca de componentes, que é disponibilizada pelo ComPublish.

A colaboração entre usuários é um outro requisito importante. A ferramenta procura obter a maior precisão possível para cada usuário e, para atingir tal objetivo, foram criados dois níveis de colaboração entre estes: o primeiro nível é tratado de usuário para usuário, onde este, ao selecionar um link, indica este para outro usuário pertencente ao seu mesmo grupo interesses. Esta restrição ocorre, pois este link, provavelmente, somente será importante para a colaboração entre os usuários de um mesmo grupo. O segundo nível trata da colaboração entre os usuários que detem o maior conhecimento naquele determinado domínio, i.e, engenheiros de domínio e especialistas, e o usuário "comum"[10] que trabalhará na modelagem e na implementação do domínio. Os primeiros indicarão informações (páginas HTML ou componentes) considerados essenciais para o segundo compreender o domínio, ou realizar efetivamente a engenheira de aplicação.

A CompAgent resolve o problema da busca por componentes através da utilização de um sistema de busca e recuperação de componentes, o ComPublish, que também é responsável pela publicação da interface do componentes. Como o ComPublish tem como uma das suas funções a publicação e recuperação de componentes para apoiar o processo de engenharia de aplicação e de domínio, este armazena a interface do componente com informações específicas para atender essas funções. Essas informações, entre outras, incluem o objetivo do componente, fase em que ele é utilizado e, principalmente, o domínio, ou os domínios em que ele é mais bem aplicado. No processamento do ComPublish, ela seleciona apenas os componentes semanticamente adequados à consulta e relacionados ao domínio indicado. Esses componentes são ainda filtrados dentro da CompAgent, nesse caso tentando obter um contexto para os componentes e também atendendo às preferências dos usuários.

As outras ferramentas de filtragem de informações estudadas apresentam certas características, tais como armazenar as preferências do usuário, tentar inserir um contexto para as consultas do usuário, implementar colaboração entre usuários do mesmo grupo de interesses ou de qualquer outro grupo de interesses. A ferramenta CompAgent apresentada possui tais características, mas sendo frisado que ela foi projetada com um objetivo específico de apoiar o usuário na busca e recuperação de informações em um ambiente de

[10] Usuário padrão utilizando o ambiente de reutilização de software, não possuindo responsabilidades de gerenciamento do domínio sendo modelado.

suporte ao processo de ED/EA. Portanto, apresenta uma abordagem diferente, associando preferências do usuário e as colaborações entre eles por domínio.

Pode-se concluir que as outras ferramentas são muito abrangentes, ou seja, tentam atender a uma vasta gama de usuários com interesses diferentes nas suas buscas por informações. Essa abrangência pode não ser tão efetivo para um usuário que necessite informações mais específicas. Outras atendem problemas de outros domínios do conhecimento, como busca por artigos técnicos de computação (Bollacker et al., 1998). Já a CompAgent é específica para a resolução do problema de apoio ao processo de ED/EA, contudo, tentando ser o mais eficiente possível. Essa eficiência consiste em mostrar o conjunto de links mais otimizado possível, ou seja, numa quantidade que satisfaça o usuário de acordo com suas necessidades.

No próximo capítulo será apresentado um detalhamento da ferramenta, além de tratar das tecnologias de engenharia de software, banco de dados e objetos distribuídos que foram utilizadas na implementação da ferramenta. Os detalhes técnicos serão abordados com discussões sobre determinadas decisões de implementação que foram tomadas.

Capítulo 5 – Implementação da Ferramenta CompAgent

5.1 Introdução

Conforme visto anteriormente, a Engenharia de Domínio é uma subárea da Engenharia de Software que envolve um conjunto de atividades complexas que devem sofrer algum gerenciamento para serem realizadas com sucesso. Para facilitar o gerenciamento, foram criados ambientes de software que suportam um determinado processo definido para a Engenharia de Domínio. Esse processo tem a função de controlar quais etapas devem ser obedecidas para a realização da engenharia de qualquer domínio do conhecimento.

O ambiente de desenvolvimento de software Odyssey (Werner et al., 2000), desenvolvido na COPPE/UFRJ, disponibiliza recursos de automatização necessária para que as etapas de um processo de engenharia de domínio sejam apoiadas de maneira adequada, auxiliando o engenheiro de domínio na especificação dos componentes do domínio e sua posterior reutilização. Esse processo foi definido por Braga (2000) e é conhecido como Odysssey-DE. O Odyssey também provê suporte ao processo de engenharia de aplicações, denominado Odyssey-EA, definido por Miller (2001). Segundo Braga (2000), sem o auxílio desta ferramenta, a execução das atividades de ED e de EA implicaria em uma grande quantidade de trabalho manual, seja na criação e manutenção dos relacionamentos entre os componentes especificados nas diversas etapas do processo, seja no auxílio ao desenvolvimento de novas aplicações no domínio.

O Odyssey funciona como uma ferramenta para suportar a reutilização no desenvolvimento de software. Ela prove os requisitos necessários para o desenvolvimento *para* reutilização através do suporte às etapas do Odyssey-DE e ao armazenamento dos produtos resultantes do processo de ED. O ambiente possui ainda suporte ao desenvolvimento *com* reutilização, através do apoio às atividades do Odyssey-EA, e também da busca e recuperação dos componentes reutilizáveis (Braga , 2000) e (Werner et al., 2000).

A ferramenta CompAgent foi implementada no contexto do ambiente Odyssey para ser utilizada pelos seus usuários durante as diversas etapas dos processos de engenharia de

domínio e de aplicação. A principal função da ferramenta é oferecer suporte aos usuários do ambiente para a busca e recuperação de informações orientadas a um domínio, com ênfase na atividade de análise de domínio e engenharia de aplicação.

Muitas das características de implementação da ferramenta foram adequadas para atender aos detalhes da implementação do ambiente, além de reutilizar alguns conceitos já implementados em outras ferramentas internas, como a Odyssey_Search, descrito na seção 4.4.1, e a ferramenta de apoio ao processo de software (Murta, 2002). Os principais conceitos associados dizem respeito à modelagem do usuário, como estereótipo, perfil e papel do usuário no ambiente, estendendo-os para atender algumas funcionalidades, explicadas na seção 5.2, necessárias a CompAgent.

Nas próximas seções descreveremos o projeto da ferramenta CompAgent, apresentando os diagramas de classes, detalhes sobre as decisões arquiteturais dos componentes envolvidos, as interfaces com o usuário e detalhes sobre heurísticas dos algoritmos. Na última seção, é mostrada uma avaliação da utilização por um usuário da ferramenta, mostrando passo-a-passo o seu funcionamento.

5.2 Técnicas utilizadas na implementação da CompAgent

A ferramenta CompAgent foi implementada na linguagem Java (SUN, 2002), utilizando a versão 1.3.1. A escolha da linguagem foi principalmente pelo fato do ambiente alvo onde ela seria implementada, no caso o Odyssey, ter sido implementado em Java. Para a persistência dos objetos da ferramenta foram utilizados os serviços de um gerente de objetos, o Sistema de Gerenciamento de Objetos GOA++ (Mauro et al., 1997), também implementado na COPPE/UFRJ.

O GOA++ possui diversos papéis no suporte ao ambiente Odyssey. Ele serve como um repositório para o armazenamento e recuperação de componentes relacionados a um domínio (Mattoso et al., 2000), sendo esta sua função principal para a dissertação.

5.2.1 Modelo do Usuário

A modelagem do usuário é extremamente importante para a ferramenta, sendo fundamental para o sistema conhecer os interesses do usuário, de modo a ser capaz de fornecer informações mais customizadas. Na figura 5.1, apresentamos o diagrama de classes na notação UML (Unified Modelling Language) (Fowler e Scott, 2002).

A classe Usuário possui apenas os atributos de login e senha, que fornecem acesso ao ambiente para o usuário, conforme mostra a figura 5.2. Cada usuário do Odyssey possui vários usuários divididos por domínio, ou seja, para cada domínio a ser modelado possuirá um usuário diferente, com perfil diferente. Portanto, ele está relacionado à classe UsuárioDom, que representa o usuário por domínio e nessa classe estão armazenadas informações tais como, se o papel dele no domínio é de engenheiro do domínio e o conjunto de palavras-chave de interesse do usuário em um determinado domínio.

Pelo modelo, pode ser interpretado que um Usuário do domínio está relacionado a apenas um estereótipo. O estereótipo escolhido para um usuário consiste na classe que ele possuir mais afinidade, sendo esta definida por dois atributos da classe Estereotipo: edGrauFamiliaridade e edObjetivos. O atributo edGrauFamiliaridade descreve o grau de conhecimento sobre o domínio de um dado usuário. O atributo edObjetivos indica o objetivo que o usuário pretende alcançar com a navegação nos modelos do domínio.

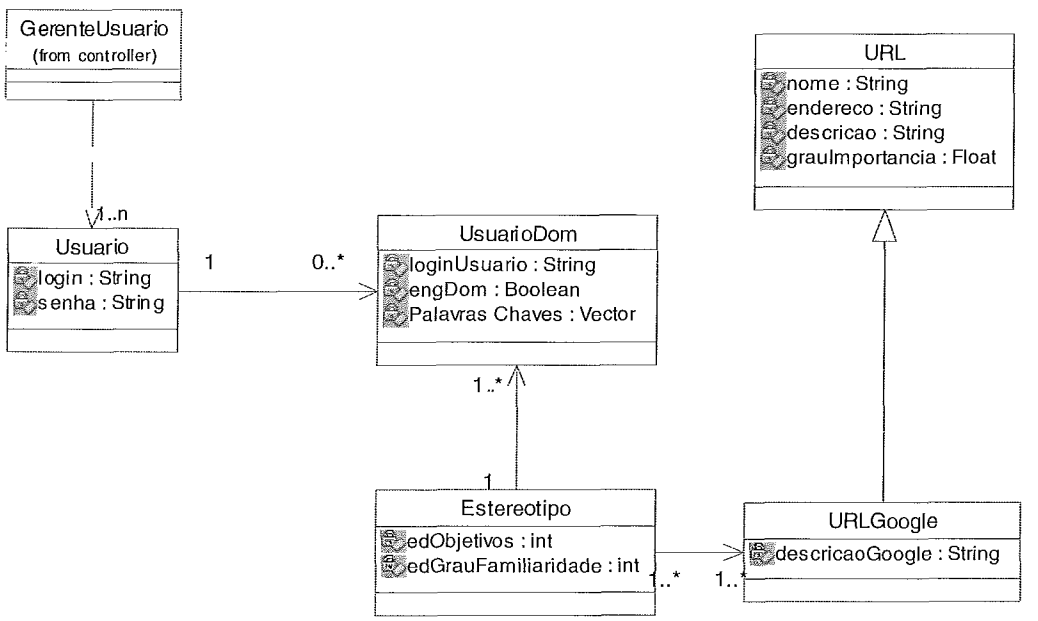


Figura 5.1 – Diagrama de classes da modelagem do usuário

A definição do estereótipo do usuário é indicada em função do acesso a um domínio específico no Odyssey (figura 5.2). Se for a primeira vez que o usuário acessa esse domínio, ele deverá responder a um questionário (Figura 5.3) e, a partir das informações cadastradas no questionário, será construído o seu perfil e definido o estereótipo a que ele pertence. Deve ser frisado que o questionário é a única forma explícita de informação que o usuário fornece ao sistema. A partir do questionário, não existe mais interação explícita com a ferramenta em relação às suas preferências.

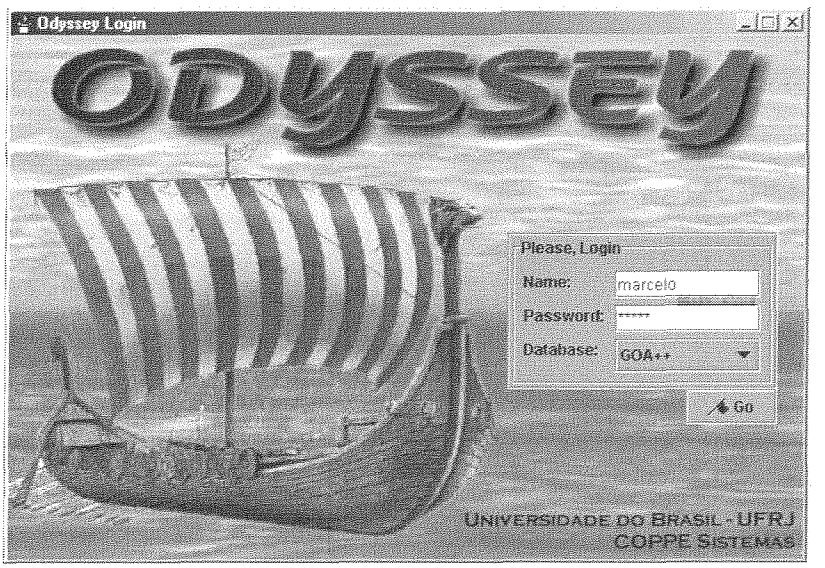


Figura 5.2 – Tela de Login do Odyssey

O questionário possui duas abas na sua interface. A primeira aba (figura 5.3), possui os seguintes campos:

- Objetivos: Indica o que o usuário pretende fazer durante a navegação e consultas a Web dentro do ambiente. Existem duas possibilidades de escolha:

  a) Entendimento do domínio: O usuário objetiva apenas obter um conhecimento do domínio em termos de seus conceitos e funcionalidades, mas não objetiva desenvolver aplicações no domínio.

  b) Desenvolvimento de Aplicação: O usuário deseja desenvolver aplicações no domínio e, portanto, necessita de um detalhamento de todos os modelos desse domínio.

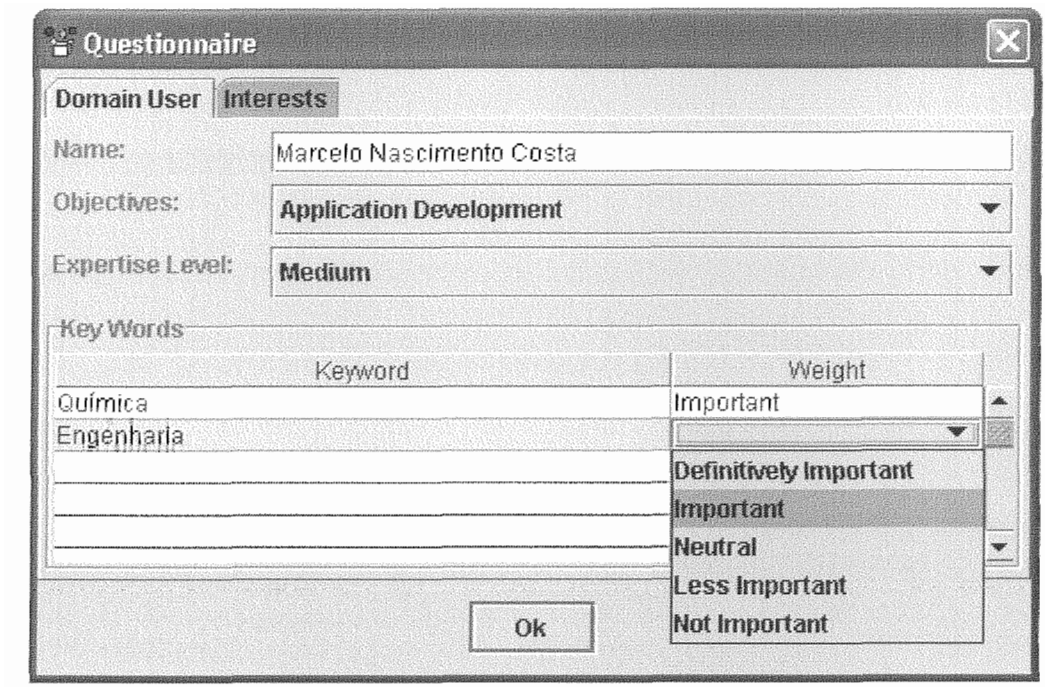


Figura 5.3 – Primeira aba do Questionário

- Nível de Conhecimento: Indica o nível de conhecimento do usuário em relação ao domínio. São divididos em três níveis:

a) Baixo: O conhecimento do domínio é mínimo, sendo considerado que o usuário nunca estudou o domínio.

b) Médio: É considerado que o usuário tem conhecimento básico sobre o domínio. O usuário conhece o domínio de modo superficial, podendo ter participado apenas da construção de uma simples aplicação relacionada ao domínio (Braga,2000).

c) Alto: O usuário é um especialista no domínio e não necessita de muitas informações relacionadas ao domínio. Normalmente, um especialista do domínio possui esse grau de especialidade.

- Palavras-Chave: O usuário insere as palavras-chave que ele considera importante no domínio. Para cada palavra-chave inserida, o usuário indica o grau de importância que a palavra possui para ele neste domínio. Esse grau de importância, ou o peso da palavra, possui a seguinte escala: Definitivamente Importante, Importante, Neutro, Pouco Importante e Não-Importante. Essa escala é convertida para valores discretos, correspondendo respectivamente a 1, 0.8, 0.6, 0.4, 0.2, de acordo com a sua

ordem de importância. As palavras-chave e seus respectivos pesos são um dos fatores considerados pelo algoritmo de filtragem, para efetuar a sua tarefa.

Na segunda aba, mostrada na figura 5.4, os usuários inserem as seguintes informações sobre o seu perfil:

- Subáreas: indica os contextos e sub-domínios que interessam ao usuário.
- Aplicação: aplicações já desenvolvidas no domínio e que o usuário tem interesse em desenvolver alguma similar àquela.

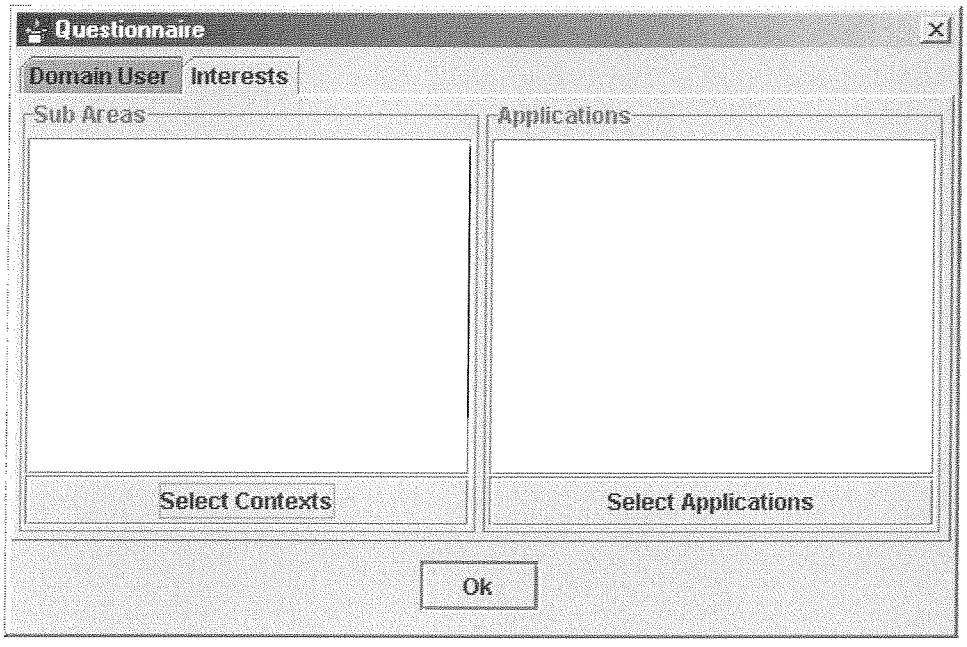


Figura 5.4 - Segunda aba do Questionário 1

Após o preenchimento do questionário pelo usuário, o ambiente procura associá-lo a um estereótipo já existente. Essa associação é feita através da procura por um estereótipo, na base de objetos de usuários de domínio, em que outros usuários possuam o mesmo objetivo e o mesmo grau de familiaridade com o domínio. Caso este não exista, é criado um novo estereótipo para o usuário. Um estereótipo possui um conjunto de perfis de usuários relacionados. Além de agrupar os usuários, o estereótipo, como foi mostrado no diagrama da Figura 5.1, possui um conjunto de URL relacionadas. Essas URLs são armazenadas de acordo com as navegações de todos os usuários pertencentes ao mesmo estereótipo, formando uma base de colaboração entre eles. Cada link selecionado pelo usuário, a partir

de um conjunto de links retornados pela ferramenta, será armazenado na lista de URLs do estereótipo ao qual ele pertence.

A base colaborativa é adaptativa, pois a cada utilização por parte dos usuários, ela sofre inserções de links que podem ser considerados importantes para outros usuários em futuras utilizações do sistema.

5.2.2 Recomendação Colaborativa

A recomendação colaborativa consiste em um módulo que apóia o engenheiro de domínio e é direcionado aos usuários "comuns" do processo de engenharia de determinado domínio. O engenheiro de domínio cadastra os links apontando para componentes ou para páginas Web que podem ser mais importantes para o usuário para a obtenção de conhecimento e compreensão do domínio analisado.

A interface de construção possui uma tela de cadastro de links (Figura 5.5). Deve ser ressaltado que somente o engenheiro de domínio possui acesso a essa interface, pois é considerado que, nesse papel, o usuário possui o maior conhecimento sobre o domínio e tem condições de indicar fontes de informações para outros usuários "comuns".

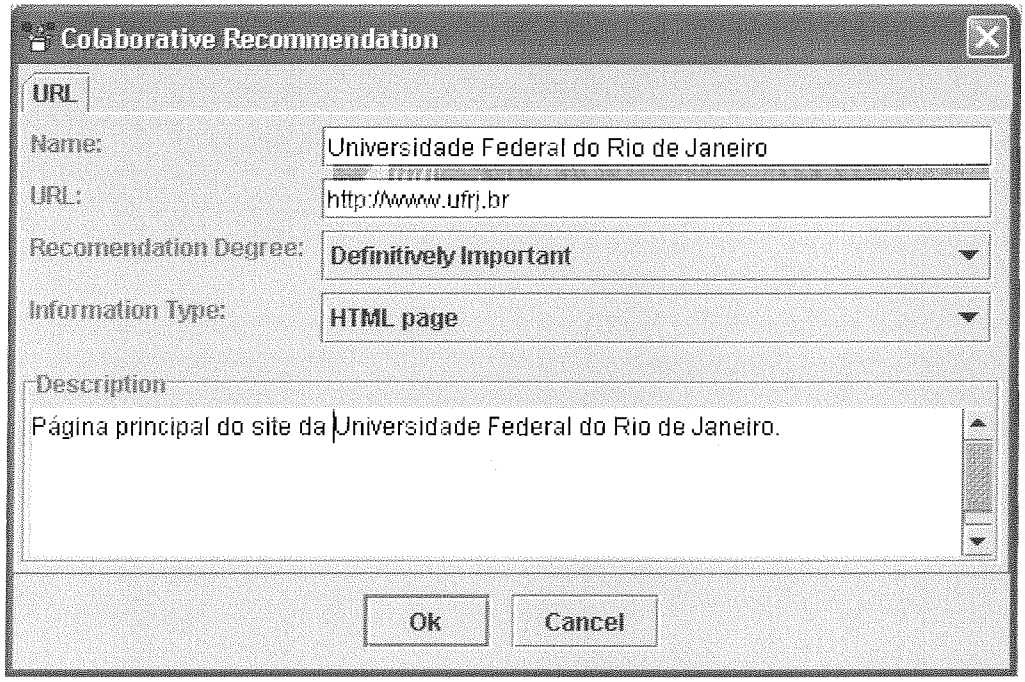


Figura 5.5 - Cadastro de links de recomendação colaborativa

Como mostrado na figura 5.5, são cadastradas as seguintes informações:

- Nome: representa o título do componente ou da página a ser cadastrado.
- URL: Representa o endereço na Web ou em um servidor LeSelect, onde pode ser realizado o download do componente, ou onde a página HTML pode ser navegada pelos usuários.
- Grau de Recomendação: Define o quão importante é esse link para os usuários do domínio. O grau de importância possui a mesma escala definida para palavras-chave no perfil do usuário, consistindo dos termos: Definitivamente Importante, Importante, Neutro, Pouco Importante e Não-Importante. Essas informações são mapeadas com a mesma escala de valores dos pesos das palavras-chave.
- Tipo de Informação: Indica qual modalidade de informação está sendo fornecida ao usuário. A modalidade pode ser uma página HTML ou um componente.

Existem algumas regras associadas ao cadastro das recomendações. Quando o link é cadastrado, o atributo peso recebe um valor unitário e um grau de recomendação atribuído pelo engenheiro do domínio que o está cadastrando. Quando outro engenheiro do domínio cadastra um link já existente na base, é apresentada uma mensagem com o login do engenheiro de domínio que o cadastrou anteriormente, e o grau de recomendação cadastrado anteriormente. Caso este último deseje, essa recomendação é atualizada pelo novo grau escolhido durante o cadastro mas, de qualquer modo, o peso do link é aumentado em uma unidade.

O usuário "comum" pode consultar esses links recomendados. Ele os obtém através de uma listagem mostrada no formato HTML, contendo todos os links recomendados para aquele domínio. O usuário pode escolher as seguintes opções: listar todo o *bookmark* de páginas HTML; listar todo o bookmark de componentes; listar o *bookmark* inteiro. A partir desta escolha, o usuário visualiza a página HTML com os links de acordo com a opção escolhida. Para cada link selecionado a ser visitado, entra em ação o agente de feedback, explicado na seção 5.1.5, que captura a escolha do usuário, atualizando a base de recomendação colaborativa. Essa atualização consiste em aumentar, na base de links, o peso deste link selecionado pelo usuário em uma unidade. Esses pesos são considerados

durante o algoritmo de filtragem, fazendo parte de uma das heurísticas utilizadas pela ferramenta.

5.2.3 Busca Local de Componentes

Conforme visto no capítulo anterior, o sistema de busca local de componentes tem o objetivo de armazenar componentes localmente e remotamente, e também, posteriormente, permite buscá-los e recuperá-los. O ComPublish (Souza, 2002), sistema de busca de componentes utilizado na arquitetura, foi implementado na linguagem C++ e, normalmente, está executando remotamente, ou seja, não está executando na mesma máquina que o Odyssey. O Odyssey e a CompAgent estão implementados na linguagem Java, tornando-se necessária uma tecnologia que possa resolver essa impedância entre as linguagens de implementação e, também, resolver a questão da distribuição de objetos, permitindo a integração entre as arquiteturas.

Dentre as tecnologias disponíveis, foi escolhido o CORBA (OMG, 2002), devido a sua transparência no acesso a servidores de aplicação remotos. Desta forma, a CompAgent não precisa conhecer antecipadamente o endereço da máquina onde está localizado o ComPublish. A integração funciona da seguinte forma (figura 5.6): um agente, componente da arquitetura CORBA, fica executando em uma máquina na rede, recebendo as consultas e disparando-as para onde estiver executando o servidor ComPublish. Esse agente recebe a resposta e devolve-a para a máquina, onde está executando a CompAgent, que enviou a consulta.

Para poder realizar a integração com o ComPublish, conforme mostrado na figura 5.6, foram realizados alguns passos necessários para atender os requisitos da especificação CORBA:

1. Foi definida uma implementação da especificação CORBA comum a ser utilizada entre os sistemas. Essa implementação é conhecida como ORB (Object Remote Broker), que é o elemento da especificação que gerencia o envio e recebimento de requisições entre as arquiteturas. O ORB utilizado foi a VisiBroker (Inprise, 2002), pois realiza a integração entre as linguagens Java e C++ e está disponível gratuitamente na Internet.

2. Foi fornecida uma IDL (Interface Definition Language), linguagem de definição utilizada pela especificação CORBA, que possui as estruturas de dados e a interface publicada dos métodos existentes no servidor ComPublish e que serão chamados pela CompAgent.
3. Essa IDL é compilada, com o compilador idl2java incluído no Visibroker, gerando um pacote de classes que foi adicionada ao Odyssey. Esse pacote é conhecido como *stub*, que contém o conjunto de classes que representa a interface do servidor no lado cliente. O Skeleton, conjunto de classes que publica a interface do servidor para outros sistemas, já foi gerado anteriormente pela ComPublish (Figura 5.6).
4. Um agente ORB deve ser inicializado na rede onde estão os clientes e o servidor. Importante frisar que, devido às características do ORB utilizado, não é necessário nenhum dos três componentes estarem na mesma máquina (Figura 5.6).
5. Para acessar o servidor CORBA, é instanciado um objeto da classe gerada como *stub*. A partir deste objeto, a CompAgent tem acesso a todos os métodos existentes no servidor.

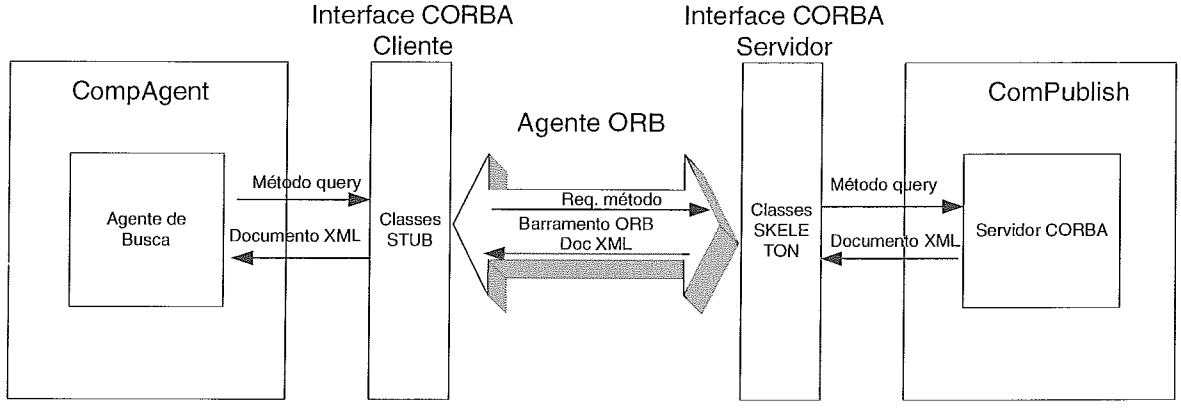


Figura 5.6 - Integração CompAgent e ComPublish

O método da interface que é chamado, internamente, na CompAgent, é o seguinte: *executeMetadataQuery(in string query, in string mediatorName)*.

São passados dois parâmetros: o primeiro *in* indica a consulta que deve ser passada, incluindo os atributos que devem conter a palavra-chave em cada componente; o segundo, o *MediatorName*, indica o nome do mediador em que deve ser feita a consulta. Como cada mediador representa um domínio, é passado, ao servidor, o nome do domínio em que devem ser procurados os componentes.

O ComPublish processa internamente a consulta que, em poucas palavras, significa que ele procura componentes nos repositórios locais e remotos que estão ligados ao domínio requisitados na consulta. A consulta é realizada sobre os metadados na busca de características dos componentes especificadas pelo usuário (Souza, 2002).

Os componentes que atenderam a consulta são retornados para a CompAgent no formato de um documento XML, conforme mostrado na figura 5.7, que então é passado por um processo de análise, explicado na seção 5.2.6. Como resultado da análise, é gerado um conjunto de objetos Java, que passa pelo algoritmo de filtragem antes de ser retornado para a seleção do usuário.

```
<?xml version="1.0" standalone="yes"?>
  <Components>

    <Component>
      <Datum>
      <Domain>Radio</Domain>
      <Name>Diagrama de Classes de Enlace</Name>
      <Date>20/10/1998</Date>
      <Version>1</Version>
      <Author>Equipe Coppetec</Author>
      <Description>Classes de modelagem do conceito de enlace de transmissao via
radio</Description>
      <Url>ftp://LocalHost:3021/Telecommunication/Radio/Documentos/Diagrama de
Classes - Enlace.mdl</Url>
      <Contact>pinheiro@cos.ufrj.br</Contact>
      <Type>Diagrama</Type>
      <FileType>mdl</FileType>
      <AssociatedApplications>Rational Rose</AssociatedApplications>
      </Datum>
    </Component>

  </Components>
```

Figura 5.7 – Exemplo de um documento XML

5.2.4 Agente de Busca

O agente de busca é o módulo responsável pela interface da ferramenta CompAgent com o mundo externo. O agente de busca tem como responsabilidade principal receber a consulta do usuário, depois realizar a interação com o serviço de busca Google, detalhado no capítulo 3, e também com a arquitetura de mediadores do Compublish, explicado na seção 5.2.3. Além de se conectar com o Google, ele executa a extração das informações das páginas HTML e dos documentos XML para serem utilizadas na filtragem de informações.

O agente de busca funciona da seguinte maneira:

1. O usuário possui uma interface para interagir com o agente. Nessa interface (figura 5.8), ele insere a(s) principal(is) característica(s) que deseja no componente ou no link a ser retornado.

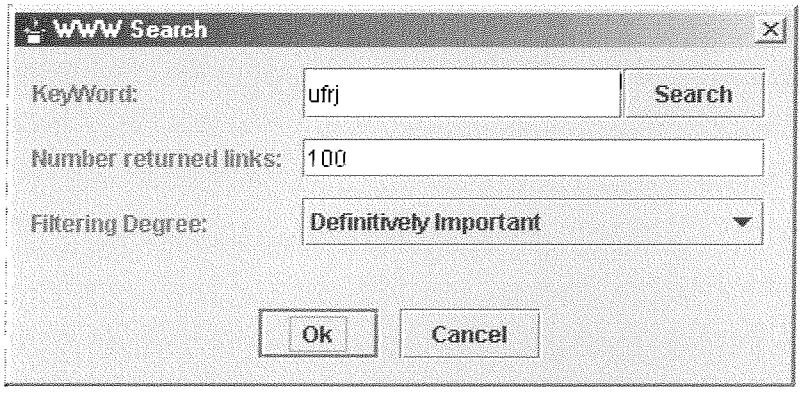


Figura 5.8 - Janela de Busca do usuário

2. No segundo campo da janela, o usuário escolhe a quantidade de links que devem ser retornados do Google. Esta questão merece ser comentada: quanto mais links o usuário desejar que seja retornado, maior será o tempo de resposta, pois são mais links a serem processados, e em contrapartida, a abrangência da resposta é maior; se ele escolher menos links, mais rápida será obtida a resposta mas com uma menor abrangência. Esta deve ser uma decisão a ser considerada pelo usuário no momento da inserção desse valor.

3. O grau de filtragem indica como deve ser o corte durante o processamento do algoritmo de filtragem, ou seja, que grau de importância deve ser considerado para que um link ou componente seja considerado importante para o usuário. Esse grau de filtragem é baseado na mesma escala em que foi fornecido pelo usuário durante o cadastro das palavras-chave importantes e também, pelo engenheiro de domínio, na recomendação

colaborativa. Os termos da escala, de acordo com o nível de importância, são os seguintes: Definitivamente Importante, Importante, Neutro, Pouco Importante e Não-Importante. Sempre que o usuário escolher um nível de filtragem, todos os links que tiverem o nível escolhido e acima dele, serão retornados para o usuário.

Por exemplo, se o usuário escolher a opção Importante, somente serão retornados os links considerados importantes e os que obtiveram uma classificação acima deste, que são os links definitivamente importantes. Essa opção é relevante, pois diminui o escopo da resposta para o usuário. O cálculo do método pelo qual é atribuído um grau de importância, para um link retornado, é mostrado na seção que descreve o algoritmo de filtragem (seção 5.1.5).

A consulta é encapsulada para ser enviada aos dois servidores de consulta: Google e Compublish. A interação com o serviço de busca Google acontece com a criação de um objeto Java, que efetua conexões com o site de pesquisa, que "escuta" o protocolo http. Após essa conexão ser aberta, o agente envia a consulta para o Google e recupera conjuntos de dez páginas[11] por vez. Cada página retornada, no formato HTML, é armazenada em um objeto em um buffer. A partir desse objeto, as informações são extraídas através de um parser que será detalhado na seção a seguir. O resultado do parser são objetos do tipo URLGoogle que serão filtrados, posteriormente, pelo agente de filtragem.

A consulta ao ComPublish é feita a partir de um método publicado na interface CORBA do servidor do ComPublish. A chamada consiste na passagem de dois parâmetros para o método: a consulta OQL (OMG, 2002), Figura 5.9, e o domínio que está sendo modelado. O seguinte método então é executado:

```
executeMetadataQuery("select c from c in Components where Name = \""+PalavraChave+ "\" or Description = \""+PalavraChave+ "\" or Category = \""+PalavraChave+ "\" or Phase = \""+parPalavraChave+ "\" or Language = \""+PalavraChave+ "\", Nome_Domínio);
```

Figura 5.9 – Consulta OQL para ComPublish

[11] Limite default do Google para o processamento de consultas, ou seja, o Google retorna esse número, normalmente, para cada consulta.

Na consulta OQL anterior estão sendo representados os predicados da consulta que a CompAgent precisa definir para que sejam retornados todos os objetos que pertençam a coleção de componentes e que existe pelo menos algum que interessa ao usuário. Essa consulta, enviada através do barramento ORB, é processada pelo ComPublish e é retornada pelo próprio barramento ORB, processo mostrado na figura 5.6. A resposta do ComPublish é feita através de um string, contendo um documento XML, um exemplo de resposta é mostrado na figura 5.7.

A partir do string retornado, é criado um documento XML que é passado para uma classe, a parserXML que, como o próprio nome indica, realiza a análise do documento XML. A análise é feita utilizando os pacotes *java.xml.parsers* e *org.xml.sax.helpers*, fornecidos pela SUN (Sun, 2002a) no kit JDK 1.3 (Java Development Kit). O pacote utiliza o padrão SAX (Simple API for XML) (Sun, 2002b) para extração de informações de documentos XML. Nesse padrão são extraídos objetos do tipo strings, sendo que cada objeto representa um elemento individual contido no esquema DTD[12], fornecido pelo ComPublish, do documento XML. Por exemplo, no caso do documento alvo da análise (retornado pelo ComPublish), exemplo na figura 5.7, serão obtidos os objetos mostrados na figura 5.10:

```
strDomain:=Radio
strName:=Diagrama de Classes de Enlace
strDate:=20/10/1998
strVersion:=1
strAuthor:=Equipe Coppetec
strDescription:=Classes de modelagem do conceito de enlace de transmissao via radio
strUrl:=ftp://LocalHost:3021/Telecommunication/Radio/Documentos/Diagrama de
Classes - Enlace.mdl
strContact:=pinheiro@cos.ufrj.br
strType:=Diagrama
strFileType:=mdl
strAssociatedApplications:=Rational Rose
```

Figura 5.10 – Strings obtidas do documento XML

Cada vez que o *parser* encontrar o *tag* </Component>, dentro da representação de um documento XML, significa que chegou ao fim da representação de um componente

[12] Representa o esquema que deve ser respeitado para que a construção de um determinado documento XML seja considerado válido semanticamente.

dentro do documento. Na seqüência, é instanciado, a partir das variáveis em memória (figura 5.10), um objeto da classe URLCompublish representando um componente retornado pelo Compublish. A análise do documento é realizada até ser encontrada o tag de fim de documento, no caso o *tag* </Components>. Esses objetos, assim como os instanciados a partir do parser dos links retornados pelo Google, são armazenados em um array de objetos, que são passados para o agente de filtragem. O conjunto de objetos da URL ComPublish e da URLGoogle são armazenados em *arrays* de objetos diferentes. Esse *array* de objetos é repassado ao agente de filtragem para a execução da tarefa de filtragem.

5.2.5 Extrator HTML

O agente de busca recupera as páginas HTML diretamente do Google. Cada página HTML possui um conjunto de dez links, que deverão ser extraídos e retornados objetos relacionados aos links para o módulo de filtragem, onde cada objeto representa um link retornado pelo Google. Para realizar a extração das informações das páginas, escritas na linguagem HTML, como objetos instanciados na linguagem Java, foi escolhida uma implementação manual.

Na implementação, cada página retornada é varrida desde o início na busca pelos tokens que interessam para a extração da informação. Os tokens que não interessam e que devem ser descartados são armazenados em uma tabela. No caso, os tokens são considerados as *tags* HTML. Quando uma tag é encontrada, ela é procurada na tabela, e se não encontrada, é considerada que a informação contida nessa tag deve ser retornada. Existe uma tag que é considerada para representar o final de um link. Quando esta é encontrada, um objeto do tipo link é instanciado e armazenado num vetor de objetos. Ao final, esse vetor de objetos é retornado ao módulo de filtragem.

Uma ferramenta de extração automática de informações poderia ter sido escolhida como a ASByE (Golgher et al., 2000) e a NoDoSe (Aldeberg, 1998). Contudo, essas ferramentas apresentam algumas desvantagens: ter que fornecer páginas exemplos para o aprendizado do formato das páginas HTML e, a partir do aprendizado, é gerado um extrator para essa página; cada vez que essa página for alterada, deve ser gerado um novo conjunto de páginas exemplos para o extrator. Como foi citado, nas ferramentas normalmente esses extratores funcionam mais adequadamente para páginas contendo tabelas, onde podem ser

definidos os delimitadores das tabelas e gerado um extrator para a página. Analisando as páginas retornadas pelo Google, foi verificado que os links, dentro da estrutura da página, não possuem forma tabular. Portanto, o extrator não apresenta a eficácia necessária para a resolução do problema. Por essas razões, foi escolhido se conceber um extrator próprio que atende a estrutura da página retornada pelo Google. Por ter sido criado internamente, a ferramenta tem uma manutenção mais fácil, tornando-se adaptativa às mudanças do Google.

5.2.6 Módulo de Filtragem

A função principal desse módulo consiste na filtragem de um conjunto de objetos (links) retornados pelo Google e de outro conjunto de objetos (componentes) retornados pelo ComPublish. A filtragem pode ser definida como a seleção dos objetos, a partir de uma lista retornada pelos servidores de consulta que, provavelmente, serão interessantes para o usuário. O usuário define o nível pelo qual deve ser feita essa filtragem, de acordo com o inserido na janela de busca (Figura 5.8). Além de realizar essa filtragem, a outra função do módulo é efetuar a classificação dos objetos. Após terem sido filtrados, são passados por uma classificação, que consiste no processo de ordenação dos links para serem mostrados ao usuário em uma ordem decrescente de importância.

O algoritmo de filtragem possui diversas etapas, onde cada etapa consiste de uma heurística que sempre consulta as informações existentes nas diversas bases de informações. Essas informações procuram prover algum conhecimento para a arquitetura a respeito dos links que foram retornados pelo agente de busca e que aquela tem que fornecer julgamentos a respeito destes.

Bem-vindo à Universidade Federal do Rio de Janeiro
... Mário Tavares e Edino Krieger receberam o Título de Doutor Honoris Causa no, último dia 10, no Salão Leopoldo Miguez da Escola de Música da **UFRJ**, que fica ...
Descrição: Ingresso na **UFRJ**, Cursos de Graduação, Recursos Disponíveis, Comissão Executiva, Contatos, Concursos...
Categoria: World > Português > ... > Brasil > Educação > Universidades
www.ufrj.br/ - 34k - Em cache - Páginas Semelhantes

Figura 5.11 - Link retornado pelo Google

Para cada link retornado, figura 5.11, são extraídas as seguintes informações: o título, que representa o título da página HTML retornada; endereço do link, que representa

a URL da página HTML; descrição do link, que representa o *snippet*[13] da página; descrição manual que representa o texto inserido pelo proprietário da página, no caso da página ter sido inserida manualmente no Google. A categoria do link não é extraída, pois não interessa para o processamento de filtragem da ferramenta.

No exemplo acima, seriam extraídos: **Título** - Bem-vindo à Universidade Federal do Rio de Janeiro; **Descrição**: Mário Tavares e Edino Krieger receberam o Título de Doutor Honoris Causa..; **Descrição Manual** – Ingresso na UFRJ, Cursos de Graduação, Recursos Disponíveis, Comissão Executiva, Contatos, Concursos; **Endereço do link** – www.ufrj.br/.

O processo de filtragem se inicia pela obtenção de um valor de classificação, RV, conforme a Equação 5.1, para cada um dos n links retornados. Esse valor é calculado a partir de várias equações que serão explicadas abaixo. Cada equação representa uma heurística utilizada pela ferramenta e é dependente do valor obtido pelo anterior, funcionando como um acumulador para a obtenção do valor final de classificação do referido link (Souza et al, 2001).

O valor RV_i é calculado depois que a função σ, que compara cada palavra-chave do perfil de usuário com os valores extraídos do link do Google, é calculada. Na Equação 5.1, para cada atributo a_a, o número de ocorrências de cada palavra-chave k_k são contabilizados e multiplicados pelo grau de importância(w_k) da palavra-chave(k_k) no perfil do usuário, onde, ainda, m é o número dos atributos existentes no link i e p é o número de palavras-chave do perfil do usuário.

$$RV_i = \sum_{a=1}^{m} \sum_{k=1}^{p} \sigma(k_k, a_a) * w_k \qquad \text{(Equação 5.1)}$$

O próximo passo consiste em checar se o link i já faz parte da base de recomendação ou da base de filtragem colaborativa. Se o link for encontrado, significa que o link foi anteriormente escolhido por um usuário do mesmo estereótipo, ou foi recomendado pelo engenheiro de domínio responsável pelo domínio corrente. Portanto, RV_i (Eq 5.2) incorpora o valor prévio (Equação 5.1) somando com o dobro dos pesos dos

[13] Trecho do texto, representando o documento inteiro, em que ocorrem as palavras desejadas na consulta que, segundo alguns autores como Casasola (1998), Pazziani et al. (1999), consegue uma boa representatividade do documento inteiro. Em Etzioni e Zamir (1998), é feito um estudo desta questão, tratando-a como *snippet-tolerance*.

links na base colaborativa (w_{rb}) e na base de filtragem colaborativa (w_{cf}) multiplicado pela relevância assinalada ao link na específica base. A relevância é representada como r_{rb} para a base de recomendação e r_{cf} para filtragem colaborativa.

$$RV_i = RV_i + 4 * (w_{rb} * r_{rb}) + 2 * (w_{cf} * r_{cf}) \qquad \text{(Equação 5.2)}$$

O terceiro passo do agente de filtragem é considerar a classificação do link retornado pela lista de resultados do Google, principalmente devido ao grau de confiabilidade fornecido pelo algoritmo de Page Rank, usado internamente por esse serviço de busca. A equação 5.3 calcula o novo valor de RVi, incorporando o valor prévio de RVi e adiciona um valor representando a classificação do link. Essa medida é a diferença entre o número dos links retornados (n_l) e a posição do link i lista retornada (pl_i). Esse valor é multiplicado por 0.3. Indicando que é considerado só um valor relativo a metade da posição do link.

$$RV_i = RV_i * (nl - pl_i) * 0.3) \qquad \text{(Equação 5.3)}$$

Depois de organizar os links de acordo os seus valores de classificação, o link com a maior valor de classificação é considerado como o principal do conjunto e serve como o valor de filtragem para os links restantes. O processo consiste em dividir o valor da classificação de cada link *i* (RV_i) pelo valor do link mais relevante (RV_h) (Equação 5.4). O valor obtido é comparado com o valor de corte previamente informado pelo usuário (Figura 5.9). No caso do valor da divisão (DV_i) ser menor que o valor de corte, o link é ignorado e não é inserido no conjunto de resultados a ser mostrado ao usuário.

$$DV_i = RV_i / RV_h \qquad \text{(Equação.5.4)}$$

Um processo similar é executado nos links retornados pelo ComPublish. Para a utilização das heurísticas, apresentadas acima, são consideradas as informações extraídas do documento XML contendo a representação dos componentes (Figura 5.10). Essas informações são comparadas com as palavras-chave do perfil do usuário, obedecendo a mesma equação apresentada na equação 5.1. No segundo passo, o link do componente é procurado nas bases de filtragem colaborativa e de recomendação colaborativa. Caso o link

do componente esteja em algumas dessas bases, obedecendo a equação 5.2, é somado ao valor encontrando na equação 5.1.

A posição dos componentes retornados, como a relativa aos links retornados pelo Google, na Equação 5.3, não é considerada. Pela especificação do ComPublish, esta não incorpora um algoritmo de classificação para estabelecer a importância do componente, portanto eles são retornados em uma ordem aleatória de importância. Sendo mantido o valor de classificação obtido na equação 5.2. Os componentes retornados são ordenados pelo seu valor de classificação e a filtragem dos componentes é definida através da equação 5.4, considerando, para divisor, o valor de classificação do componente que teve o valor mais alto entre todos e, para dividendo, o valor de classificação do link. O valor resultante da equação 5.4 é comparado com o ponto de corte definido pelo usuário. Caso seja menor, o componente não é inserido na resposta para o usuário.

Após a filtragem, os resultados dos componentes são unidos em um único conjunto. Os primeiros desse conjunto são os componentes filtrados a partir do ComPublish e, posteriormente, são listados os links, já filtrados, retornados pelo Google. Esse conjunto é mostrado ao usuário em uma janela criada na linguagem Java, que simula um browser Web (Figura 5.12). A partir desse momento, o usuário pode selecionar o componente ou o link que mais lhe interessar para o seu contexto. O Agente de Feedback entra em ação no momento da escolha de um link pelo usuário. O agente é explicado na próxima seção.

Na seção 5.3, é mostrado um exemplo com todos os detalhes de como é tratada a interação com o usuário.

5.2.7 Agente de Feedback

Esse agente é responsável pela atualização das bases de objetos utilizadas pela ferramenta. Esta atualização é feita a partir de informações recolhidas da observação, realizada de modo implícito, do comportamento do usuário.

A ferramenta retorna, para o usuário um browser com a resposta já previamente filtrada (Figura 5.12). Ao lado de cada link está localizado o seu grau de importância, numa escala contendo os seguintes valores: Definitivamente Importante, Importante, Neutro, Pouco Importante e Não-Importante. A partir dessa janela, o usuário pode selecionar o link ou componente que seja mais conveniente. A seleção é feita através do clique no título do

link. No caso de um componente ser escolhido, é feito um download do site onde o servidor do componente está instalado para a máquina do cliente, sendo que, posteriormente, este pode ser agregado ao projeto do usuário. No caso de um link, é aberto outro *browser* com a página escolhida pelo usuário. Nesse momento, o usuário pode salvar a página, através do browser, para consultá-la depois.

Essa seleção, para visualização do link ou para *download* do componente, indica para a ferramenta que houve um julgamento em relação a algum link ou componente retornado. Como ele foi selecionado pelo usuário, é considerado, implicitamente, como um julgamento positivo em relação às informações retornadas pela ferramenta. Portanto, a ferramenta irá atualizar a sua base para sempre possuir a informação mais apurada possível, de forma que apóie as próximas consultas feitas por usuários com o mesmo interesse do atual, ou qualquer um que utilize a ferramenta.

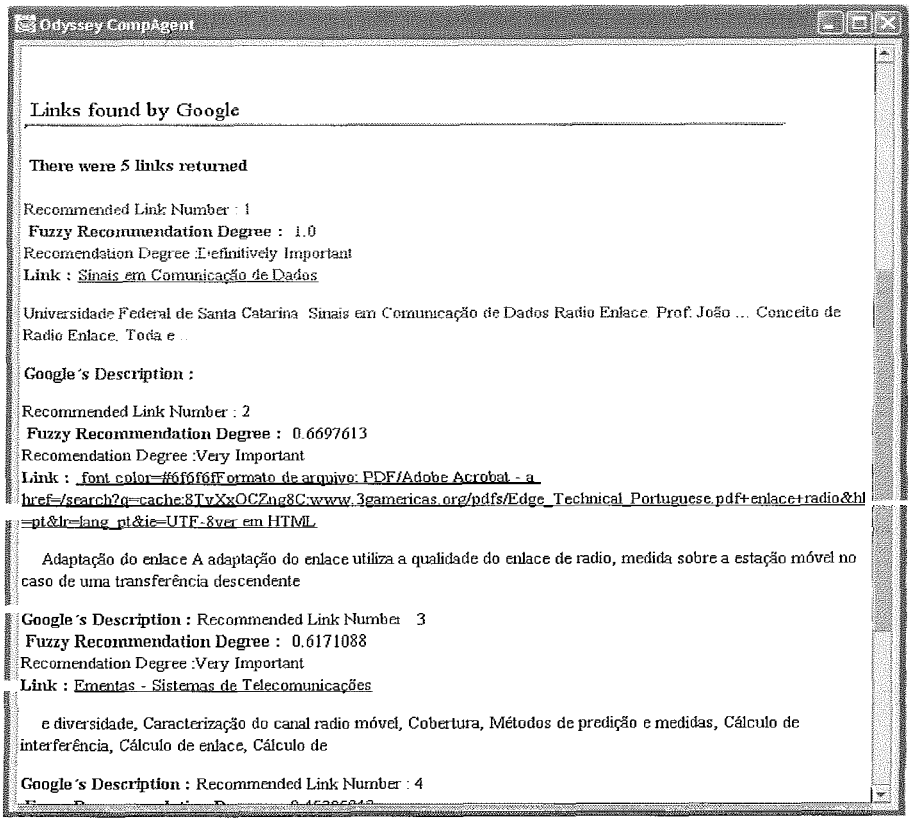


Figura 5.12 – *Browser* com Resultados

O agente procura o link selecionado pelo usuário para a visualização dentro da base de objetos de colaboração. Esta possui os links selecionados anteriormente por usuários do mesmo estereótipo. Para realizar a busca, é utilizada como chave o endereço (URL) do link. No caso do link apresentado na figura 5.11, o endereço é http://www.ufrj.br. Após essa busca, são obedecidas duas regras:

- **Link existente na base**: A ferramenta verifica se o link já existe na base. Nesta situação, o peso do links é acrescido em uma unidade e seu grau de importância é mantido o mesmo anterior. Por exemplo, o endereço http://www.ufrj.br já existe na base e seu peso possui o valor 9 e grau de importância Muito Importante (valor 0.8). O peso do link é aumentado para 10, contudo, o seu grau de importância é mantido inalterado.
- **Link inexistente na base**: Caso a busca não retorne nenhum link encontrado, o agente age da seguinte forma: considera todas as informações extraídas pelo parser HTML, incluindo a URL do link, o título do link e a descrição do link. É criado um novo objeto que, juntamente com as informações anteriores, também armazenará o grau de importância do link e o seu peso. O grau de importância do link é calculado pelo algoritmo de filtragem, sendo o valor é obtido a partir do resultado da equação 5.4. Como o link é novo na base, o seu peso é considerado uma unidade. Finalmente, este novo objeto é inserido na base de objetos de colaboração entre usuários. Na próxima execução da ferramenta, quando um usuário com perfil semelhante escolher esse link, será obedecida a regra acima, cujo peso é acrescido em uma unidade ao peso do link.

A outra ação tomada pelo agente trata da busca por esse link na base de recomendação colaborativa. A chave de busca continua a mesma do caso acima, ou seja, é feita pelo endereço do link. São obedecidas duas regras, semelhantes às do caso anterior:

- **Link Existente**: Nesta regra, o peso do link, na base de objetos é acrescido de uma unidade. No entanto, o grau de importância do link é mantido o mesmo anterior. A regra utilizada para o link http://www.cos.ufrj.br, no caso acima, é a mesma para este caso da base de objetos de links colaborativos.
- **Link Inexistente**: Nesse caso, o link somente será inserido na base de objetos se o usuário consultando tiver o papel de engenheiro de domínio no domínio sendo modelado. Sendo constatado que o usuário corrente tem o papel de engenheiro de domínio, então o mesmo procedimento da regra de link inexistente descrita anteriormente é realizado. É

criado um novo objeto com as informações extraídas pelo parser HTML, considerando peso um e grau de importância derivado da equação 5.4. O processo é terminado com a inserção do objeto na base.

Quando o link é inserido na base de objetos de recomendação colaborativa, ele passa a fazer parte do *bookmark* da ferramenta. Quando o usuário requisitar o *bookmark*, esse link constará deste e poderá ser selecionado pelo usuário. O agente de feedback atuará no bookmark com o mesmo comportamento descrito nas regras anteriores (feedback do usuário) em relação à base de colaboração.

5.3 Avaliação de uso da CompAgent

Toda avaliação,mostrada ao longo desta, foi implementada dentro do ambiente Odyssey, sendo que apenas a interação com o usuário, o seu perfil, interesses, os links armazenados nas bases de informação é que estão sendo simulados. Na avaliação, consideraremos que o usuário está modelando um subdomínio de telecomunicações que trata da comunicação via rádio. Esse usuário está interessado em entender os conceitos dessa subárea de conhecimento e, portanto, irá preencher o seu perfil conforme ilustrado na figura 5.13.

Pelo perfil do usuário, ele possui um nível de conhecimento médio em relação ao domínio, tendo como objetivo o entendimento da comunicação via rádio. Considerando essa característica, ele inseriu o entendimento do domínio no objetivo durante a navegação pelo Odyssey.

Antes da navegação e na realização de consultas pelo usuário durante sua navegação pelo domínio, a ferramenta já foi ajustada para atendê-lo da melhor maneira possível. O sistema de recomendação é o mais suscetível a esses ajustes, que é feito por um especialista por domínio para apoiar o usuário. Alguns desses ajustes são explicados a seguir.

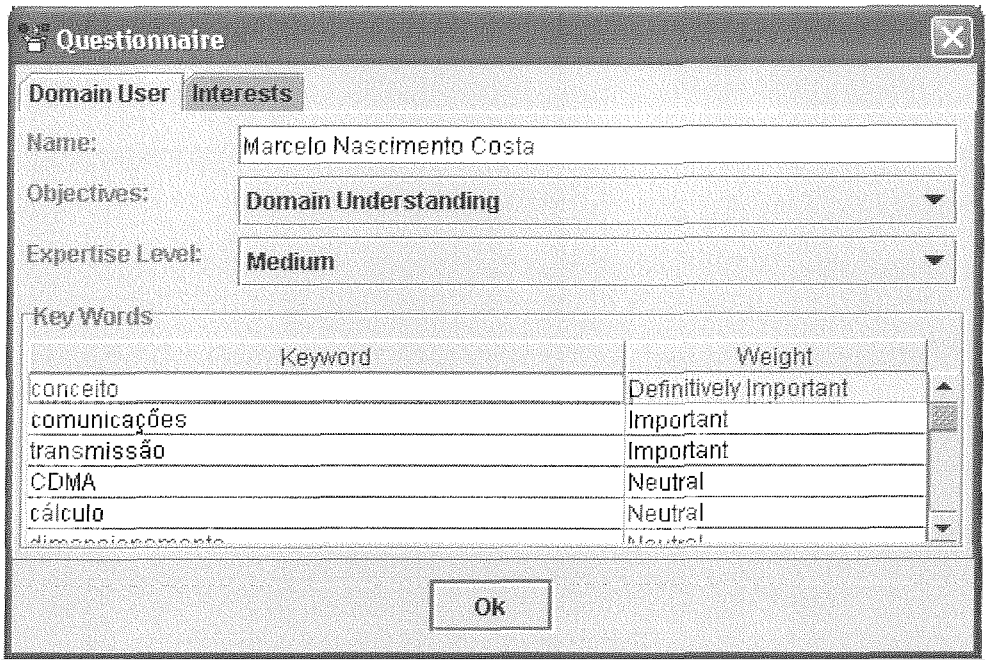
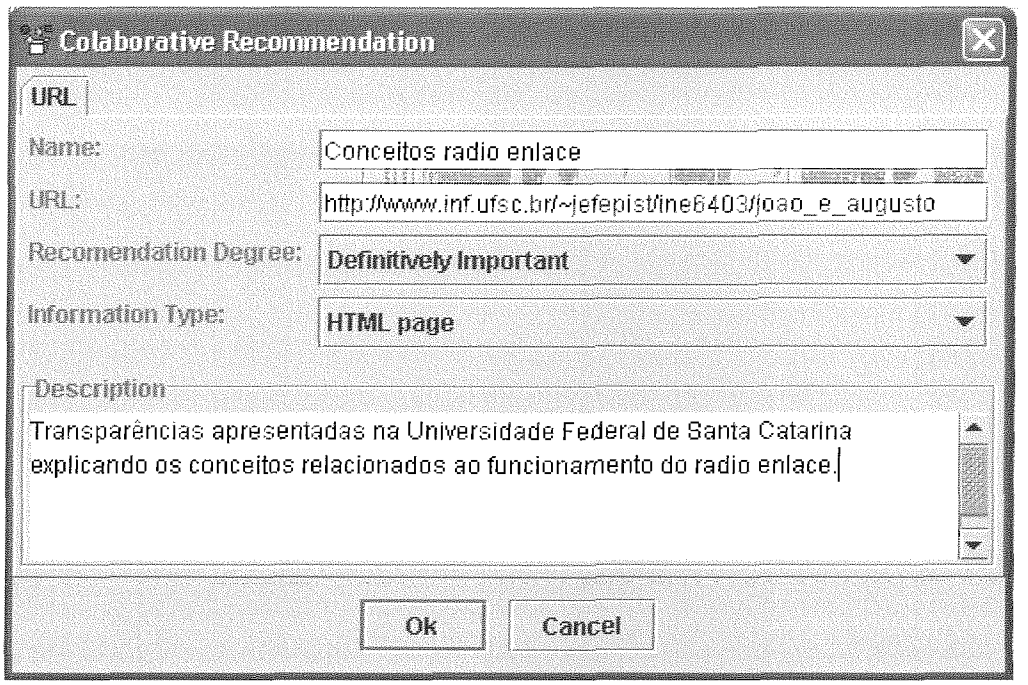


Figura 5.13 – Perfil do Usuário

As seguintes fontes de informações foram pesquisadas e analisadas por um especialista de domínio que as considerou importantes para o usuário entender o domínio Rádio. Essas fontes de informações foram encontradas em páginas HTML e, a seguir, cadastradas no sistema de recomendação colaborativa. A figura 5.14 mostra o cadastro de um link HTML, ilustrando o cadastro no sistema de recomendação.

Os outros links cadastrados são mostrados na tabela 5.1.

O especialista do domínio em nosso exemplo possuía ainda conhecimento a respeito de alguns componentes importantes para o conhecimento do usuário. O exemplo do cadastro de um componente é mostrado na figura 5.15.



5.14 – Link recomendado

Tabela 5.1 – Links cadastrados na base de recomendação

Nome	**URL**	**Grau Recomendação**	**Descrição**
Documento Técnico	www.3gamericas.org/pdfs/Edge_Technical_Portuguese.pdf	Important	Documento de referência para comunicação de dados via rádios
Glossário sobre telecomunicações	www.sidneydg.hpg.ig.com.br/ciencia_e_educacao/ 8/index_int_4.html	Important	Glossário a respeito de informações sobre telecomunicações

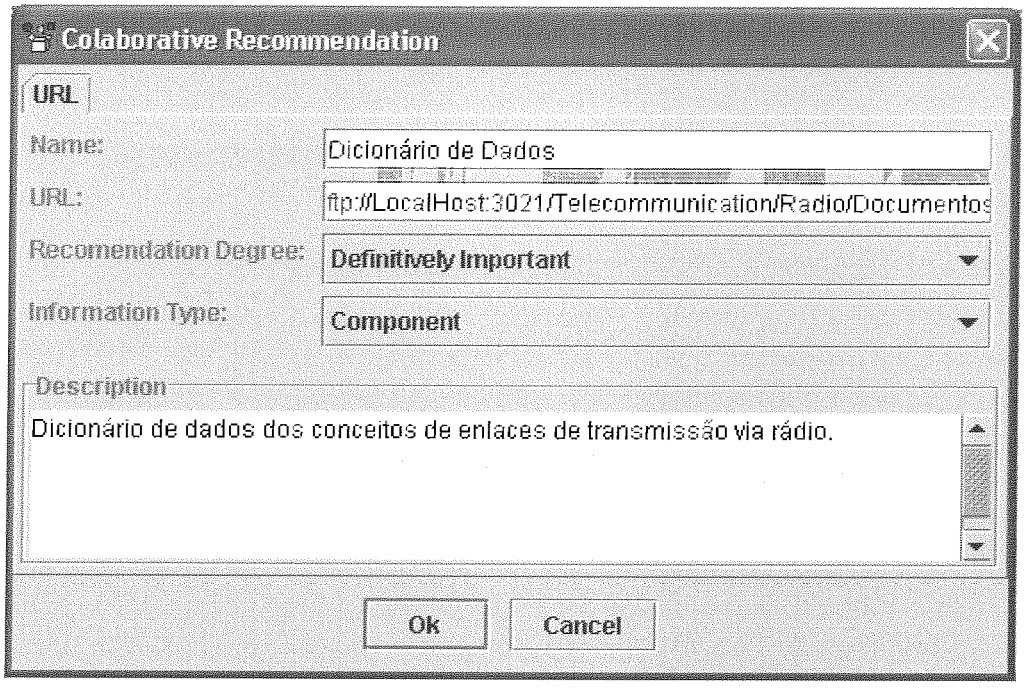


Figura 5.15 – Artefato de Software Recomendado

O seguinte componente também foi cadastrado (Tabela 5.2).

Tabela 5.2 – Artefatos de Software cadastrados

Nome	**URL**	**Grau Recomendação**	**Descrição**
Glossário sobre telecomunicações	ftp://LocalHost:3021/Telecommunication/Radio/Documentos/Diagrama de Classes - Enlace.mdl	Very Important	Classes de modelagem do conceito de enlace de transmissão via rádio.

Ao necessitar de alguma informação, o usuário executa a função de Busca na Web. Nesta janela, ele insere a seguinte consulta (Figura 5.16):

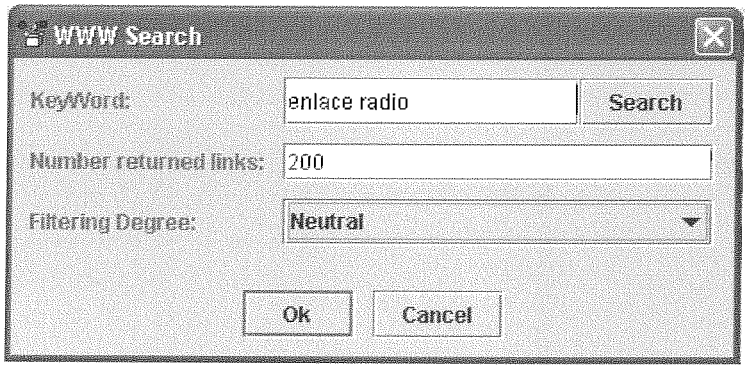


Figura 5.16 – Consulta do usuário

Uma consulta é enviada à ComPublish, retornando o seguinte conjunto de objetos, considerando já realizado o parser do documento XML (Ver Tabela 5.3).

Tabela 5.3 – Objetos retornados pela ComPublish

Objeto Compublish 1	Objeto ComPublish 2
Category:= Datum Domain:= Radio Name:= Diagrama de Classes de Enlace Date:= 20/10/1998 Version:= 1 Author:= Equipe Coppetec Description:= Classes de modelagem do conceito de enlace de transmissao via radio Url:=ftp://LocalHost:=3021/Telecommunication/Radio/Documentos/Diagrama de Classes - Enlace.mdl Contact:= pinheiro@cos.ufrj.br Type:= Diagrama FileType:= mdl AssociatedApplications:= Rational Rose	Category := Datum Domain:= Radio Name:= Dicionario de Dados de Enlace-Radio Date:= 05/07/1998 Version 3.1 Author:= Equipe Coppetec Description:= Dicionario de dados dos conceitos de enlaces de transmissao via radio Url:=Ftp://LocalHost:=3021/Telecommunication/Radio/Documentos/Dicionario de Dados.doc Contact:= pinheiro@cos.ufrj.br Type:= Texto FileType:= doc AssociatedApplications:= Microsoft Word

De acordo com a equação 5.1, todas as palavras-chave existentes no perfil do usuário são comparadas para cada atributo retornado nos objetos da ComPublish, multiplicando o número de ocorrências pelo grau de importância da palavra-chave. A palavra-chave conceito foi encontrada, com uma ocorrência no objeto1. Essa palavra possui grau de importância Definitively Important (peso 1.0). O mesmo ocorreu para a palavra transmissão. Contudo, ela possui grau de importância Important (peso 0.8). A palavra-chave conceitos foi encontrada com uma ocorrência no objeto 2. Essa palavra possui grau de importância Definitively Important (peso 1.0). O mesmo ocorreu para a palavra transmissão que apresenta grau de importância Important (peso 0.8). Esses valores são multiplicados e depois somados para cada palavra-chave, obtendo-se o seguinte resultado parcial de classificação apresentada na Tabela 5.4:

Tabela 5.4 – Valor de classificação obtido da Equação 5.1

Número Objeto	Valor de Classificação
1	RV1 = 1 * 1.0 + 1 * 0.8 = 1.8
2	RV2 = 1 * 1.0 + 1 * 0.8 = 1.8

De acordo com a equação 5.2, a existência desses objetos é verificada na base de recomendação colaborativa e na base de filtragem colaborativa. O objeto 1 foi encontrado na base de recomendação colaborativa com peso 1 e grau de importância 0.8, porém não foi encontrado na base de filtragem colaborativa. O objeto 2 foi encontrado na base de recomendação colaborativa e de filtragem colaborativa com peso 2 em cada base e grau de importância um. Esses resultados das consultas à base de recomendação colaborativa e de filtragem colaborativa são adicionados ao valor anterior:

Tabela 5.5 – Valor de classificação obtido da Equação 5.2

Número Objeto	Valor de Classificação
1	$RV_1 = RV1 + 4 * (0.8) * 1 \Rightarrow RV_1 = 1.8 + 3.2 = 5.0$
2	$RV_2 = RV2 + 4 * (0.8) * 2 + 2 * (1.0) * 1 \Rightarrow RV_2 = 1.8 + 6.4 + 2.0 = 9.8$

Após o cálculo do valor de classificação, os objetos são ordenados. Com a ordenação o objeto 2 é o primeiro na lista (valor 9.8) e o objeto 1 é o segundo da lista (5.0). O valor normalizado a ser retirado é o de maior valor, isto é 19.8. O grau de corte definido foi o neutro, ou seja, todos os valores obtidos maiores do que 0.4 (CV) deverão ser incluídos para a visualização do usuário. Obedecendo a equação 5.4, a lista de resultado é montada conforme apresentado na Figura 5.17.

$CV_i = 0.4$; $DV_h = 9.8$

$DV_1 = DV_1/DV_h = 5.0 / 9.8 = 0.51$

Figura 5.17 – Cálculo final para obtenção da classificação

Portanto, com o valor 0.51 o objeto 1 deve ser incluído, pois tem valor de filtragem maior que o valor de corte. Ele terá como valor de classificação Neutral, pois ficou na faixa

entre 0.4 e 0.6. O segundo objeto terá valor 1, pois será divido pelo seu próprio valor de classificação. O resultado é mostrado na figura 5.18.

Após a filtragem dos componentes retornados pelo ComPublish, ocorre a filtragem das informações retornadas pelo Google. Na tabela 5.6, é listada uma amostra do conjunto dos links retornados a partir da consulta da figura 5.16, e que serão filtrados pela CompAgent. Foram escolhidos links que não receberão valor nenhum de classificação, alguns que receberão valor de classificação mas não passarão pela filtragem e o restante que será selecionado para ser mostrado ao usuário.

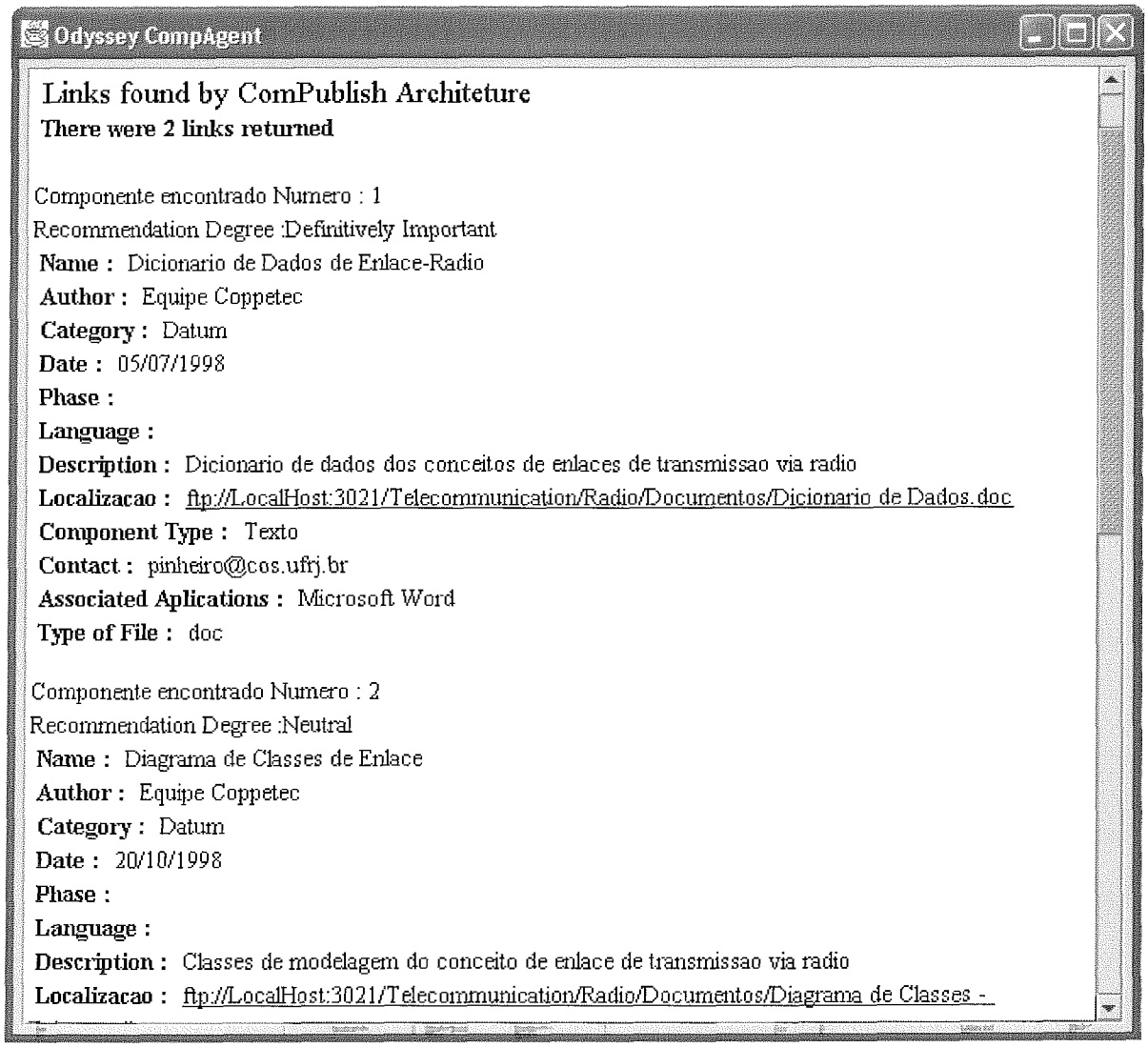


Figura 5.18 – Parte do browser com o resultado do ComPublish

Tabela 5.6 – Amostra dos links retornados

Número Objeto	Link retornado	Posição Google
1	**Título**: Planet Radio » Dramas **Descrição**: Portuguesa - Oboré; Enlace: Música e Culturas dos Povos de Língua Portuguesa - Oboré; Frontera caliente - Tarsicio García Oliva; **Link**:moebius.amarc.org/planet_radio/dramasdb.html?cat_name=Reportajes	1
2	**Título**: Bridge On Line - Radio **Descrição**: Quanto custa? INICIO. O custo do Acesso Dedicado Bridge On Line – RÁDIO (Enlace Rádio) varia de acordo com a capacidade do canal contratado. **Link**: www.bridge.com.br/s_add_radio.asp	2
3	**Título**: Sinais em Comunicação de Dados Descrição: Universidade Federal de Santa Catarina. Sinais em Comunicação de Dados Radio Enlace. Prof: João Conceito de Radio Enlace. Toda e **Link**: www.inf.ufsc.br/~jefepist/ine6403/joao_e_augusto/	8
4	**Título**: Competence Center - Sistemas Celulares Descrição: e protocolos CDMA 2000; pilha de protocolo CDMA 2000, camada física e camada de enlace, MAC(médium acess control), Multiplexação, SRBP (Signaling Radio **Link**:	15
5	**Título**: Ementas - Sistemas de Telecomunicações **Descrição**: e diversidade, Caracterização do canal radio móvel, Cobertura, Métodos de predição e medidas, Cálculo de interferência, Cálculo de enlace, **Link**: www.cce.puc-rio.br/ementas/esptelecomunicacoes.htm	29
6	**Título**: documento de Referência **Descrição**: Adaptação do enlace A adaptação do enlace utiliza a qualidade do enlace de radio, medida sobre a estação móvel no caso de uma transferência descendente **Link**: http://www.3gamericas.org/pdfs/Edge_Technical_Portuguese.pdf	31
7	**Título**: ENGTECH **Descrição**: Telefones celulares, radio-comunicadores, radio-modems e outros deixaram de ser ... Dimensionamento de Enlaces Estudo do enlace, dimensionamento dos equipamentos **Link**:	42
8	**Título**: Professor Wagner Zanco **Descrição**: Full-duplex - O enlace é utilizado nos dois sentidos possíveis de transmissão ... são transmitidas "através do ar", em canais de freqüências de radio **Link**:www.wagnerzanco.hpg.com.br/cursos/telecomunicacoes/capitulo1/meiosdetransmissao.htm	46
9	**Título**: CDMA **Descrição**: O CDMA tem avançado sua aceitação internacional entre operadoras de sistemas de radio O enlace com a primeira ERB só será interrompido quando o nível do **Link**: www.senwt.hpg.ig.com.br/cdma.htm	106

De acordo com a equação 5.1, são comparadas todas as palavras-chave existentes no perfil do usuário para cada atributo retornado pela Google, multiplicando o número de ocorrências, em cada atributo, pelo grau de importância da palavra-chave. São gerados os seguintes valores:

Tabela 5.7 - Valor de classificação obtido da Equação 5.1

Número Objeto	Valor de Classificação
1	Não foi encontrada palavra-chave no titulo e descrição
2	Não foi encontrada palavra-chave no titulo e descrição
3	Encontrou palavra-chave Conceito em uma posição RV3 = 1 * 1.0 = 1.2
4	Encontrou palavra-chave CDMA em duas posições RV4 = 2 * 0.6 = 1.2
5	Encontrou palavra-chave cálculo em três posições RV5 = 3 * 0.6 = 1.8
6	Não foi encontrada palavra-chave no titulo e descrição
7	Encontrou palavra-chave dimensionamento em duas posições RV7 = 2 * 0.8 = 1.6
8	Encontrou palavra-chave transmissão em uma posição RV8 = 1 * 0.8 = 0.8
9	Encontrou palavra-chave transmissão em uma posição RV8 = 2 * 0.6 = 1.2

De acordo com a equação 5.2, a existência prévia do cadastro desses links é verificada na base de recomendação colaborativa e na base de filtragem colaborativa. Após o cálculo da Equação 5.2, é acumulada a posição do link no conjunto retornado pelo Google, de acordo com a equação 5.3.

No próximo passo, são ordenados os elementos de acordo com o seu valor de classificação e efetivamente é realizada a filtragem destes. O usuário escolheu o grau de filtragem *Neutral*, ou seja, serão considerados todos os elementos que obtiverem, da equação 5.4, valor maior do que 0.4 (CVi = 0.4). Esse cálculo é mostrado na tabela 5.9. Pela tabela 5.8, o elemento com maior valor é o objeto 3, então ele considerado como Definitively Important e o seu valor de classificação (DVh = 301,56) será o valor normalizado.

Tabela 5.8 - Valor de classificação obtido da Equação 5.2 e da Equação 5.3

Número Objeto	Valor de classificação	Posição Google
1	Não foi encontrada nas bases colaborativa e de filtragem	1
2	Não foi encontrada nas bases colaborativa e de filtragem	2
3	Foi encontrada na base recomendação colaborativa RV3 = RV3 + 4 * (0.8) => 1.2 + 3.2 = 4.4 RV3 = RV3 * ((200 – 7) * 0.3) = > 4.4 * 57,9 = 301.56	7
4	Foi encontrada na base de filtragem colaborativa RV4 = 1.2 7+ 2 * (0.6) = 2.4 RV4 = RV4 * ((200 – 15) * 0.3) = > 2.4 * 55.5 = 133.2	15
5	Foi encontrada na base de filtragem colaborativa RV5 = 1.8 + 2 * (1.0) = 3.8 RV5 = 3.8 * ((200 – 28) * 0.3) => 196.12001	28
6	Foi encontrada na base recomendação colaborativa RV6 = 4* 1.0= 4.0 RV6 = 4.0 * ((200 – 32) * 0.3) => 201,6	32
7	Foi encontrada na base de filtragem colaborativa RV7 = 1.6 + 2 * (0.8) = 3.2 RV7 = 3.2 * ((200 – 58) * 0.3) => 136,32	58
8	RV8 = 0.8 * ((200 – 45) * 0.3) => 13,95	45
9	RV9 = 1.2 * ((200 – 106) * 0.3) => 33.84	106

Tabela 5.9 – Resultado final de classificação

Número Objeto	Valor de classificação	Posição Retorno
3	DV3= DV3/Dvh = 301,56/301,56 = 1.0 ⇔ DV3 > CVi	1
6	DV6= DV6/Dvh = 201,56/301,56 = 0.66 ⇔ DV3 > CVi	2
5	DV5= DV5/Dvh = 196,12,56/301,56 = 0.65 ⇔ DV3 > CVi	3
7	DV7= DV7/Dvh = 136,32,56/301,56 = 0.45 ⇔ DV3 > CVi	4
4	DV7= DV7/Dvh = 136,32,56/301,56 = 0.45 ⇔ DV3 > CVi	5
9	DV9= DV9/Dvh = 33.84/301,56 = 0.45 ⇔ DV9 < CVi	-
8	DV8= DV8/Dvh = 13,95/301,56 = 0.45 ⇔ DV9 < CVi	-
1	DV1 = 0 ⇔ DV1 < CVi	-
2	DV2 = 0 ⇔ DV2 < CVi	-

Como resultado final são listados os links na Figura 5.19 e Figura 5.20.

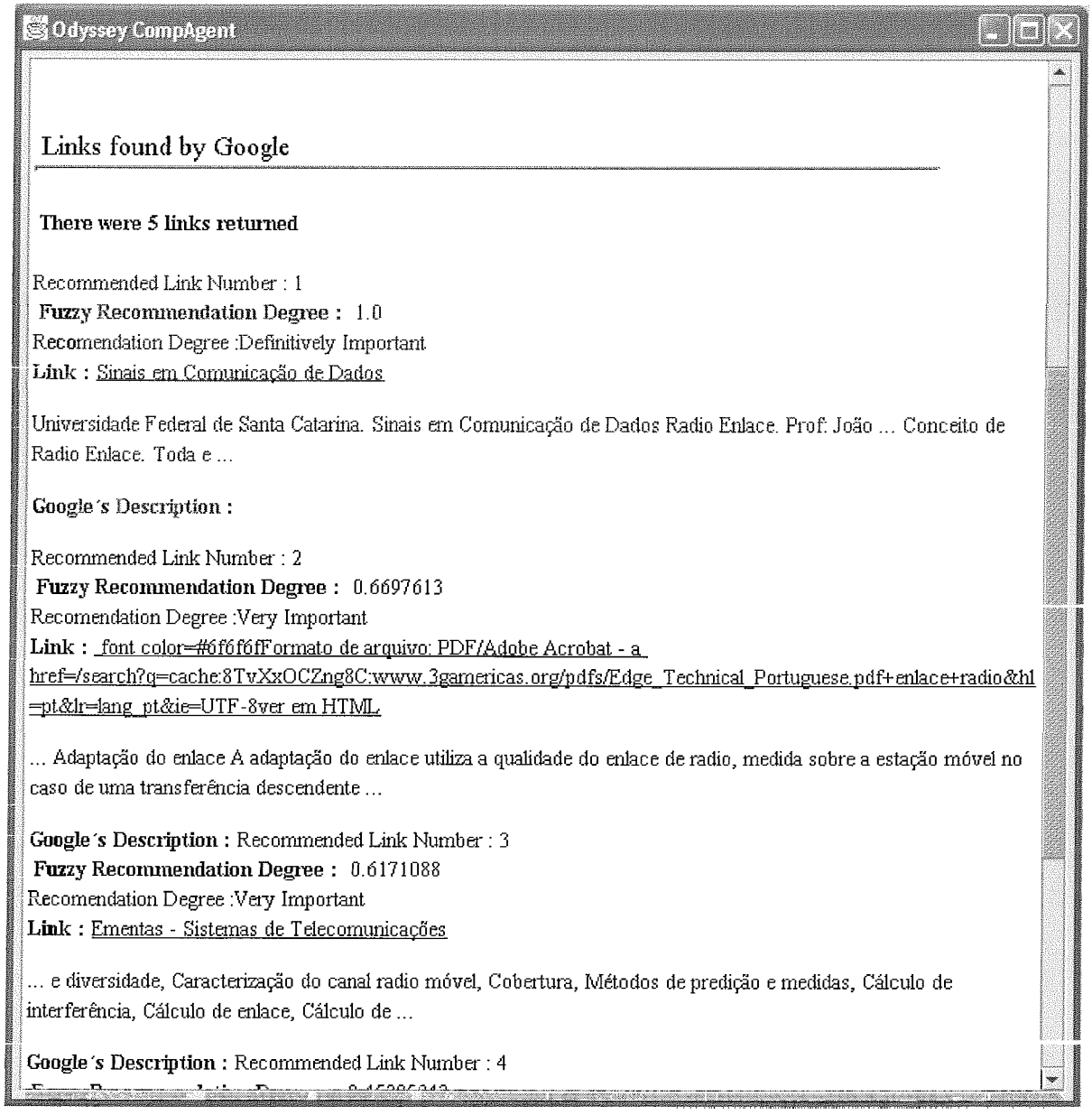


Figura 5.19 – Browser gerado pelo CompAgent

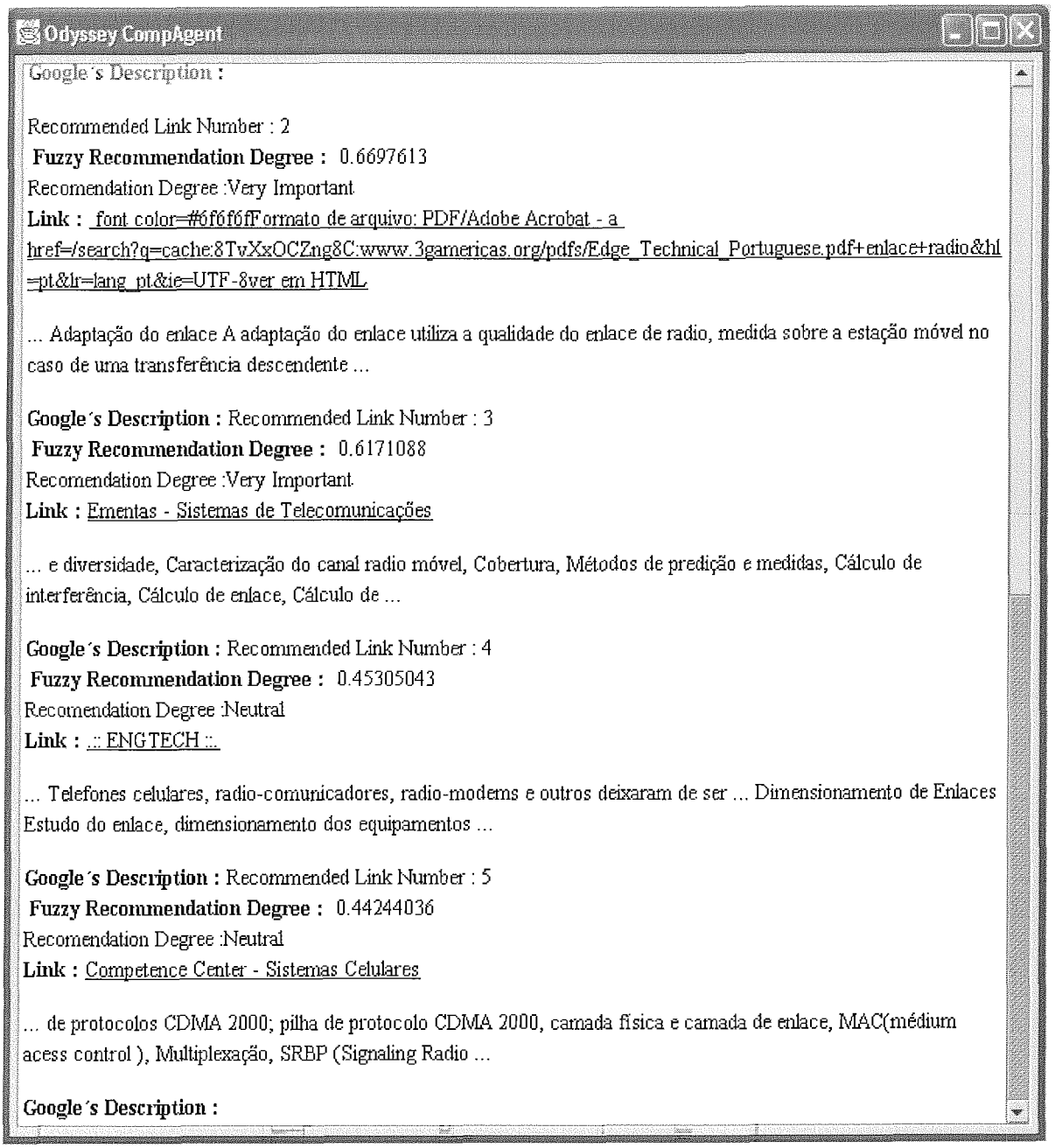


Figura 5.20 – Complemento do browser gerado pela CompAgent

Para a mesma consulta, o Google retorna a página apresentada na Figura 5.21.


Pesquisadas **Portugues** páginas para **enlace radio**. Resultados **1** - **10** sobre 471. A pesquisa demorou **0.23** segundos.

Planet **Radio** » Dramas
... **Enlace**: Música e Culturas dos Povos de Língua Portuguesa - Oboré; Frontera caliente - Tarsicio García Oliva; La Aynata Aymara - **Radio** Arte y ERBOL (La Paz ...
moebius.amarc.org/planet_radio/ dramasdb.html?cat_name=Reportajes - 14k - Em cache - Páginas Semelhantes

Soluções Click - **Radio**
... Voltar. Instalação, Alinhamento e Teste de Aceitação. - Instalação do **enlace** de rádio; - Ajuste do **enlace** (alinhamento das antenas e outras ...
www.clickti.com.br/index_Sol_Radio.htm - 8k - Em cache - Páginas Semelhantes

Bridge On Line - **Radio**
... INICIO. O custo do Acesso Dedicado Bridge On Line - RÁDIO (**Enlace** Rádio) varia de acordo com a capacidade do canal contratado. Consulte a tabela abaixo: ...
www.bridge.com.br/s_add_radio.asp - 50k - Em cache - Páginas Semelhantes

Radio Brasil
Volver al indice de selección **enlace** de Paises. Emisoras en: "Brasil", **Enlace** Real Audio. ... Globo - 1180 AM, Globo - 1180 AM. **RADIO** CIDADE, SALVADOR, BAHIA, BRASIL. ...
www.ali-atrac.es/radio_emisoras/brasil.html - 8k - Em cache - Páginas Semelhantes

Radio Internacional de China
... 5- Rádio **Enlace**, Rádio Nederland. 6- Actualidade DX Radiodifusão Argentina. 7- Amigos de la Onda Corta, Rádio Exterior da Espanha. 8- World of **Radio**, WWCR eR ...
www.crinews.com/portugal/2000/Sep/3882.htm - 26k - Em cache - Páginas Semelhantes

A Telefonia Virtual > Frequências Espanha > Valencia
... 96.9, Kiss FM. 97.7, LA 97.7 **Radio**. 98.0, **Radio** Gondella. 98.1, **Radio Enlace**. 98.4, Cadena Dial. 98.7, Canal 13 **Radio**. 99.0, Cadena 100. 99.6 / 103.0, Si **Radio**. 99.9, **Radio** Luz. ...
www.telefonia-virtual.com/freqesvalencia.shtml - 44k - Em cache - Páginas Semelhantes

Microfone: o site do radialista !
... Amigos de la Onda Corta: transmitido em espanhol pela Rádio Exterior de España. **Radio-Enlace**: transmitido em espanhol pela Rádio Nederland. ...
www.microfone.jor.br/links_radioescuta.htm - 24k - Em cache - Páginas Semelhantes

Figura 5.21– Consulta similar retornada pelo Google

Comparando com a consulta retornada pela CompAgent, pode ser verificada que o Google retornaria um grande quantidade de links irrelevantes. No caso desta consulta seriam 471 links. Essa grande quantidade de links é considerado o *overload* de informações para o usuário, que teria que navegar por essas diversas páginas de resposta do Google até encontrar algum link interessante. A resposta da CompAgent é bem mais resumida, indicando, com um grau de relevância, o que realmente seria interessante para o usuário, listando inclusive links para componentes armazenados no sistema ComPublish.

O desempenho é uma questão a ser abordada. A ferramenta leva alguns minutos para realizar todo o seu processo de busca e de filtragem de informações. Alguns autores, como Kerschberg et al. (2001), afirmam que o tempo de espera do usuário para receber a resposta pode compensar com relação ao benefício de se encontrar realmente o que se deseja ou, pelo menos, algo aproximado. O retorno do Google pode até ser consideravelmente rápido, mas o volume de informações é muito grande e, provavelmente, o usuário perderá um tempo superior procurando por um link importante.

5.3 Conclusão

Neste capítulo foi abordada a implementação da ferramenta CompAgent no contexto do ambiente Odyssey. Foi descrita a sua função neste ambiente, que consiste em prover a seus usuários um suporte à busca e recuperação de informações na Web para os seus usuários. Esse suporte não se limita simplesmente à busca de informações, procurando ser específico para atender as necessidades de cada usuário.

Foi descrito como a CompAgent obtém a colaboração entre usuários, permitindo que os mais experientes no domínio indicassem informações importantes para os menos experientes, e que usuários compartilhassem conhecimento entre aqueles com interesses semelhantes. O compartilhamento de conhecimento entre usuários não experientes é construído implicitamente, através da capacidade adaptativa da ferramenta. Essa adaptação é feita através da análise implícita do comportamento de cada usuário.

Foi mostrado como é realizada a integração da CompAgent ao mundo externo, representado pelo serviço de busca Google, e o sistema para armazenamento e recuperação de componentes ComPublish. Esses módulos externos compõem a metabusca realizada, onde a partir desta foi mostrada a execução da filtragem das informações retornadas.

Conforme descrita, a filtragem dos links utiliza diversas heurísticas que, por sua vez, são baseadas em algumas técnicas, tais como filtragem colaborativa, recomendação colaborativa e perfil do usuário. Essas técnicas são eficazes, sendo ressaltado no entanto, que é necessária uma preparação da ferramenta antecipadamente por parte dos responsáveis pelo domínio. Os especialistas precisam inserir os links importantes para que outros usuários os encontrem nas suas pesquisas ao Google ou ComPublish, ou no momento que um usuário tiver a necessidade de gerar um *bookmark* de recomendações.

Outra questão trata da inserção das informações no perfil do usuário. O mesmo tem que analisar quais informações ele deve inserir que melhor represente o seu perfil e, principalmente, quais palavras-chave são importantes para realizar sua modelagem. No entanto, ele pode também dizer que uma palavra não é tão importante para ele. Como a ferramenta é adaptativa, principalmente entre usuários, quanto mais ela for utilizada em um determinado domínio, mais retornos ela obterá do usuário, melhorando consideravelmente a sua eficiência na resposta ao usuário neste domínio.

Na avaliação apresentada no capítulo mostrou-se como a ferramenta pode ser útil, comparando com a grande quantidade de links retornados pelo Google para a mesma consulta. Vale observar que, neste exemplo, havia uma preparação da ferramenta, por parte de especialistas desse domínio com links indicados pelos mesmos. Ela também já havia sofrido uma adaptação por parte dos usuários com os mesmos interesses, pois no caso a base de filtragem colaborativa já possuía links que foram considerados importantes por outros usuários semelhantes. Nos cálculos mostrados, foi percebida essa adaptação, pois, devido aos links já estarem na base, os valores de classificação destes foram alterados e, conseqüentemente, mudando a ordem em relação aos outros. Os cálculos indicam que, provavelmente, estes links deveriam ser retornados.

Capítulo 6 - Conclusão

O processo de Engenharia de Domínio(ED) e seu correspondente processo de Engenharia de Aplicação (EA) necessitam de um conjunto de ferramentas que apóiem a sua realização. A principal ferramenta é um ADS para apoiar todas as atividades dos dois. Para se entender um domínio é preciso aprofundar o conhecimento relativo sobre ele. Entretanto, a obtenção deste conhecimento é um problema importante enfrentado durante a modelagem do domínio, e envolve diferentes fontes de informação que podem estar dispersas. Dependendo do domínio em questão, o volume de informações pode ser muito grande. O usuário que participa da modelagem do domínio pode ter sérias dificuldades em relação a onde e como obter essas informações que lhe seriam úteis para efetivar a modelagem.

A Web é atualmente uma fonte valiosa de conhecimento, mas, ao mesmo tempo em que ela ajuda o usuário, ela pode se tornar um espaço muito grande para consultas feitas pelos usuários, podendo dificultar a recuperação de informações. O principal motivo está relacionado ao fato de que as informações são distribuídas pelos mais diversos sites e nem sempre são relevantes ou estão validadas. Portanto, os sites apresentados em uma busca genérica não servem como fonte de conhecimento confiável.

A ferramenta CompAgent, apresentada nesta dissertação, provê meios para a solução do problema de recuperação de informações específicas na Web, no contexto de um ambiente de reutilização de software. Outro objetivo importante está relacionado à busca e recuperação de componentes para ajudar nos processos de ED/EA. A ferramenta foi projetada procurando atender requisitos específicos necessários em um ambiente de modelagem de ED/EA como: retorno de respostas inseridas no contexto do domínio; atendimento às preferências específicas do usuário; busca orientada por especialistas que conheçam profundamente o domínio; colaboração entre usuários com perfis semelhantes e superação à sobrecarga de informações retornadas em uma consulta.

Como foi mostrado nos capítulos anteriores, foram implementadas diversas características importantes e largamente utilizadas na literatura. Algumas técnicas pertencem à área de BD: mediadores, realizando buscas a uma arquitetura de mediação de banco de dados heterogêneos; filtragem colaborativa, onde são compartilhadas as bases de informações por usuários com interesses similares; e recomendação colaborativa, na qual

usuários experientes indicam componentes que são considerados importantes para usuários comuns. Outras técnicas pertencem à área de IA: modelagem de usuário, onde são cadastrados os interesses dos usuários, e estes são agrupados em estereótipos, representando usuários com interesses em comum; e *relevance feedback*, que observa o comportamento do usuário para atualizar as suas bases de informação. A ferramenta proposta tenta se moldar ao comportamento dos usuários, para ser o mais eficaz possível no atendimento às necessidades impostas.

Um outro problema da Web tratado neste trabalho diz respeito a apresentação de informações válidas ao usuário, através da recomendação colaborativa, em que links já validados são apresentados pelo responsável da modelagem do domínio (engenheiro de domínio). A Engenharia da Aplicação é então apoiada através da recomendação colaborativa, com a apresentação de componentes que, provavelmente, serão úteis para a construção da aplicação.

6.1 Contribuições

Esta dissertação apresenta uma contribuição em uma corrente de pesquisa específica na área de Recuperação de Informação, que consiste na personalização de busca (Lawrence, 2000). Essa área trata da completa personalização da busca, ou seja, o serviço de busca conhece todas as requisições prévias e os interesses do usuário, podendo utilizar essas informações para ajustar os resultados a serem retornados ao usuário. Em (Lawrence, 2000), são indicadas duas formas de personalização: no cliente ou no servidor. No caso desta dissertação foi construído um módulo de personalização no cliente, que controla os interesses do usuário e do grupo de usuários similares, a fim de modelar um conjunto de informações a respeito desse usuário. Todas essas informações no cliente fornecem um contexto a sua consulta, não o deixando completamente livre num imenso volume de informações como a Web.

Na tabela 2.2 foram mostradas algumas características das ferramentas estudadas no capítulo 2. Após a descrição detalhada da implementação ferramenta CompAgent, podemos montar novamente a tabela, incluindo-a na comparação (tabela 6.1).

Tabela 6.1 – CompAgent em relação a outras ferramentas

Ferramenta	Usuários-Alvo	Tipo de Recomendação	Tipo de Busca	Tipo de Arquitetura	Forma de Colaboração
CompAgent	Usuários procurando documentos para o processo de ED/EA	Documentos WWW e artefatos de software.	Metabusca	Agentes Individual	Colaborativo
Syskill & Webert	Usuários navegando por páginas Web	Links WWW	Predição de links	Agentes Individual	Individual
Fab	Usuários de um serviço de recomendação de documentos	Documentos WWW	Predição de links	Agentes Individual	Colaborativo
Oyster	Usuários buscando documentos WWW	Documentos WWW	Metabusca	Sistema Multi-Agente	Colaborativo
WebWatcher	Usuários procurando links importantes	Links de navegação entre páginas WWW	Predição de links	Agentes Individual	Colaborativo
Letizia	Usuários procurando links importantes	Links de navegação entre páginas WWW	Predição de links	Sistema Multi-Agente	Individual
Inquirus 2	Usuários buscando documentos WWW	Documentos WWW	Metabusca	Agentes Individual	Colaborativo

Analisando a CompAgent em relação às outras ferramentas, percebe-se sua especificidade em atender um determinado domínio de problema, que no caso é o apoio às atividades do processo de ED/EA. Ao apoiar estas atividades, a busca e recuperação não ficam restritos a apenas links ou documentos WWW, ampliando seu conjunto de busca para que artefatos de software possam ser recuperados de modo a serem reutilizados.

As ferramentas relacionadas na Tabela 6.1 são abrangentes, atingindo diversas categorias de usuários, mas, no entanto, retornam diversos tipos de informações, dentre os

quais, podem ser encontrados componentes. Diferentemente da ferramenta implementada, que possui características específicas para apoiar a Engenharia de Aplicação e de Domínio, tal como a possibilidade de se conectar a uma arquitetura otimizada, especificamente para a publicação e recuperação de artefatos de software, a ComPublish. A CompAgent possui ainda os módulos de recomendação e filtragem colaborativa, construídos com o intuito de abordar a recomendação e armazenamento de informações a respeito de artefatos de software.

Na literatura não foi encontrada nenhuma ferramenta que abrangesse esse segmento alvo da CompAgent, que é composto por usuários de um ambiente de reutilização de software. Inclusive, a CompAgent foi acoplada, internamente, a um ambiente de reutilização de software, utilizando alguns conceitos inerentes a este ambiente, a exemplo da divisão por papéis utilizados por cada usuário em diferentes processos de Engenharia de Domínio. Entretanto, deve ser frisado que a ferramenta pode ser incorporada, com alguns ajustes, a qualquer outro ambiente de reutilização de software.

6.2 Limitações e Trabalhos futuros

Durante a construção da ferramenta foram detectadas algumas limitações, ajustes ou novas tecnologias que surgiram nas áreas de pesquisas abordadas pela dissertação que poderiam melhorar ou dar uma nova forma de abordagem ao mesmo problema.

O processo de filtragem pode ser ajustado. Uma melhoria seria alterar o algoritmo que compara palavras-chave do perfil do usuário com as informações retornadas pelo Google e ComPublish, para fornecer "maior semântica" para este algoritmo. Para exemplificar a aplicação de uma "maior semântica", considere a comparação da palavra "conceito" informada no perfil do usuário com "conceitos" retornadas na descrição de um link. O algoritmo poderia considerar ambos os termos como sendo a mesma palavra, pois possuem a mesma semântica.

A comparação entre links também poderia ser melhorada, fornecendo graus de similaridade. No atual algoritmo esta comparação funciona como uma lógica booleana, ou seja, ou o link é exatamente igual ao outro, ou então não é considerado. Poderia ser feita uma comparação parcial, onde um link poderia ser igual até determinada parte da URL, ganhando um peso parcial de similaridade.

Um trabalho importante seria testar a exaustão das buscas feitas pelos usuários, na tentativa de ajustar os pesos do algoritmo de filtragem. Os pesos foram escolhidos de forma aleatória, sendo que o ideal seria formar as equações, utilizadas na filtragem, com pesos de forma que estas se ajustassem ao máximo à realidade do usuário. Cada ajuste feito no algoritmo deveria verificar se atende realmente ao usuário.

A ferramenta pode ser mais pró-ativa em relação aos usuários. Atualmente, a ferramenta precisa que o usuário acione uma janela para efetuar a busca das informações. Poderiam ser criadas regras associadas às já existentes na base de conhecimento da OdysseySearch, para que no caso dele navegar e não receber nenhuma recomendação de componentes internos, a regra iria acionar a janela de busca automaticamente para o usuário realizar a sua consulta. Nessa janela de consulta também poderiam ter armazenadas as últimas consultas feitas por usuários do mesmo estereótipo, fornecendo sugestões para o usuário realizar sua consulta. Essas regras também poderiam ser ajustadas com o uso da ferramenta para serem as mais eficientes possíveis para os usuários.

Uma grande tendência na integração de sistemas de plataformas de hardware e/ou linguagens de programação diferentes são os WebServices (W3C, 2002). Nesta tecnologia, os sistemas publicam suas interfaces e métodos através da Web para o mundo externo. Cada interface e método funcionam como se fossem serviços disponíveis publicamente, permitindo que outros sistemas externos tenham acesso a informações antes restritas aos sistemas locais.

Um componente participante da arquitetura, o serviço de busca Google, está fornecendo, atualmente, um serviço Web (Lewin, 2002) para acessar sua API através deste. Uma mudança seria alterar a implementação para que o Google fosse acessado através desse serviço publicado. Este serviço acessaria internamente o Google e, provavelmente, teria um tempo de retorno melhor e mais confiável, pois não ocorreriam os problemas comuns de *timeout* na conexão ao site. Este é um problema que ocorre, normalmente, em uma conexão comum, pois existem milhares de usuários concorrendo na busca.

Existem algumas restrições ao WebServices, pois ainda é uma tecnologia em desenvolvimento e que diversos mecanismos necessários ainda não estão maduros o suficiente. Entre eles, a segurança, principalmente a autenticação e autorização, o controle

transacional e a definição de um *workflow* de execução que se destacam como pontos que ainda não estão padronizados.

Referências Bibliográficas

ADELBERG.B., 1998, *"Nodose: A tool for semiautomatically extracting structured and semistructured data from text documents"*, SIGMOD Record, No 27, pp. 283-294.

ARANGO,G., 1988, *Domain Engineering for Software Reuse*, Technical Report UCI-ICS 88-27, Universidade da Califórnia.

AROCENA, G., MENDELZON, A., WebOQL: Restructuring Documents, Databases, and the Web, In: *Proceedings of International Conference of Data Engineering*, 1998, Orlando, Florida, pp 24-33.

ATKINS, D., BIRMIGHAM, W., DURFEE E., et al.,1996, "Toward inquiry-based education through interacting software agents". *IEEE Computer*, No 39, Vol. 5, pp. 69-76.

BAEZA-YATES, R., RIBEIRO-NETO, B., 1999, *Modern Information Retrieval*, Addison Wesley, New York, NY, USA.

BALABANOVIC, M., 1997a, "An adaptive Web page recommendation service". In *Proceedings of the 1st International Conference on Autonomous Agents*, Marina del Rey, California, Fevereiro, pp- 378-385.

BALABANOVIC, M., SHOAM, Y., 1997b, Fab: Content-Based, Colaborative Recommedation. In: Communication of ACM. ,Vol. 40, N° 3, Março, pp. 66-72.

BELKIN, N.J., 1997, User modelling in information retrieval. Tutorial presented at the Sixth International Conference on User Modelling (UM97), Chia Laguna, Sardinia, Itália, Junho. Disponível em: http://www.scils.rutgers.edu/~belkin/um97oh/.

BENAKI et al., 1997, User Modelling in WWW: the UMIE Prototype, In: Proceedings of the Workshop "Adaptive Systems and User Modelling on the World Wide Web" Sixth International Conference on User Modelling, Chia, Laguna, Sardenha, Itália, Junho, pp. 55-57.

BERBENS-LEE, T., et al, 2001, *Semantic Web* at http://www.w3c.org/2001/sw. Acessado em: 20/07/2001.

BICHLER, M., SEGEV, A. ZHAO, J. L., 1998, Component-Based E-Commerce: Assessment of Current Practices and Future Directions, *ACM Sigmod Record: Special Section on Electronic Commerce*, Vol. 27, No. 4, Dezembro, pp. 7-14.

BILLSUS, D. PAZZANI, M., 1996, *Revising User Profiles: The Search for Interesting Web Sites*. In Proceedings of the Third International Workshop on Multistrategy Learning (MSL '96), AAAI Press, pp. 54-61.

BOLLACKER, D., LAWRENCE S., GILES C., 1998, CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications. *2nd International ACM Conference on Autonomous Agents*, pp. 116-123, ACM Press, Maio.

BRAGA, R., 2000, *Busca e Recuperação de Componentes em Ambientes de Reutilização de Software*, Tese de DSc., COPPE/UFRJ, Rio de Janeiro, Brasil.

BRAGA, R., WERNER,C., 1999, Processo de Engenharia de Domínio do Ambiente Odyssey, Relatório Técnico do Projeto Odyssey 4/99, COPPE/UFRJ.

BRAGA, R., WERNER,C., 2000, Desenvolvimento Baseado em Componentes, Minicurso, Anais do XIV Simpósio Brasileiro de Engenharia de Software, João Pessoa, Brasil, Outubro.

BRAGA,R., COSTA,M., WERNER, C., MATTOSO, M., 2000, A Multi-Agent System for Domain Information Discovery and Filtering, Sessão Técnica, Anais do XIV Simpósio Brasileiro de Engenharia de Software, João Pessoa, Brasil, Outubro, pp. 179-194.

BRAGA.R., MATTOSO, M., WERNER, C., 2001, "The Use of Mediation and Ontology Technologies for Software Component Information Retrieval", In: *Proceedings of the Symposium on Software Reusability: Putting Software Reuse in Context*, Maio, Toronto, Ontario, Canada. ACM, pp. 19-28.

BRENNER, W., ZARNEKOW, R., WITTIG, H., 1998, *Intelligents Software Agents – Foundations and Applications*, editors : Springer Verlag, Berlim

BRIN, S., PAGE, L., 1998, The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: *Proceedings of Seventh World Wide Web Conference*, Brisbane, Australia, Abril, pp. 107-117.

BROWN, S.M., SANTOS Jr., E., BANKS, S.B., 1998, "Utility Theory-Based User Models for Intelligent Interface Agents", In: *Proceedings of the 12 Biennial Conf. of the Canadian Society for Computational Studies of Intelligence*, Vancouver, Canada, pp. 379-393.

CADE, 2002, disponível em: www.cade.com.br. Acessado em 20/11/2002.

CASASOLA, E., 1998, *ProFusion Personal Assistant: an agent for the personalizes information filtering on the Web*. Master's thesis, The University of Kansas, Lawrence, KS.

CATHRO, W. 1997. Metadata: An Overview, *Standards Australia Seminar*, August, disponível em: http://www.nla.gov.au/nla/staffpaper/cathro3.html

CHARNY, B. 2000. The World Wide $#@%@$ing Web! ZDNet News. December 2000. http://www.zdnet.com/zdnn/stories/news/0,4586,2667216,00.html

CLARK, D., 2000, "Shopbots Become Agents for Business Change", *IEEE Computer*, Vol. 33, No 2, pp.18-21.

COSTA, M., 2000, Um sistema Multi-Agente para a recuperação de informações de domínio na Internet, *V Workshop de Teses e Dissertações em Engenharia de Software, XIV Simpósio Brasileiro de Engenharia de Software*, João Pessoa, Brasil, Outubro.

DELGADO, J., ISHII, N., "Memory-Based WeightedMajority Prediction for Recommender Systems," ACM SIGIR'99, Workshop on Recommender Systems, 1999. Disponível em: http://citeseer.nj.nec.com/article/delgado99memorybased.html. Acessado em 15/11/2002.

DILCHÉ, J., 2001, *The CRM HandBook: A Business Guide to Customer Relationship Management*, Addison-Wesley Pub Co, 1° Edição.

DIRK, E.,2001 , "Ranking of search results using AltaVista", Disponível Em http://www.ping.be/dirk_van_eylen/avrank.html. Acessado em 20/08/2001.

DUBLIN CORE, 2002, Projeto de Pesquisa Dublin Core, Em: www.dublincore.org. Acessado 15/09/2002.

ETZIONI, O., ZAMIR. O., 1998, Web document clustering: A feasibility demonstration. In *Proceedings of the 21st Annual International ACM SIGIR Conference*, Melbourne, Austrália, Agosto 24-28, pp. 46-54

FOWLER, M., SCOTT, K., 2000, *UML Essencial*, 2a Edição, Bookman, São Paulo.

FSF, 2002, Categories of Free and Non-Free Software, Free Software Foundation, Disponível em: http://www.fsf.org/philosophy/categories.html.

GHERARD Project, 2002, Em: http://www.gerhard.de. Acessado em: 20/09/2002.

GLOVER, E., LAWRENCE, S., GORDON, M., BIRMIGHAM, W., GILES, C., 1999, Recommending Web Documents Based on User Preferences. In: *Proceedings on the SIGIR 99, Workshop on Recommender Systems*, Berkeley, California, Agosto.

GOLGHER P., LAENDER, A., SILVA, A, RIBEIRO-NETO, B., 2000, *"AsByE: Uma Ferramenta Baseada em Exemplos para Especificação de Agentes para Coleta de Documentos Web"*, In: Anais do XV Simpósio Brasileiro de Banco de Dados, João Pessoa, Brasil, Outubro.

GVU, 1998, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, "GVU's WWW User Surveys", Novembro, Disponível em: http://www.cc.gatech.edu/gvu/user_surveys/survey1998_10/. Acessado em: 20/09/2001.

GUDIVADA, V., RAGHAVAN, V., GROSKY, W., KASANAGOTTU, R., 1997, "Information Retrieval on the World Wide Web", *IEEE Internet Computing*, Setembro-Outubro, pp. 58-68.

GLOVER, E., LAWRENCE, S., GORDON, M., BIRMIGHAM, W., GILES, C., 1999, Recommending Web Documents Based on User Preferences. In: *Proceedings oh the SIGIR 99, Workshop on Recommender Systems*, Berkeley, California, Agosto.

GOLGHER P., LAENDER, A., SILVA, A, RIBEIRO-NETO, B., 2000, AsByE: Uma Ferramenta Baseada em Exemplos para Especificação de Agentes para Coleta de Documentos Web", In: Anais do XV Simpósio Brasileiro de Banco de Dados, João Pessoa, Brasil, Outubro.

GVU, 1998, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, "GVU's WWW User Surveys", Novembro, Disponível em: http://www.cc.gatech.edu/gvu/user_surveys/survey1998_10/. Acessado em: 20/10/2001.

GUDIVADA, V., RAGHAVAN, V., GROSKY, W., KASANAGOTTU, R., 1997, "Information Retrieval on the World Wide Web", *IEEE Internet Computing*, Vol. 1, No 5, Setembro-Outubro, pp. 58-68.

HEINEMAN, G., e COUNCIL, B., (editores), 2001, "Component-Based Software Engineering: Putting the pieces together,", Addison-Wesley, June 2001.

HENZINGER, M., HEYDON, A., MITZENMACHER, M., NAJORK, M., 1999, "Measuring Index Quality using Random Walks on the Web", *Proceedings of the 8th International World Wide Web Conference* (WWW8), Toronto, Canada, pp. 213-225.

HOWE, A., DREILINGER, 1999 "SavvySearch: A Meta-Search Engine that learns which Search Engines to Query", *AI Magazine*, Volume 2, No 18, pp. 19-25.

HUBERMAN, B., LUKOSE, R., 1997, "Social dilemmas and Internet congestion", *Science*, 277, 1997, 532-537.

HUBERMAN, B., PIROLLI, P., PITKOW, J., LUKOSE, R., 1998, "Strong regularities in World Wide Web surfing", *Science* 280, Abril, 1998, 95-97.

INQUIRUS, 2002, Serviço de MetaBusca, Disponível em http://inquirus.nj.nec.com/i2/inq2.pl.

ISC, 2002, Internet Software Consortium, Disponível em: http://www.nw.com. Acessado em 20/07/2002.

JACOBSON, I., GRISS, M., JONSSON, P., 1997, *Software Reuse (Architecture, Process and Organization for Business Success),* Addison-Wesley, ACM Press, New York.

JOACHIMS T., FREITAG D., MITCHELL T., 1997, "WebWatcher: A Tour Guide for the World Wide Web"; *Proceedings of Fifteenth International Joint Conference Artificial Intelligence*, Nagoya, Japão, Agosto, pp. 770-777.

KAHLE, B., 1996, "Archiving the Internet". Versão estendida do artigo "Preserving the Internet" que apareceu In: *Scientific American*, March 1997. Versão estendida disponível em: http://www.alexa.com/~brewster/essays/sciam_article.html. Acessada em 15/11/2002.

KERSCHBERG, L., KIM, W., SCIME, A., 2001, "WebSifter II: A Personalizable Meta-Search Agent Based on Semantic Weighted Taxonomy Tree". Em: *The 2001 International Conference on Internet Computing*, IC 2001, Las Vegas, Nevada, Junho, pp. 25-28.

KIM, J., DOUGLAS, O., KATHLEEN, R., 2000, *Using Implicit Feedback for User Modeling in Internet and Intranet Searching*, Relatório Técnico CLIS-TR-00-01, College of Library and Information Services, University of Maryland, Maio. Disponível em: http://www.clis.umd.edu/research/reports/. Acessado em 20/11/2001.

KOBAYASHI, M., TAKEDA, K., 1999, *Information Retrieval on the Web: Selected Topics*, Technical Report, IBM Research, Tokyo Research Laboratory, IBM Japan, Dezembro.

LAM, S., 2001, *The Overview of Web Search Engine*, Technical Report, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, Fevereiro .

LAWRENCE, S., GILES, C., 1999, "Searching the Web: General and Scientific Information Access", *IEEE Communications*, No 37, Vol. 1, pp. 116–122, 1999.

LAWRENCE, S., GILES, C., 1998, *Searching the World Wide Web*, Technical Report, Science, 280, Abril, pp. 98-100.

LAWRENCE, S., 2000, Context in Web Search, *IEEE Data Engineering Bulletin*, Volume 23, Number 3, pp. 25–32.

LE SELECT, 2002, disponível em: http://www-caravel.inria.fr/Eaction_Le_Select.html. Acessado em 10/06/2002.

LEWIN, J., 2002, "*A gaggle of Google Web services*", International Data Group, Novembro. Disponível Em: http://www.idg.net/ic_964156_1794_9-10000.html. Acessado em 20/11/2002.

LIN, I., HUANG, X., CHEN, M., 1999, Capturing User Access Patterns in the Web for Data Mining, *The Eleventh IEEE International Conference Tools with Artificial Intelligence Chicago*, Ilinois, Novembro, pp. 345-348.

LIEBERMAN, H., VAN DYKE, N., VIVACQUA, A., 1999, "Let's Browse: A Collaborative Web Browsing Agent", *Proceedings of the 1999 International Conference on Intelligent User Interfaces*, ACM, Los Angeles, Estados Unidos, Janeiro 5-8, pp. 65-68.

LYCOS, 2002, http://www.lycos.com, Acessado em: 25/11/2002.

MAES, 2000, disponível em http://agents.media.mit.edu/index.html. Acessado em 25/11/2002.

MATTOSO, M., WERNER, C., BRAGA, R., PINHEIRO, R., MURTA, L., ALMEIDA, V., COSTA, M., BEZERRA, E., SOARES, J., RUBERG, N., 2000, "Persistência de Componentes num Ambiente de Reuso", *Caderno de Ferramentas do XV Simpósio de Ferramentas do XV Simpósio de Engenharia de Software (XIV SBES)*, João Pessoa, Brasil, Outubro.

MAURO, R., et al., 1997, GOA++: Tecnología, implementação e extensões aos servidores de gerência de objetos, *Anais do XII Simpósio Brasileiro de Banco de Dados (XII SBBD),* Fortaleza, Brasil, Outubro, pp. 272-286.

MEWS, 2002, First International Workshop on Mining for Enhanced Web Search, Singapura, Dezembro, disponível em http://pipe.cais.ntu.edu.sg:8000/wise2002/mews/default.htm.

MILLER, N., 2000, *A Engenharia de Aplicações no Contexto da Reutilização Baseada em Modelos de Domínio*, Dissertação de MSc., COPPE/UFRJ, Rio de Janeiro, Brasil.

MOURA, A., 2002, *The Semantic Web: fundamentals, technologies, trends*, Minicurso, Anais do XVII Simpósio Brasileiro de Banco de Dados, Gramado, Brasil, Outubro.

MÜELLER, M. E., 1999, "An Inteligent Multi-Agent Architecture for Information retrieval from Internet". In: *Proceeedings 4th. Int. Conf. On the Pratical Applications of Inteligents and Multi-Agents* (PAAM 99), Londres, Inglaterra.

MURTA, L., 2002, CHARON: *Uma Máquina de Processos Extensível Baseada em Agentes Inteligentes*, Dissertação de MSc., COPPE/UFRJ, Rio de Janeiro, Brasil.

NAGAO K., HASIDA, K., "Automatic Text Summarization Based on the Global Document Annotation." In: *Proceedings of the Seventeenth International Conference on Computational Linguistics (COLING-ACL'98)*, Montreal, Canada, Agosto, pp. 917-921.

NICHOLS, D.,M., 1997, Implicit rating and filtering. Em: *Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering*, Budapeste, Hungria, Novembro, European Research Consortium for informatics and mathematics (ERCIM). Disponível em: http://www.ercim.org/publication/ws-proceedings/DELOS5/index.html. Acessado em 10/10/2002.

NIS, 2002, NUA Internet Survey, http://www.nua.ie/surveys/. Acessado em 10/09/2002.

OARD, D. W., KIM, J., 1998, "Implicit Feedback System Recommender" In: *AAAI Workshop on Recommender Systems*, Julho, Madison, Wisconsin, pp 81-83. Disponível em : http://www.glue.umd.edu/~oard/research.html. Acessado em 10/10/2002.

OIL, 2002, disponível em: http://www.ontoknowledge.org/oil/. Acessado em 20/11/2002.

OLIVEIRA, A., 2002, Servidor de Ontologias, dissertação de mestrado em andamento, COPPE/UFRJ, Rio de Janeiro, Brasil.

OMG, 2002, Object Management Group, htttp://www.omg.org. Acessado em: 25/11/2002.

PAEPCKE, A., GARCIA-MOLINA, H., RODRIGUEZ-NULA, G., CHO, J., 1999, Beyond Document Similarity: Understanding Value-Based Search and Browsing Technologies, Disponível em: http://www.cs.stanford.edu. Acessado em 20/08/2002.

PAGE L., BRIN S., MOTWANI R., WINOGRAD, T.,1998, *The PageRank, citation ranking: Bringing order to the Web*. Technical Report, Stanford University, Stanford, CA.

PAZZANI, M., MURAMATSU, J., BILLSUS, D., 1996, Syskill & Webert: identifyng interesting Web sites. In: *Proceedings of 13th International Conference of Artificial Intelligence*, Oregon, Estados Unidos, Agosto, pp. 54-61.

PIRES, P., 1997, *HIMPAR: Uma arquitetura para a interoperabilidade de Objetos Distribuídos*, Dissertação de MSc., COPPE/UFRJ, Rio de Janeiro, Brasil.

PRRETCSHNER, A., GAUCH, S., 1999, *Personalization on the Web*, Technical Report-TR-13591-01, Information and Telecommunication Technology Center, Department of Eletrical Engineering and Computer Science, University of Kansas, Dezembro.

RICH, E., 1983, "Users are Individuals: Individualizing User Models", *International Journal of Man-Machine Studies*, Vol. 18, No 3, pp. 199-214

RIJSBERG, C. J. van, *Information Retrieval.* London:Butterworths, 1979

SALTON, G., 1989, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley.

SALTON, G., McGILL, M., 1983, *Introduction to Modern Information Retrieval*, McGraw-Hill Book Company.

SAMETINGER, J., 1997, *Software Engineering With Reusable Components*, Springer-Verlag, Berlin Heidelberg.

SEARCHENGINE WATCH, 2001 Search Engine Watch, http://www.searchenginewatch.com, Acessado em: 10/11/2001.

SEARCHIQ, 2001, "Best General Use Search Engine & Directories", Disponível Em: http://www.zdnet.com/searchiq/hotlist/individualsearch.html, Acessado em: 20/09/2001.

SHARDANAND, U., MAES, P., 1995, "Social Information Filtering: Algorithms for Automating "Word of Mouth"". *CHI 95: Human Factors in Computing Systems*, Denver, Estados Unidos, Maio, pp. 210-217

SHOE, 2002, disponível em: www.cs.umd.edu/projects/plus/SHOE/. Acessado em: 20/11/2002.

SRIVASTAVA, J., COOLEY, R., DESHPANDE, M., TAN. P-T., 2000, *"Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data"*. In: SIGKDD Explorations, Vol. 1, No 2, http://citeseer.nj.nec.com/srivastava00Web.html.

SEI, 2000, Software Engineering Institute, In: http://www.sei.cmu.edu/domain-engineering/domain_anal.html. Acessado em 10/06/2000.

SELTZER, R., RAY, D., RAY, E., 1997, *The AltaVista Search Revolution,* Berkeley:Osborne/McGraw-Hill.

SESC, 2001, Search Engine Strategies Conferences, San Francisco, Estados Unidos, Agosto 16-17.

SOUZA, R., 2002, *ComPublish: Um sistema para a publicação e recuperação de componentes na Internet*, Dissertação de MSc., COPPE/UFRJ, Rio de Janeiro, Brasil.

SOUZA, R., COSTA, M., BRAGA, R., MATTOSO, M., WERNER, C., 2001, "Reuse Componentes Reuse Thorugh Web Search and Retrieval", *Proceedings of the International Workshop on Information Integration on the Web*, Itaipava, Rio de

Janeiro, Brasil, Abril, pp 12-18. Aceito para publicação no *Journal of Brazilian Computing Science.*

SUN, 2002a, The Source for Java technology. Disponível em: http://java.sun.com. Acessado em 25/11/2002.

SUN, 2002b, The Simple API for XML (SAX) APIs. Disponível em: http://java.sun.com/Webservices/docs/ea2/tutorial/doc/JAXPIntro4.html. Acessado em 25/11/2002.

UDC, 2002, *UDC in a Brief*, Disponível em: http://www.niss.ac.uk/resource-description/udcbrief.html. Acessado em 20/09/2002.

UM, 2003, The 9th International Conference on User Modeling, Pittsburgh, Junho, Disponível em: http://www2.sis.pitt.edu/~um2003/. Acessado em: 20/11/2002.

USENET, 2002, Disponível em: http://www.usenet.org. Acessado em: 25/11/2002.

YAHOO, 2002, www.yahoo.com. Acessado em: 25/11/2002.

W3C, 2002, World Wide Web Consortium, http://www.w3c.org. Acessado em: 25/11/2002.

WERNER, C., COSTA, M., MATTOSO, M., et al, 2000, Odyssey Estágio Atual, *Caderno de Ferramentas do XV Simpósio de Ferramentas do XV Simpósio de Engenharia de Software (XIV SBES)*, João Pessoa, Brasil, Outubro, pp. 366-369.

WIDOM., J., 1999, "Data management for XML: Research directions". *IEEE Data Engineering Bulletin*, Vol. 22 No 3, Setembro, pp. 44-52.

WEB SERVICES, 2002, disponível em: http://www.Webservices.org. Acessado em: 25/11/2002.

WISE, 2002, The 3rd International Conference on Web Information Systems Engineering, Singapura, Dezembro, disponível em http://pipe.cais.ntu.edu.sg:8000/wise2002/. Acessado em: 25/11/2002.

WIEDERHOLD, G., 1992, "Mediators in the Architecture of Future Information Systems"; *IEEE Computer Society Press*, Vol.25, pp. 38-49; Março

WIEDERHOLD, G., GENESERETH, M., 1997, "The Conceptual Basis for Mediation Services"; *IEEE Expert*, Vol. 12, No5, Setembro-Outubro, pp 38-47.

WOOLDRIDGE, M., JENNINGS, R., 1998, editors: Intelligents Agents IV Springer-Verlag Lecture Notes, *AI Volume* 1365, Fevereiro.