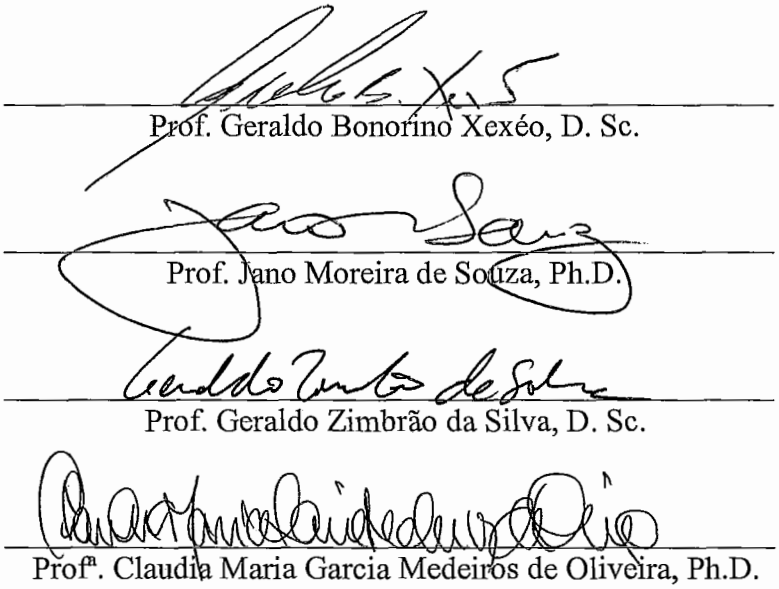
COOPRACTICE – COMUNIDADES DE PRÁTICA VIRTUAIS APOIADAS POR ONTOLOGIAS

Amanda Nascimento Varella

DISSERTAÇÃO 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. Geraldo Bonorino Xexéo, D. Sc.

Prof. Jano Moreira de Souza, Ph.D.

Prof. Geraldo Zimbrão da Silva, D. Sc.

Profª. Claudia Maria Garcia Medeiros de Oliveira, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 2007

VARELLA, AMANDA NASCIMENTO

COOPRACTICE – Comunidades de PráticaVirtuais Apoiadas por Ontologias [Rio de Janeiro] 2007

XIII, 123 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2007)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Comunidades de Prática
2. Ontologias
3. Agentes de Software
4. Busca e Recuperação de Informação

I. COPPE/UFRJ II. Título ( série )

À minha mãe e minha avó que investiram na minha educação

Ao meu marido Vinícios, que me apoiou em todos os momentos

## UM APÓLOGO

Trecho do filme A.I. – Inteligência Artificial, de Steven Spielberg

**David:** Mas, Joe, cadê a fada azul?
**Gigolô Joe:** É o que vamos descobrir,
perguntando ao Dr. Know.
É onde todos encontram
o que precisam saber.
Conheça o bom doutor.
**Dr. Know:** Mentes famintas,
bem-vindas ao Dr. Know...
onde fast-food para a mente
é servida horas por dia...
em 40 mil pontos no país todo!
Pergunte ao Dr. Know,
não há nada que eu não saiba.
**David:** Diga onde posso
encontrar a fada azul!
**Dr. Know:** Pra perguntar, tem de pagar.
Duas por cinco, uma é de graça.
Duas perguntas custam 5 Novapratas,
e a terceira é por conta da casa.
Hoje em dia, nada custa mais
que a informação.
**David:** Só tenho isso.
10 Novapratas e uma moeda de 10...
**Gigolô Joe:** Ele é perspicaz, vai nos levar ao limite.
Mas devemos tentar.
**Dr. Know:** PERGUNTE QUALQUER COISA.
O QUE VOCÊ QUER SABER?
Saudações, colegas! Em oferta,
textos factuais ou de ficção...
primeira ou terceira pessoa,
de nível primário a pós-doutorado.
Estilos vão de contos de fada
a religião.
Quem é quem, onde fica onde,
ou simples fato.
**David:** Simples fato?
**Dr. Know:** Obrigado pela primeira pergunta.
"Simples fato" é um termo
que exige resposta...
**David:** Não pode contar!
Não fiz a pergunta.
**Gigolô Joe:** Cuidado para não levantar
a voz no fim da frase.
**David:** Simples fato.
**Dr. Know:** Você tem mais seis perguntas.
**David:** Onde está a fada azul?
**Dr. Know:** No jardim.
Vascostylis fada azul.
Floresce duas vezes ao ano,
com um ramo de flores azul-claras.
Uma híbrida
da "Ascolameda Arnold".
Você tem mais cinco perguntas.
**David:** Quem é a fada azul?
**Dr. Know:** Você está triste, solitário,
procurando uma amiga?
O serviço de acompanhantes
Fada Azul te achará uma!

**Dr. Know:** Você tem mais quatro perguntas.

**David:** Joe?

**Gigolô Joe:** Tente "conto de fadas".

**David:** Nova categoria.

"Cantos" de fada.

**Gigolô Joe:** Não! Conto de fadas.

**David:** Não. Conto de fadas.

O que é a fada azul?

**Dr. Know:** "Pinóquio", de Carlo Collodi:

"A esse sinal, houve um barulho
como um bater de asas...
e um grande falcão
voou para o peitoril.
'Quais são as suas ordens,
bela fada?'"

**David:** É ela!

**Dr. Know:** "Saiba que a criança
de cabelo azul...
não era ninguém menos
que a bondosa fada...
que havia morado naquela floresta
havia mais de mil..."

**Gigolô Joe:** David! David!

**David:** É ela!

**Gigolô Joe:** Era um exemplo dela.

Mas acho que estamos perto.

**David:** Mas, se o conto de fadas é real, não seria um fato?

Um simples fato?

**Gigolô Joe:** Não fale.

**Gigolô Joe:** Nova categoria, por favor.

Combine...
fato...
com...
conto de fadas.
Agora...
pergunte de novo.

**David:** Como pode a fada azul...
transformar...
um robô...
num menino de verdade?

**Dr. Know:** "Venha, ó criança humana
Para os oceanos e as selvas
Com uma fada, de mãos dadas
Pois o mundo é mais cheio de lamentos
Do que você pode entender
Sua busca será perigosa
Mas a recompensa não tem preço"

## AGRADECIMENTOS

Primeiramente a Deus, por ter me dado saúde e a direção correta para que eu pudesse conquistar meus objetivos, além de ter complementado o meu trabalho, e me recompensando muito além do que eu realmente fiz.

Às minhas queridas mãe e avó que me ensinaram os valores da vida sempre me agraciando com elogios, fazendo com que eu me tornasse uma mulher de fibra assim como elas.

Ao meu marido Vinícios, minha fonte de calma e inspiração. Agradeço-o não só por ser meu marido, mas pelos anos de amizade, amor e dedicação, sem ele, este trabalho não seria possível.

À minha madrinha Beth, que tal qual uma mãe, sempre amiga, vibrou com cada uma das minhas conquistas.

Ao meu orientador Xexéo, que permitiu que minhas idéias fluíssem livremente, me guiando na direção correta.

Ao prof. Blaschek, meu orientador de profissão e criador de oportunidades, que confiou em mim ao me dar um enorme desafio o que fez com que eu crescesse enormemente como pessoa e profissionalmente.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo suporte financeiro a este trabalho.

Aos meus amigos do mestrado, COPPE, faculdade e de todos os tempos de escola.

À dra. Andréia por ter me livrado da maldita TPM.

Ao meu carrinho, velhinho, que agüentou firmemente todos os dias de ida à faculdade e ao trabalho.

Obrigada a todos vocês que de alguma maneira me ajudaram a realizar esta conquista!

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

# COOPRACTICE – COMUNIDADES DE PRÁTICA VIRTUAIS APOIADAS POR ONTOLOGIAS

Amanda Nascimento Varella

Março/2007

Orientadores: Geraldo Bonorino Xexéo
Jano Moreira de Souza

Programa: Engenharia de Sistemas e Computação

Membros de comunidades de prática virtuais trocam uma enorme quantidade de conhecimento diariamente. Entretanto, todo este conhecimento pode ser desperdiçado se não existirem meios efetivos de armazenamento e recuperação destas informações. Informação não contextualizada pode ser informação perdida, uma vez que a mesma informação pode ser descrita de diversas maneiras diferentes. Os princípios da Web semântica sugerem a utilização de ontologias e agentes de software no auxílio à contextualização e recuperação de informações. Nesta dissertação analisamos as principais teorias de comunidades de prática, web semântica e busca de recuperação de informações para propor a criação de uma arquitetura multi-agente, que, apoiada por uma ontologia de domínio será responsável por armazenar, filtrar, contextualizar e recuperar informações relacionadas semanticamente em uma base de conhecimento de uma comunidade de prática virtual.

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

COOPRACTICE – VIRTUAL COMMUNITIES OF PRACTICE SUPPORTED BY ONTOLOGIES

Amanda Nascimento Varella

March/2007

Advisors: Geraldo Bonorino Xexéo
Jano Moreira de Souza

Department: Systems and Computer Engineering

Members of virtual communities of practice exchange a huge amount of knowledge on a daily basis. However, all this knowledge can be wasted if there are no effective ways of storage and retrieval of all this information. Non-contextualized information can be lost information, once the same information can be described in many different ways. Semantic Web principles suggest ontologies and software agents as a solution to the problem of contextualizing and information retrieval. This dissertation analyzes the main theories of communities of practice, semantic web and information retrieval, to create a multi-agent architecture that, supported by ontology, will have the tasks of storage, filtering, contextualization and retrieval of semantically related information in a virtual community of practice knowledge base.

# SUMÁRIO

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABELAS

# Capítulo 1 – Introdução

## 1.1 - Motivação

Não é novidade que a quantidade de conhecimento trafegada e criada diariamente tem crescido exponencialmente nos últimos anos. A Internet fez com que a informação trafegasse a uma velocidade inimaginável há alguns anos atrás. Com a utilização de bandas e Internet cada vez mais velozes, a tendência desta quantidade de tráfego de informação é aumentar exponencialmente.

Já não existem mais fronteiras à transmissão do conhecimento, e esta abundância de informação faz com que sejam necessários mecanismos eficazes de manipulação deste conhecimento. Se os mecanismos de busca de informação não evoluírem com a quantidade de informação, no futuro poderemos estar com uma grande quantidade de conhecimento inacessível, já que fica cada vez mais difícil encontrar a informação exata desejada dentro de um mar de informações cada vez maior.

O conhecimento ainda não é servido em lojas de lanches rápidos, como no filme Inteligência Artificial de Steven Spielberg, entretanto, é fascinante pensar nesta possibilidade. A existência de um "guru" que responda a todas as nossas perguntas, e que tenha informações sobre todas as áreas de conhecimento é uma perspectiva bastante ousada, mas que talvez não esteja tão distante assim de nossos tempos.

Quando pensamos nos primórdios da troca de informação por computador, podemos ir ao fim da década de 70 quando um boletim de informações se tornou bastante popular. As BBS (BBS, 2006) *Bulletin Board System* possibilitavam que usuários se conectassem através de uma linha telefônica uns aos outros, realizando atividades bastante parecidas com as que fazemos hoje como *download* de programas, troca de mensagens, leitura de notícias, publicação de artigos e jogos.

Poucos sabem, mas as BBS tiveram um precursor de nome bastante interessante. A *Community Memory* (CM) foi criada em 1973 (CMEMORY, 2006), por Efrem Lipkin, Mark Szpakowski, e Lee Felsenstein como parte do Community Memory Project (CMP). O CMP foi um experimento para observar como as pessoas reagiriam à troca de informação utilizando computador. Nesta época, poucas pessoas possuíam acesso a computadores. Quando o sistema ficou disponível os usuários

demonstraram que aquela era uma mídia onde se poderia trocar informações sobre arte, literatura, jornalismo, comércio e permitir interações sociais. O primeiro terminal, um teletipo ASR-33 conectado a um computador SDS 940 via telefone(o teletipo era uma espécie de máquina de escrever elétrica que utilizava somente letras maiúsculas, construída com tecnologia de antigas máquinas de *pinball*) foi levado para uma loja de discos em Berkeley, Estados Unidos. Neste terminal as pessoas podiam adicionar itens, atribuir palavras-chave e localizar itens inseridos por outras pessoas através de pesquisas por palavra-chave. A CM da loja de discos atraiu pessoas interessadas em entrar para bandas, bandas procurando por pessoas e até poetas. Algum tempo depois um novo terminal foi instalado em São Francisco. Em 1974 a CM migrou do SDS 940 para os minicomputadores. Em 1975 a CM foi desligada, para que os pesquisadores envolvidos no projeto pudessem se dedicar a novas formas de comunicação via uma rede interconectada de máquinas.

O importante é que a *Community Memory* abriu uma enorme porta para o que temos hoje em troca de informações via meios eletrônicos. Essa porta já nasceu com nome de comunidade, o que é um dos grandes assuntos a serem abordados nesta dissertação.

Desde a *Community Memory* comunidades virtuais vêm produzindo conteúdo valiosíssimo de informação. Entretanto, embora com a evolução dos meios pelos quais estas informações são trocadas, os mecanismos de armazenamento e recuperação de informações produzidas por essas comunidades são a grosso modo bastante semelhantes ao de seus ancestrais. Um participante da comunidade faz uma pergunta, outro participante responde, a informação fica catalogada. Muitas vezes outro participante faz a mesma pergunta, sem nem mesmo verificar se o assunto já havia sido abordado antes. Ao longo do tempo essas comunidades vão criando um grande repositório de conhecimento, peritos no assunto vão se formando através da interação da comunidade, novos membros entram, com questionamentos bem básicos sobre o assunto. O que fazer para facilitar essa comunicação? Como nossas tecnologias atuais podem participar efetivamente do processo de colaboração de uma comunidade, evitando que a mesma pergunta seja respondida inúmeras vezes, e fazendo com que a interação humana seja dedicada a apenas assuntos mais complexos, que exijam reflexões de caráter psicológico e filosófico? E finalmente como tirar das mãos dos membros da comunidade o trabalho árduo de busca por

determinada informação, que em geral seria de fácil localização, mas que não é encontrada simplesmente pela falta de conhecimento inicial do usuário ou porque ele utilizou as palavras erradas em uma busca?

Responder a essas questões é o objetivo desta dissertação. Aproveitar o conhecimento armazenado, agregar estruturas semânticas de maneira a co-relacionar os assuntos presentes na base de conhecimento, para que, através de uma arquitetura de agentes de conhecimento, seja possível agregar, minerar, selecionar e classificar a informação de maneira que os resultados obtidos através de uma consulta sejam os mais próximos possíveis às necessidades do usuário.

## 1.2 - Caracterização do Problema

Definido o objetivo: auxiliar membros de comunidades de prática na busca por informações, precisamos estabelecer quais mecanismos serão utilizados para a realização desses objetivos.

Optando por dar um enfoque semântico às mensagens trocadas pela comunidade, nada mais natural e atual do que abordar o problema sob a perspectiva da Web Semântica (BERNERS-LEE *et al.*, 2001).

O fenômeno da Web Semântica tem sido chamado de Web 3.0, ou a terceira geração da Internet. Na primeira geração a Internet era estática. As pessoas entravam em sítios, baixavam informação.A interação era pouca e o foco estava no sítio em si. Aos poucos a Internet foi ficando mais interativa, e com a explosão de ferramentas onde o usuário é o principal personagem da Web, foi cunhado o nome Web 2.0. Com a Web 2.0 foram popularizados os WebLogs, as comunidades de relacionamento e a grande migração dos aplicativos *desktop* para a Internet. Processadores de texto, planilhas eletrônicas, editores gráficos e até mesmo pequenos sistemas operacionais já dão seus primeiros passos na Internet. Toda essa migração de aplicativos e informações para a grande rede aumentou ainda mais a quantidade de dados trafegada pela Internet. Entretanto, junto com o aumento de informações presentes na rede, surgiu também o fenômeno que poderá ser um dos passos que nos guiarão para a terceira geração da Internet: a pré-disposição dos usuários da Internet em "anotar" a Web. Sítios de fotos estimulam seus usuários a classificarem suas fotos, através da atribuição de etiquetas. Outros convidam seus usuários a compartilharem seus favoritos de Internet, atribuindo etiquetas e informações sobre esses sítios,com ferramentas que se denominam "marcadores sociais". Até mesmo o Google Earth

(GOOGLEEARTH, 2007) (visualização de fotos do mundo via satélite) possui uma ferramenta onde é possível aos usuários ajudarem a "anotar o mundo" escrevendo informações sobre os lugares exibidos. Sem saber, estes usuários estão criando meta-dados sobre as informações da Internet, o primeiro grande passo para a criação de uma Web Semântica.

A Web Semântica de Berners-Lee *et al.* (2001) é baseada em dois grandes pilares: ontologias e agentes de *software.* Com a utilização de ontologias, será possível categorizar o conhecimento de diversas fontes, inferir novas informações a partir de informações existentes e adicionar contexto às informações trocadas. Os agentes de software seriam os responsáveis pela transmissão destas informações, e realização dos trabalhos de filtragem, categorização, busca por novas informações entre inúmeras atividades possíveis.

Baseados nos preceitos de Berners-Lee *et al.* (2001), construímos o arcabouço de nossa dissertação. Uma proposta de busca e recuperação de informações baseada na Web Semântica, no contexto das comunidades de prática.

Partindo dos princípios da Web Semântica tivemos que nos ater aos fatos mais específicos, onde agentes seriam utilizados na nossa arquitetura e o aspecto mais desafiador de todos: como a ontologia seria utilizada de modo a obter resultados significativos de acordo com o nosso objetivo.

Todas estas informações serão detalhadamente discutidas ao longo de nossa dissertação, cuja organização e breve descrição de cada capítulo serão demonstradas na seção seguinte.

## 1.3 - Organização da Dissertação

Nesta dissertação analisaremos os principais fundamentos teóricos para a elaboração de uma arquitetura multi-agente que atuará em uma comunidade de prática com o objetivo de responder a perguntas levantadas pelos membros da comunidade utilizando uma ontologia de domínio que correlacionará conceitos do assunto principal da comunidade.

No **capítulo 1** mostramos as principais motivações para o desenvolvimento deste trabalho, falando sobre as primeiras comunidades virtuais e a necessidade de organização e recuperação do conhecimento produzido por elas. No **capítulo 2** são descritas as principais definições sobre comunidades de prática, passando pelas

comunidades de prática virtuais e analisando os principais aspectos sobre a memória das comunidades de prática. Ao final deste capítulo é feita uma breve descrição sobre as comunidades de linguagens de programação e mais especificamente sobre a linguagem Java, assunto abordado pelas comunidades escolhidas para a construção do protótipo desta dissertação.

No **capítulo 3** abordamos os pilares da Web Semântica, Agentes de Software e Ontologia, sobre os quais baseamos a proposta aqui descrita. Definiremos Agentes de Software e suas principais aplicações. Estes agentes serão responsáveis por catalogar, analisar, classificar e entregar conhecimento, enquanto que a Ontologia será utilizada para categorizar este conhecimento, através do reconhecimento de conceitos e organização de relacionamento entre os conceitos do assunto abordado. Neste capítulo também são demonstradas algumas linguagens de construção de ontologias, e os benefícios que podem ser adquiridos na utilização de ontologias em comunidades de prática virtuais.

No **capítulo 4** analisamos as principais técnicas de busca e recuperação de informação, sobre as quais nos baseamos para a implementação do protótipo. Estas técnicas tiveram que ser extensamente estudadas, para que pudéssemos escolher qual seria o melhor método que se adequaria à nossa proposta. Neste capítulo também foram analisados os principais índices utilizados na avaliação de sistemas de busca e recuperação de informação, além de importantes informações sobre trabalhos consagrados na área, e que foram utilizados como referência à nossa pesquisa.

No **capítulo 5** mostramos todo o processo de construção do COOPRACTICE. Desde a arquitetura de agentes e seus sub-módulos, passando por um detalhamento da construção da ontologia de Java até o funcionamento do sistema como um todo, seus mecanismos de manipulação de consultas e a utilização da ontologia para produzir as respostas que serão retornadas aos membros da comunidade.

No **capítulo 6** descrevemos um experimento que foi realizado com o COOPRACTICE onde onze avaliadores especialistas na linguagem Java analisaram as respostas fornecidas pelo sistema. Neste capítulo são também demonstrados e analisados os resultados deste experimento.

E finalmente no **capítulo 7** concluímos a nossa dissertação, falando sobre os principais objetivos alcançados e as perspectivas de crescimento e expansão da nossa

arquitetura, assim como um panorama geral do que o futuro reserva aos trabalhos da área.

# Capítulo 2 – Comunidades de Prática

Este trabalho possui como foco principal a atuação de um membro virtual em uma comunidade de prática, e não se pode analisar esta atividade sem primeiramente definir o que é uma comunidade de prática. Como ponto inicial é necessário entender como funcionam estas comunidades, quais são seus objetivos e quais são as entidades que participam de comunidades deste tipo.

Neste capítulo serão discutidos alguns conceitos e definições a respeito de comunidades de prática, bem como seus mecanismos de interação e fluxo de informações.

## 2.1 - Definição

O termo "comunidades de prática" é relativamente recente, embora o fenômeno já tenha sua origem há bastante tempo. O conceito tem se mostrado bastante útil, no que diz respeito a conhecimento e aprendizado. Um número crescente de pessoas e organizações em vários setores têm associado às comunidades de prática uma possível chave para proporcionar um aumento em seu desempenho.

Cientistas sociais têm usado versões do conceito de comunidade de prática para uma variedade de propósitos analíticos, mas, a origem e os primeiros usos do conceito têm sido utilizados na teoria do aprendizado. Etienne Wenger juntamente com Jean Lave, criaram o termo enquanto estudavam mecanismos de aprendizagem. Uma vez que o conceito foi articulado, ambos começaram a visualizar a presença destas comunidades em todos os lugares, mesmo onde mecanismos formais de aprendizagem não existiam.

Comunidades de prática existem desde o momento em que os humanos passam a aprender em conjunto. Em casa, na escola, no trabalho, em nossos *hobbies*, todos nós pertencemos a um número razoável de comunidades de prática. Em algumas somos participantes ativos (*core members*) em outras, somos apenas membros periféricos. Ao longo de nossas vidas participamos de uma série de comunidades deste tipo. O fato é que comunidades de prática estão em todo lugar. Elas são uma experiência familiar, tão familiar que talvez escapem à nossa atenção. Entretanto, quando um nome lhes é atribuído e o seu conceito é trazido à tona, ele se transforma

em uma perspectiva, que pode nos ajudar a entender o nosso mundo melhor, e, com isso perceber como extrair conhecimento através do aprendizado informal.

Segundo Wenger (1998) comunidades de prática são conjuntos de pessoas reunidas por um interesse comum em aprender e aplicar práticas comuns de forma espontânea, compartilhando e absorvendo conhecimento e experiência juntamente a outros membros. Comunidades de prática também podem ser caracterizadas como grupos de pessoas que compartilham um conceito ou uma paixão por algo que elas fazem e que interagem regularmente para aprender a fazê-lo melhor.

O conceito de comunidades de prática tem sido encontrado em várias aplicações de negócios, projeto organizacional, governo, educação, associações profissionais, desenvolvimento de projetos e vida cívica. A seguir vemos uma explanação sobre a utilização de comunidades de prática em algumas dessas entidades:

**Organizações** – O conceito tem sido adotado de maneira mais suave por pessoas de negócio, devido ao reconhecimento de que o conhecimento é um bem crítico e que precisa ser gerenciado estrategicamente. Esforços iniciais na gestão de conhecimento focados em sistemas de informação não mostraram resultados satisfatórios (WENGER, 2005). Comunidades de prática puderam mostrar uma nova abordagem, com foco em pessoas e nas estruturas sociais que as habilitam a aprender umas com as outras. Atualmente, dificilmente existe uma organização de tamanho razoável que não tenha alguma iniciativa de comunidades de prática. Um conjunto de características explica este interesse nas comunidades de prática como um veículo para o desenvolvimento de capacidades estratégicas nas organizações. Estas características são:

- Comunidades de prática fazem com o que os praticantes tomem a responsabilidade coletiva por gerenciar o conhecimento de que eles precisam, reconhecendo que, dada a estrutura apropriada, eles estão na melhor posição para fazê-lo.
- Comunidades entre praticantes criam uma ligação direta entre o aprendizado e o desempenho, porque as mesmas pessoas que participam das comunidades participam também das equipes e unidades do negócio.

- Praticantes podem direcionar o aspecto tácito e dinâmico do conhecimento, sua criação (NONAKA e TAKEUCHI, 1995) e compartilhamento, assim como os aspectos mais explícitos.
- Comunidades não estão limitadas por estruturas formais: elas criam conexões entre pessoas, atravessando a organização e fronteiras geográficas.

Partindo desta perspectiva, o conhecimento de uma organização vive em uma constelação de comunidades de prática, cada uma cuidando de um aspecto específico da competência que a organização precisa. Entretanto, as muitas características que fazem uma comunidade de prática uma alternativa adequada para captar conhecimento como: autonomia, orientação ao praticante, informalidade, e transposição de barreiras, são também características que as fazem um desafio para organizações hierárquicas tradicionais.

**Governo** – Assim como nos negócios, organizações governamentais encaram os desafios do conhecimento em crescente complexidade e escala. Eles têm adotado comunidades de prática pelas mesmas razões, embora a formalidade da burocracia possa ir de encontro ao caminho do compartilhamento aberto de conhecimento. Além de comunidades internas, existem problemas típicos do governo como educação, saúde e segurança que requerem a coordenação e compartilhamento de conhecimento entre vários níveis de governo. Há também comunidades de prática que possibilitam a criação de conexões de pessoas entre estruturas formais.

**Educação** – Escolas são na verdade uma forma diferenciada de organização, e elas também tem que encarar crescentes desafios de gerência de conhecimento. As primeiras aplicações de comunidades de prática surgiram no treinamento de professores e na interação com colegas de trabalho. Existe uma grande tendência de interesse nestas atividades de desenvolvimento profissional. Mas, no setor de educação, o aprendizado não é somente um meio para um fim, ele é o produto final. A perspectiva de comunidades de prática é também relevante neste nível. No ambiente de negócios, o enfoque nas comunidades de prática adiciona uma camada de complexidade à organização, mas, não interfere no objetivo final do negócio. Em escolas, mudar a teoria de aprendizado é uma transformação muito mais profunda. Irá

inevitavelmente demorar mais tempo. A perspectiva de comunidades de prática afeta as práticas educacionais em três dimensões:

- Internamente: Como organizar experiências educacionais que escolas aprendem na prática através da participação em comunidades direcionadas a assuntos específicos?
- Externamente: Como conectar a experiência dos estudantes na prática atual através das formas periféricas de participação em comunidades externas através das paredes da escola?
- Durante a vida dos estudantes: Como suprir as necessidades diárias de aprendizado dos estudantes através da organização de comunidades de prática focadas nos tópicos de interesse contínuo dos estudantes além do período inicial de educação?

**Setor Social** – No domínio cívico há um interesse emergente na construção de comunidades entre praticantes. Em um mundo onde o lucro não importa, por exemplo, fundações têm reconhecido que a filantropia precisa dar atenção aos sistemas de aprendizado de maneira a erguer projetos financiados. Mas, praticantes têm procurado conexões entre pares e oportunidades de aprendizado com ou sem o suporte das instituições. Isto inclui o desenvolvimento regional econômico, com comunidades intra-regionais em vários domínios, assim como aprendizado inter-regional com comunidades colhendo informações de praticantes de várias regiões.

**Desenvolvimento Internacional** – Há um crescente reconhecimento de que o desafio em desenvolver nações é muito mais um desafio de conhecimento do que financeiro. Um grande número de pessoas acredita que a abordagem de comunidades de prática pode prover um novo paradigma para o trabalho de desenvolvimento. Sua ênfase está na construção de conhecimento entre os praticantes. Algumas agências de desenvolvimento agora vêem seu papel como os estimuladores de reuniões deste tipo de comunidade ao invés de provedores de conhecimento.

O conceito de comunidades de prática tem influenciado a teoria e a prática em muitos domínios. Partindo do modesto início nos estudos de aprendizado, o conceito passou a ser utilizado por pessoas de negócio interessadas no gerenciamento do conhecimento, e tem progressivamente chamado atenção em outros setores.

Entretanto, é necessário definir as fronteiras que delimitam o conceito de comunidade de prática. Nem tudo que é denominado comunidade é uma comunidade de prática. Uma comunidade de prática está caracterizada por três aspectos (WENGER, 2005):

1 – O domínio: uma comunidade de prática não é meramente um clube de amigos ou uma rede de conexões entre pessoas. Ela tem uma identidade definida por um domínio de interesse compartilhado. Ser um membro de uma comunidade implica um compromisso com o domínio, portanto uma competência compartilhada que distingue os membros de outras pessoas.

2 – A comunidade: ao perseguir seus interesses em seu domínio os membros se envolvem em atividades conjuntas e discussões, ajudam uns aos outros e compartilham informações. Eles constroem relações que os possibilita aprender uns dos outros.

3 – A prática: uma comunidade de prática também não é somente uma comunidade de pessoas que se interessam por filmes, por exemplo. Membros de uma comunidade de prática são praticantes. Eles desenvolvem um repertório compartilhado de recursos: experiências, histórias, ferramentas, maneiras de endereçar problemas recorrentes em uma prática compartilhada. Isto toma tempo e mantém a interação.

Além destes aspectos, comunidades de prática desenvolvem sua prática através de uma variedade de atividades. A seguir são listadas algumas delas, seguidas pelos exemplos típicos de informações trocadas:

- Solução de problemas: alguém possui uma questão a ser resolvida e propõe à comunidade uma discussão em torno do tema em busca de soluções. Ex: "Poderíamos fazer um *brainstorm* para discutir as novas funcionalidades do editor de texto *open source*?".
- Solicitação de informações: o membro pergunta onde ele pode achar determinada informação. Ex: "Onde posso achar o código para conectar ao servidor?".

- Procurando por experiência: Ex: "Alguém já teve problemas ao destruir um componente criado em tempo de execução?".
- Reutilizando recursos: Ex: "Eu possuo um componente de validação de CPF. Eu posso lhe enviar para que você possa acoplá-lo em seu sistema".
- Coordenação e sinergia: Ex: "Poderíamos comprar juntos os livros de C++ junto à editora e assim obtermos um desconto".
- Discussão de desenvolvimentos: Ex: "O que você acha do novo sistema de CAD? Ele realmente ajuda?".
- Documentação de projetos: Ex: "Nós enfrentamos este problema por cinco vezes. Vamos documentá-lo para que quando acontecer de novo possamos resolvê-lo facilmente."
- Mapear conhecimento e identificar *gaps*: Ex: "Quem sabe o que o nosso grupo não possui? Com que outros grupos nós deveríamos nos relacionar?".

## 2.1 - Comunidades Virtuais, Comunidades de Prática Virtuais e Comunidades de Prática Distribuídas

O conceito de comunidades de prática independe do meio na qual elas estão inseridas. Porém, a existência de comunidades de prática mediadas (ao menos em parte) pela Internet é uma realidade, e isto faz com que o conceito se transforme em algo bastante globalizado.

Segundo Reinghold (1993) comunidades virtuais são agregações sociais que emergem da Rede quando existe um número suficiente de pessoas, em discussões suficientemente longas e que envolvam emoções humanas, para formar uma rede de relações pessoais no ciberespaço.

Uma rede de comunidades virtuais relativamente antiga e que possui importância até os dias de hoje é a USENET (USENET, 2005). A USENET é um sistema distribuído de discussão na Internet onde usuários lêem e postam mensagens semelhantes a *e-mails* (chamados artigos) para determinados grupos de notícia (*newsgroups*). Um programa chamado *newsreader* é utilizado para a leitura, controle de mensagens lidas e respostas. Atualmente a maioria dos navegadores e clientes de e-

mail possui um cliente de notícias. Por não possuir mecanismos de controle de usuários (como acontece nas listas de discussão) qualquer um pode postar uma mensagem, e isto acabou fazendo com que a USENET passasse a receber uma grande quantidade de lixo eletrônico e mensagens que fogem ao assunto do grupo. Embora ainda ativa, a USENET abriu caminho a novos mecanismos para troca de mensagens entre grupos de pessoas de um mesmo interesse. A USENET possui um grande significado cultural na grande Rede, por ter popularizado conceitos amplamente reconhecidos e termos como FAQ (FAQ, 2005) e SPAM (mensagens eletrônicas indesejadas).

Da evolução da USENET, surgiu o que conhecemos hoje como os fóruns e as listas de discussão. Os fóruns de discussão (FORUM, 2005) se assemelham bastante ao mecanismo de interação da USENET, onde usuários iniciam tópicos de discussão a serem analisados por outros usuários. Fóruns possuem uma característica estática, onde os usuários escrevem suas mensagens em uma espécie de mural que pode ser lido por todas as outras pessoas, participantes ou não do fórum. Já nas listas de discussão os usuários participantes da lista enviam suas mensagens para um endereço de e-mail (que representa o endereço da lista) e sua mensagem é repassada a todos os endereços de e-mails dos membros da lista (ou grupo). Como exemplos de gerenciadores de grupos de discussão podem ser citados o Yahoo!Groups (YAHOOGROUPS, 2005), Google Groups (GOOGLEGROUPS, 2005) entre outros.

Partindo do conceito de comunidade virtual, facilmente poderíamos entender o significado de uma comunidade de prática virtual, porém, esta definição é bastante controversa, e ainda gera polêmica entre os autores da área. A definição de Lave e Wenger que diz que comunidades de prática são "conjuntos de pessoas reunidas por um interesse comum em aprender e aplicar práticas comuns de forma espontânea, compartilhando e absorvendo conhecimento e experiência juntamente a outros membros" e "grupos de pessoas que compartilham um conceito ou uma paixão por algo que elas fazem e que interagem regularmente para aprender a fazê-lo melhor". As duas suposições não excluem a possibilidade de que estas comunidades possam existir no ambiente virtual, já que o "interagir regularmente" pode ser feito através do ambiente virtual. Porém um dos grandes benefícios adquiridos com a participação em comunidades de prática seria a transmissão de conhecimento tácito (POLANYI, 1983) o conhecimento que é difícil de ser explicitado e armazenado em meios digitais,

necessitando então de uma interação presencial para ser repassado. Um outro fator é a questão da confiança. O conceito de comunidades de prática envolve também características como laços de amizade e confiança, o que faz com que a identidade da comunidade seja maximizada aumentando o interesse de participação por parte dos membros.

Seely Brown e Duguid cunharam a expressão "Redes de prática" (Networks of practice, NoPs) (BROWN e DUGUID, 2000) para descrever grupos de pessoas que estão geograficamente separadas e que podem nunca chegar a se conhecer pessoalmente, mas que compartilham um interesse ou trabalho similar.

Embora por parte de alguns autores haja uma resistência no sentido de aceitar a existência de comunidades de prática virtuais, é notório que abrir mão das vantagens de se ter uma comunidade interligada por meios digitais representaria uma perda muito grande para comunidade. A Internet trouxe inúmeras possibilidades de interconexão entre pessoas diferentes de lugares diferentes e renegar estes benefícios seria o mesmo que andar um passo atrás.

Como o conceito de comunidades de prática virtuais ainda não é algo plenamente consolidado existe uma tendência que sugere que as comunidades podem até existir no mundo virtual, porém não completamente. Além do contato virtual esta abordagem sugere que deva existir uma tentativa de manter um contato social, de maneira que os membros possam trocar conhecimento através de encontros pessoais.

Hildreth et al. (2000) propõe ainda o conceito de comunidade de prática distribuída, e dá como exemplo um grupo de membros de uma empresa, distribuído em vários países, mas parcialmente co-localizado. Esses membros interagem regularmente através de e-mail, vídeo conferência, telefone e outros meios digitais, e duas vezes por ano é feita uma reunião com todos os membros. Os membros reconhecem a importância de uma reunião presencial com o objetivo de criar e manter relacionamentos.

Lueg através de uma análise de comunidades de prática virtuais e comunidades de prática distribuídas tenta fazer uma co-relação entre *newsgroups* e comunidades de prática virtuais e distribuídas, salientado suas similaridades e diferenças. Em Lueg (2000), através de um estudo de caso ele sugere que membros de um *newsgroup* como a USENET podem sim possuir um senso de comunidade. Ele

ainda ressalta que embora as atividades aconteçam no ambiente virtual, o aprendizado e a aplicação deste aprendizado em geral acontecem no mundo real.

Para o objetivo deste trabalho utilizaremos o termo "comunidade de prática virtual", utilizando a análise proposta por Lueg, que sugere que newsgroups (e conseqüentemente listas de discussão) podem ser qualificados como uma comunidade de prática. Neste trabalho não estamos interessados em analisar o aspecto social presente em uma comunidade de prática, mas sim a atividade que pode ser realizada através da mediação por computador, simulando a participação de um membro (agente de sofware) que terá como objetivo responder a questões levantadas na comunidade. Segundo Stahl (2000), a tecnologia de computação, além de conduzir a explosão de informação também tem o potencial de ajudar indivíduos e grupos a aprenderem muito do que eles precisam sob demanda. Na verdade, aplicações na Internet podem ser projetadas para capturar conhecimento da maneira que ele é gerado dentro da comunidade de prática, e fornecer este conhecimento quando ele é útil.

Na próxima seção discutiremos como uma comunidade de prática utiliza o conhecimento gerado ao longo de sua existência.

## 2.2 - Memória de Comunidades de Prática Virtuais: criação, armazenamento e reutilização de conhecimento suportados por computador

Comunidades de prática precisam estar aptas a manter suas próprias memórias. A evolução humana e social pode ser vista como o desenvolvimento sucessivo de formas de memória para aprendizado, armazenamento e compartilhamento de conhecimento. A evolução biológica nos deu memória de fatos e acontecimentos, de imitação e de imaginação. A evolução cultural nos proporcionou memória oral e escrita (externa e compartilhada) e finalmente a evolução da tecnologia moderna gerou memórias digitais (baseadas em computador) e globais (baseadas na Internet). (Donald, Norman *apud* STAHL,2000)

Memórias de comunidades são para comunidades de prática o que memórias humanas são para indivíduos. Elas fazem uso de representações simbólicas explícitas

e externas que possibilitam o aprendizado coletivo dentro de uma comunidade. (Ackerman *et al apud* STAHL, 2000).

A memória de uma comunidade de prática é construída gradualmente, através da interação entre seus membros. Estes membros devem estar integrados uns aos os outros, de maneira que permita que algo que algum membro aprendeu no passado seja disponibilizado a outros membros que possam precisar deste conhecimento no futuro. Um exemplo deste tipo de integração (no nível individual) é o próprio cérebro humano, que armazena um conjunto de memórias durante uma experiência de vida e aprendizado, em uma rede associativa de idéias, que permite recuperação efetiva de acordo com a relevância de determinado assunto.

É desejável que o conteúdo de uma comunidade de prática virtual esteja em artefatos digitais (como um banco de dados), e que este conteúdo possa ser manipulado, de maneira que assim como no cérebro humano possam ser criadas relações entre as idéias e conceitos presentes na memória da comunidade.

A realização de um projeto suportado por uma comunidade é caracterizada por vários ciclos de resolução de problemas e implementações de idéias. Acontece que a habilidade de projetistas para prosseguir com uma dada idéia baseada em sua experiência tácita, é freqüentemente interrompida (*breakdown*), e estes entram em um período caracterizado por "ausência de idéias". Para um dado projetista, é necessário que ele reconstrua sua compreensão da situação através da reflexão explícita (Schon, *apud* STAHL, 2000). Este estado de reflexão pode ser ajudado se o projetista possui o suporte de uma comunidade e de tecnologia, para trazer novas informações relevantes que o ajude a solucionar seus problemas. Uma vez que ele entende o problema e incorpora a nova compreensão em sua memória pessoal, podemos dizer que o projetista aprendeu.

Quando o desenvolvimento de um projeto está apoiado por um contexto colaborativo, a reflexão resulta na articulação de soluções em língua natural ou outras representações simbólicas. O novo conhecimento articulado pode então ser compartilhado dentro da comunidade de prática. Tal conhecimento, criado pela comunidade pode ser usado em situações futuras para ajudar um outro membro a superar períodos de "ausência de idéias" ou de solução de problemas. Este ciclo de colaboração é chamado de aprendizado organizacional, e é representado graficamente através do diagrama sugerido em Stahl (1993) e demonstrado na figura 1.

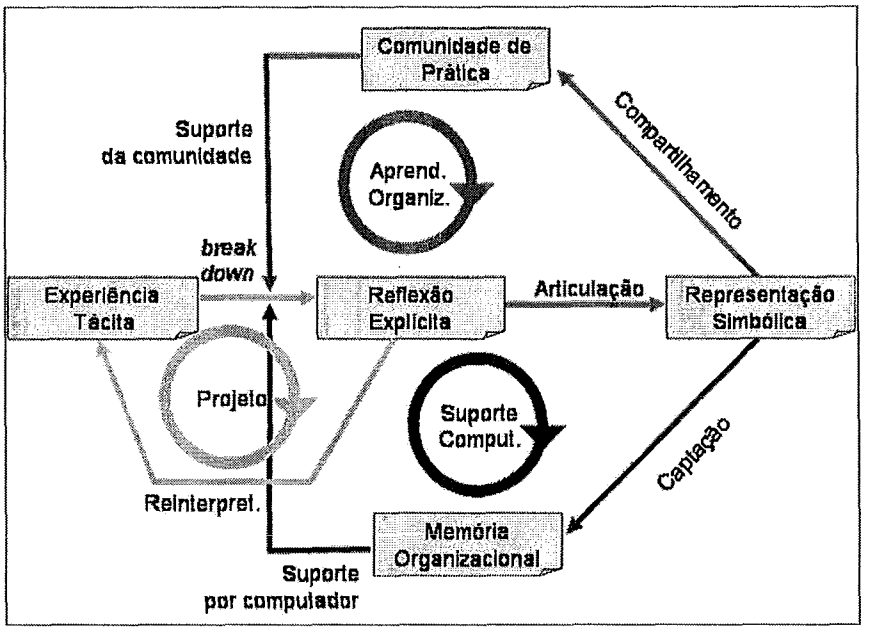



**Figura 1 – Ciclos de projeto, suporte de computador e aprendizado organizacional (STAHL, 1993)**

O aprendizado organizacional pode ser suportado por sistemas de memória organizacional baseados em computador, se o conhecimento articulado puder ser capturado em uma representação digital simbólica. A informação precisa ser armazenada e organizada em um formato que facilite sua subseqüente identificação e recuperação. Com o objetivo de prover suporte baseado em computador, o sistema precisa estar apto a reconhecer situações de "ausência de idéias" ou problemas, quando itens particulares de informação armazenada possam ser úteis para a reflexão humana. (STAHL, 2000).

## 2.3 - Comunidades de Prática e Linguagens de Programação

Grupos de discussão de linguagens de programação representam parte significativa dos grupos presentes na *web*. O aprendizado de linguagens de programação é um processo contínuo. Inicialmente os desenvolvedores de uma determinada linguagem possuem os conhecimentos básicos de estruturas de controle, e comandos comuns a todas às linguagens. Com a utilização intensa da linguagem, à medida que o conhecimento e habilidade em lidar com a linguagem aumentam, surgem também novas dúvidas e problemas. Além disso, o constante desenvolvimento de novas bibliotecas, aspecto dinâmico de uma linguagem, faz com que o usuário da

linguagem procure por comunidades onde ele possa tirar suas dúvidas de como utilizar os recursos destas novas bibliotecas, cuja informação e documentação não se encontram em livro algum.

No início do aprendizado de determinada linguagem de programação, existem basicamente três recursos principais com que o aprendiz pode contar: os livros, que fornecem informações sobre os fundamentos da linguagem, assim como aplicações específicas; a web, que possui um grande número de páginas e artigos falando sobre os mais variados tipos de linguagens de programação; e os grupos de discussão, que provêem uma interação real com pessoas que já passaram pela fase inicial de aprendizado, e que podem oferecer ajuda sob demanda de acordo com as dúvidas do aprendiz. Porém, o papel do aprendiz não é só de absorvedor de informações, colaboração efetiva requer colaboração intensa dos pares participantes. O iniciante pode também trazer várias questões interessantes à comunidade, fazendo com que os membros mais experientes possam pensar em novas soluções para antigos problemas.

Um caso bastante específico de comunidades de prática e linguagens de programação diz respeito a comunidades de desenvolvimento de sistemas *open-source*. Desenvolvedores deste tipo de sistema em geral não recebem por isso. Por este motivo, assim como nas comunidades tradicionais sobre linguagens de programação, os membros de comunidades de desenvolvimento de sistemas *open-source* participam destes projetos com o único objetivo de aprender. Além dos desenvolvedores, usuários comuns do sistema também participam destas comunidades. São pessoas que estão interessadas em saber como o sistema funciona, algumas delas eventualmente acabam se tornando desenvolvedoras. Neste tipo de comunidade também é freqüente a troca de mensagens envolvendo código-fonte e a linguagem do sistema desenvolvido, portanto são também um foco de estudo para o sistema proposto neste trabalho.

Na próxima seção estudaremos um tipo de comunidade de prática específica relacionada a uma linguagem de programação: a linguagem Java.

### 2.3.1 - As Comunidades de Prática da linguagem Java

A linguagem JAVA (JAVA, 1995) surgiu como um sucessor da linguagem Oak. O *Green Project* era um projeto que pretendia se tornar a próxima geração de software embarcado, e foi o pontapé inicial da idéia Java. Entre 1991 e 1992 James

Gosling, Mike Sheridan e Patrick Naughton iniciaram seus trabalhos na linguagem Oak, que futuramente seria acoplada a produtos como a TV interativa. No entanto, a tentativa da Sun em vender esta nova linguagem para equipamentos de TV não levou a bons resultados. Entre 1993 e 1994 a linguagem Oak foi revisada, e proposta como linguagem de programação para Web, rebatizada como Java. O compilador Java originalmente escrito em linguagem C foi então reescrito em Java. Escrever um compilador de uma linguagem nela própria é um marco importante da maturidade da nova linguagem. Em 1995 nasce então a linguagem Java.

A partir deste ponto vários acréscimos à linguagem foram ocorrendo ao longo do tempo. A versão JDK 1.0 alpha lançada em maio de 1995 foi em janeiro 1996 substituída pelo JDK 1.0, seguido pelo JDK 1.1 para Windows e JDK 1.1.3 para Linux e vem evoluindo progressivamente. Paralela a evolução da linguagem foram surgindo novas API's para conexão com banco de dados, implementação de servidores, interface gráfica, ambientes de programação e para suporte a aplicações distribuídas e manipulação de arquivos XML.

A linguagem Java surgiu em um novo paradigma: apoiada pela Internet, nenhuma outra linguagem teve tão rápida expansão e disseminação. Ela surgiu com um forte apelo entre a comunidade de desenvolvedores de software, por ser uma linguagem realmente orientada a objetos, independente de plataforma devido à máquina virtual Java, e de livre distribuição.

Porém, um fator foi determinante no crescimento e popularização da linguagem Java: o surgimento de comunidades cujo tema estava relacionado à utilização, evolução da linguagem, e tudo mais que estivesse relacionado à mesma.

As comunidades de Java ajudaram a disseminar tudo que existia de mais novo em relação à linguagem, suas extensões, API's e como acontece em listas de discussão e fóruns, a troca de informações e geração de conhecimento.

Entretanto, as comunidades de Java possuíam uma característica bastante específica: seus membros eram verdadeiros entusiastas da linguagem, e não participavam desta comunidade apenas pela troca de informações, mas por uma paixão, por algo que os mantinha juntos. Durante toda a evolução da linguagem, desde a sua criação, começaram então a surgir várias comunidades de Java que possuíam estas características. Dentre estas comunidade se destacam os "Java User Groups (JUGs)", que são grupos de pessoas que compartilham um interesse comum

pela tecnologia Java e promovem encontros com freqüência, com o objetivo de compartilhar idéias e informações. Nestes encontros são apresentadas as últimas novidades da linguagem, artigos relacionados, seminários, além de promover o encontro entre membros, o aumento da confiabilidade entre os mesmos e a criação de novas conexões.

Como podemos perceber, relacionando todos esses aspectos pertinentes à comunidades deste tipo, e relembrando os conceitos aqui citados, os Java User Groups constituem realmente as comunidades de prática idealizadas por Wenger, sem ferir em nada os conceitos de: praticantes que compartilham um conceito ou uma paixão por algo que eles fazem, e que interagem regularmente para aprender a fazê-lo melhor.

Ao longo deste trabalho estudaremos que tipo de informação é trocada nas listas de discussão que dão suporte a estas comunidades. Analisaremos como ocorre o fluxo desta informação. Além da atuação de um membro virtual, que por possuir extensa base de conhecimento captada da própria comunidade e de outros meios, buscará por oportunidades de ajuda e colaboração junto aos membros da comunidade.

## 2.4 - Conclusão

Ao longo deste capítulo analisamos os vários conceitos relacionados a comunidades de prática. Mostramos as opiniões dos principais autores da área, tentando fazer uma ligação entre estas opiniões, de maneira que o assunto a ser tratado neste trabalho pudesse ser justificado.

Partimos do conceito mais amplo de comunidades de prática, passando pelo discutido conceito de comunidades de prática virtuais, direcionando as informações seguintes até chegarmos às comunidades que serão o objeto de estudo aqui abordado.

Comunidades de prática são um tópico bastante extenso, e não cabe aqui citar todos os aspectos teóricos relativos a este tema, mas sim às teorias que darão suporte à continuação deste trabalho. Cabe ressaltar que é um assunto bastante em voga nas publicações e literatura da gestão do conhecimento atual. A valorização da interação humana e compartilhamento de informação para geração de novo conhecimento ganha cada vez mais força entre as estratégias mais eficazes de geração e captação de conhecimento.

Quando auxiliadas por tecnologia moderna, conceitos de psicologia e aprendizado, juntamente a mecanismos de extração e agrupamentos de conceitos e

possibilidade de capacidade associativa, o conteúdo produzido por uma comunidade deste tipo passa a se transformar não só em um repositório de conhecimento, mas uma máquina de geração de conhecimento.

Por este motivo tem crescido cada vez mais a produção de softwares de apoio a comunidades, que possuem entre outros objetivos: a potencialização dos mecanismos de interação entre os membros, o gerenciamento da informação trocada na comunidade, a identificação de oportunidades de colaboração na comunidade, etc. Este último é o principal objetivo deste trabalho. Para que possamos alcançar este objetivo, na próxima seção serão estudadas uma série de técnicas, que, tendo como base a comunidade de prática, serão utilizadas para a obtenção de um dos produtos finais deste trabalho: a criação de um agente identificador de oportunidades de colaboração em uma comunidade de prática cujo o tema abordado é a linguagem Java.

# Capítulo 3 – Ontologias e Agentes de Software: dois grandes pilares da Web Semântica

No capítulo 2, definimos os principais conceitos que estão relacionados às comunidades de prática que utilizam mediação por computador. Uma característica principal dessas comunidades é o assunto abordado, a motivação que move essas comunidades. Em geral este assunto ou tópico principal possui características bem definidas, um domínio, que pode ser compartilhado por outras comunidades. Através da mediação por computador, utilizando um agente de software que possua informações sobre este domínio, a reutilização de informação pode ser de grande ajuda a todos os demais membros da comunidade. Isto pode ser feito adicionando semântica ao conteúdo trocado nas listas de discussão ou fórum, através do inter-relacionamento de tópicos semelhantes e conceitos.

Este capítulo tem como objetivo definir os principais conceitos e teorias relacionadas aos esforços de representação semântica de informação trafegada na web, isto inclui: Web Semântica, Ontologias e Agentes, de maneira que a junção dos conceitos de comunidades de prática e semântica de informações possam dar suporte à idéia apresentada neste trabalho.

## 3.1 - Introdução

Sistemas de busca de informações bastante populares e de resultados bastante satisfatórios como o Google utilizam o já consagrado método de busca por palavras-chave. A metodologia é simples, e de bom custo benefício, entretanto nem sempre os resultados são os esperados, já que por trás de informações desejadas residem conceitos, muitas vezes bastante complexos, e que nem sempre podem ser expressos por um conjunto de três ou quatro palavras.

Os seres humanos são em sua essência seres associativos. A todo o momento fazemos associações. Associações de imagens, idéias, textos, sons, cheiros, sensações. Mesmo com tanta informação a qual somos expostos, diariamente aprendemos muito mais do que o somatório destas informações. Isto porque somos capazes de associar

novos conhecimentos com conhecimentos já existentes, criando assim um novo conhecimento (LANDAUER e DUMAIS, 1997).

Embora em estágio bastante avançado, ainda são grandes os esforços em reproduzir computacionalmente aspectos inerentes à essência humana, isto inclui a capacidade associativa, indutiva, cognitiva e de comunicação via uma língua natural.

Algoritmos de Inteligência Artificial vêm alcançando este objetivo, e em um futuro próximo teremos o comportamento e ações de artefatos eletrônicos bastante semelhantes aos do comportamento humano.

Uma outra grande área de estudos que tem direcionado seus esforços na compreensão de informação, com o objetivo de aumentar a capacidade associativa de máquinas é a de criação de semântica sobre os dados e informações.

A Web é um grande repositório de informações, porém muitas das vezes são de difícil acesso, já que grande parte dessas informações não pode ser trabalhada de maneira automática. Ao mesmo tempo, não dispomos de tempo para procurar por todas as informações que estariam relacionadas ao assunto desejado.

Partindo da junção desses dois tópicos, surgiu a idéia da Web Semântica (BERNERS-LEE *et al.*, 2001). A proposta não tem como objetivo criar uma nova Web, mas uma extensão da Web atual, onde a informação seja fornecida associada a um significado.

A proposta da Web Semântica está consolidada em três grandes conceitos, que são: representação do conhecimento, ontologias e agentes de software.

O foco deste trabalho está na utilização de Ontologias e Agentes de software. Daremos uma breve introdução sobre a representação do conhecimento proposta pela Web Semântica, entretanto não nos estenderemos sobre este assunto, já que para a realização do objetivo deste trabalho não será necessária a análise de estruturação de informação da web. Entretanto, é importante que façamos uma análise da representação do conhecimento em geral, pois esta será útil no estabelecimento dos mecanismos de representação do conhecimento trocado pelas comunidades de prática.

Nas próximas seções daremos uma breve introdução sobre os objetivos da Web Semântica, explicando em detalhes as definições dos seus componentes principais. A seção de ontologias é bastante extensa, pois visa a rever e analisar todos os conceitos presentes no projeto e utilização de ontologias, bem como a sua utilização associada a comunidades de prática. Finalmente concluímos com um tópico

sobre agentes de software, que visa a explicitar os principais mecanismos pelos quais estes podem atuar sobre a informação disponível.

## 3.2 - Web Semântica

De acordo com Berners-Lee *et al.* (2001) a Web Semântica é uma extensão da Web atual, onde a informação é dada juntamente a um significado bem definido, possibilitando assim que computadores e pessoas possam trabalhar em cooperação. Segundo Berners-Lee, os primeiros passos no sentido de estender a Web atual para a Web Semântica já estão a caminho. Em um futuro próximo estes desenvolvimentos culminarão em significativas novas funcionalidades, ao mesmo tempo em que as máquinas estarão muito mais aptas a processarem e "entenderem" os dados que elas meramente exibem atualmente.

O objetivo principal da Web Semântica é a capacidade de representar e gerenciar o conteúdo semântico da Web.

## 3.3 - Representação do conhecimento

Para que a Web Semântica funcione, os computadores devem ter acesso a coleções estruturadas de informações e conjuntos de regras de inferência que possam ser utilizadas de maneira a conduzir uma linha de raciocínio automática. Essas regras fazem parte de um esforço de representação do conhecimento.

Pesquisadores da área de inteligência artificial têm estudado sistemas de representação do conhecimento desde que a Web foi desenvolvida. A representação do conhecimento, está em um estado que atualmente é comparado ao hipertexto antes da Web: não há dúvidas de que é uma boa idéia, e existem boas demonstrações disso, porém, ainda não foi capaz de mudar o mundo. Contém as sementes de aplicações importantes, mas, para que o seu potencial seja mostrado por completo, deve estar ligado a um sistema global. O problema relacionado a sistemas de representação de conhecimento está no fato de que usualmente o número de questões que esses sistemas podem responder de maneira confiável está limitado. Assim, alguns sistemas tentam resolver estas questões adicionando seu próprio conjunto de regras para fazer inferências sobre seus dados. A grande dificuldade é: com regras tão específicas,

mesmo transportando-se o conjunto de dados de um sistema para o outro, nem sempre as regras podem ser validadas.

Os pesquisadores da área de Web Semântica aceitam que exista m questões que não podem ser resolvidas, e que estas questões são um preço a pagar em função da versatilidade. A linguagem para expressar as regras é tão expressiva quanto desejarmos de maneira que permita que a Web "raciocine" tão amplamente quanto desejado. O desafio da Web Semântica é semelhante ao da Web tradicional. Há algum tempo atrás muitos disseram que a Web jamais seria uma biblioteca organizada, sem uma árvore ou banco de dados central, e ninguém teria certeza de que acharia a informação desejada. De certa forma o pensamento estava correto, no entanto a evolução dos sistemas de informação fez com que os grandes sistemas de busca (impraticáveis há alguns anos atrás) pudessem criar índices completos de grande parte do material existente na grande rede. Assim, o grande objetivo da Web semântica é prover uma linguagem que expresse tanto os dados como as regras de inferência sobre os mesmos, e que permita que regras de qualquer representação de conhecimento existente possam ser exportadas para a Web (BERNERS-LEE *et al.*, 2001).

## 3.4 - Ontologias

Ontologias são a proposta da Web semântica para a organização do conhecimento existente. Segundo a Filosofia, Ontologia é o estudo da existência do ser. Em Inteligência Artificial, uma Ontologia pode ser definida como "uma especificação formal e explícita de uma conceitualização compartilhada" (GRUBER, 1993). A palavra "conceitualização" refere-se a uma abstração, visão simplificada do mundo que desejamos representar para algum propósito, construído através da identificação dos conceitos e relações relevantes. O termo "explícita" indica que os tipos de conceitos e as restrições ao seu uso são explicitamente definidos. "Formal" significa que a ontologia deve ser compreensível por um computador, sendo necessário que ela seja representável matematicamente, de forma não-ambígua. Finalmente, "compartilhada" implica que o conhecimento representado é consensual, aceito por um grupo como um todo e não apenas por um único indivíduo. Basicamente, uma ontologia é o vocabulário comum utilizado para representar um certo domínio do conhecimento e a conceituação que estes termos pretendem capturar.

Em geral, o tipo de ontologia mais comum para a Web possui uma taxonomia e um conjunto de regras de inferência. A taxonomia define classes de objetos e relações entre eles. Classes, subclasses e relações entre entidades são uma ferramenta muito poderosa, pois com elas podemos expressar um grande número de relações entre entidades atribuindo propriedades a classes e permitindo que subclasses herdem estas propriedades (BERNERS-LEE *et al.*, 2001). As regras de inferência adicionam poder de processamento desta informação pois a ontologia expressa uma regra e o programa, ou agente, pode, a partir daí, utilizando os dados da ontologia, tirar conclusões a partir das associações presentes nas regras.

### 3.4.1 - Construção de Ontologias

Embora ontologias sejam a base da Web semântica, não é somente nela que sua utilização pode ser aplicada. Existe uma infinidade de ambientes e situações onde a criação de ontologias seria extremamente útil. A pergunta que em geral é feita é: porque alguém precisaria de uma ontologia? Algumas respostas a esta pergunta são (NOY e MCGUINESS, 2001):

- Para compartilhar entendimento comum da estrutura de informação entre pessoas ou agentes de software: por exemplo, se *sites* médicos utilizam a mesma ontologia como base, agentes de software podem utilizar esta informação de maneira a responder consultas de usuários.
- Para possibilitar a reutilização de conhecimento de domínio: se for necessário construir uma ontologia mais geral, ontologias mais específicas já construídas anteriormente podem ser integradas de maneira a ajudar a compor a ontologia mais geral.
- Para fazer suposições de domínio explícitas: a especificação de conceitos básicos faz com que iniciantes no domínio da ontologia possam ter mais familiarização com os termos utilizados. Ex: aprendizado de linguagens de programação.
- Para separar o conhecimento de domínio do conhecimento operacional: uma ontologia sobre medicamentos servirá para qualquer farmácia, entretanto, cada farmácia pode ter sua própria ontologia, que descreverá seu funcionamento particular e suas regras de negócio.

- Para analisar o conhecimento do domínio: uma vez que termos de um domínio estão explicitamente declarados, é possível fazer uma análise formal destes termos.

Na maioria das iniciativas de construção de uma ontologia, a ontologia não é um objetivo em si. Definir uma ontologia está relacionado a definir um conjunto de dados e sua estrutura, que devem estar preparados para serem usados por outras aplicações.

Ontologias são constituídas por uma descrição formal e explícita de um conceito em um domínio de discurso (classes, muitas das vezes denominadas somente como conceitos), propriedades de cada conceito descrevendo várias características e atributos do conceito (*slots*, papéis, propriedades ou atributos) e restrições sobre esses atributos (*facets*). Uma ontologia junto com um conjunto de instâncias individuais de classes constituem uma base de conhecimento. Na verdade existe uma tênue fronteira, onde a ontologia termina e começa a base de conhecimento. Sendo assim, a construção de uma ontologia pode ser simplificada nos seguintes passos:

- Definir classes na ontologia
- Organizar as classes em uma hierarquia (subclasse – superclasse)
- Definir propriedades e descrever os valores permitidos para estas propriedades
- Preencher os valores dessas propriedades para as instâncias

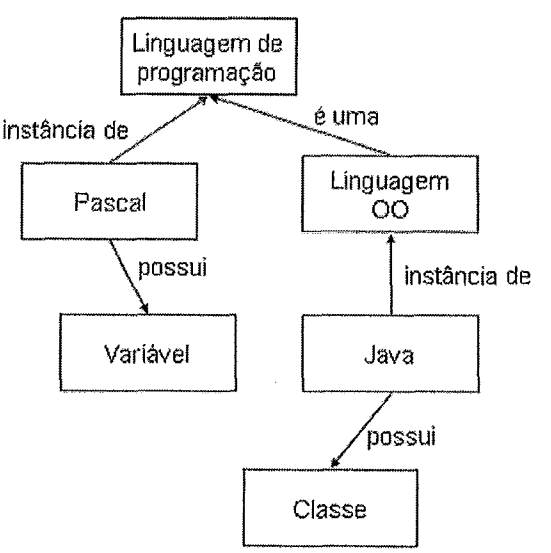



**Figura 2 – Exemplo de uma ontologia com classes e instâncias**

Na figura 2 é possível reconhecer uma série assertivas que podem ser feitas em relação ao domínio exemplificado que tem como raiz principal linguagens de programação. As caixas pretas representam classes, enquanto que as caixas vermelhas representam instâncias de classes. As setas são as propriedades das classes (ou instâncias) que serão responsáveis pelo estabelecimento da relação entre as classes. Podemos derivar desta ontologia as seguintes conclusões:

- Pascal e linguagens orientadas a objeto são linguagens de programação
- Como Java é uma linguagem orientada a objeto, também é uma linguagem de programação.
- Como Pascal é uma linguagem de programação, e Pascal tem variáveis, então qualquer linguagem de programação orientada a objeto, por ser também uma linguagem de programação, terá variáveis.
- Sendo Java uma linguagem de programação orientada a objeto, Java também terá variáveis.
- Toda linguagem orientada a objeto tem classes. Porém, não podemos afirmar que toda linguagem de programação possua classes.

Este é um exemplo bastante interessante do poder de expressividade de uma ontologia. Inferências bastante complexas podem ser feitas utilizando-se a representação de conceitos e seus relacionamentos.

Ontologias são uma ferramenta bastante poderosa para a descrição do mundo que nos rodeia. Podemos utilizar os termos fornecidos pelo domínio da ontologia para afirmar proposições específicas sobre o domínio ou uma situação em um domínio. Uma vez que tenhamos a base para essas proposições, poderemos também representar o conhecimento envolvido em atitudes proposicionais como: hipótese, crença, esperança, desejo e temor. Como exemplo podemos citar uma ontologia médica, que defina conceitos sobre proposições relacionadas ao diagnóstico de doenças. Assim, uma ontologia pode representar objetivos, crenças, hipóteses e predições sobre um domínio, além de meramente representar fatos.

O grande poder de expressividade das ontologias pode representar um problema no momento da construção. Isto porque pode se tornar bastante difícil encontrar o escopo de atuação de uma ontologia.

Toda ontologia deve possuir um domínio, e este domínio deve ser o primeiro aspecto a ser estabelecido pelo engenheiro da ontologia. Após estabelecer o domínio, o projetista deve concentrar seus esforços em determinar o escopo desta ontologia. Em geral, quando alguém opta por construir uma ontologia, é porque possui alguma tarefa em mente, para a qual a utilização da ontologia irá ajudar a organizar informações, fazer inferências, responder perguntas, entre outras funções. Sendo assim, podemos levantar a seguinte questão: de que maneira a ontologia a ser construída depende da tarefa que queremos realizar? Podemos concluir facilmente que as coisas que existem no mundo atualmente independem de nossos objetivos. Olhando por esta perspectiva ontologias não são dependentes de tarefas. Por outro lado, os aspectos da realidade a serem escolhidos para fazer parte da ontologia dependem de quais tarefas desejamos realizar.

Com o objetivo de definir o domínio e o escopo de uma ontologia, o projetista pode levantar as seguintes questões (NOY e MCGUINESS, 2001):

- Que domínio a ontologia deverá cobrir?
- Para que a ontologia será utilizada?
- Quais tipos de questões a ontologia deve responder?
- Quem irá usar e manter a ontologia?

A resposta a essas questões são um bom indicativo da delimitação das fronteiras da ontologia, e podem ajudar o projetista a estabelecer os limites da sua base de conhecimento de acordo com o objetivo desejado.

Dado um domínio, a ontologia será o coração de qualquer sistema de representação de conhecimento daquele domínio. Sem ontologias, ou conceitualizações que estejam por baixo do conhecimento, não haverá vocabulário suficiente para representar o conhecimento. Desta maneira, o primeiro passo na idealização de um sistema efetivo de representação do conhecimento, e vocabulário, é realizar uma análise ontológica efetiva do campo de atuação da ontologia. Análises fracas fatalmente culminarão em bases de conhecimento inconsistentes.

Uma vez definido o domínio e o escopo, poderá ter início o processo de construção da ontologia. Entretanto, ainda existe um aspecto a ser considerado: a possibilidade de reuso de ontologias existentes. Assim como nas teorias de reutilização de componentes, ontologias podem ser reaproveitadas e compartilhadas.

Por este motivo é muito importante que ao desenvolver a ontologia, se possível, o projetista mantenha os conceitos concretos e consolidados separados do aspecto operacional. Por exemplo, podemos citar ontologias relacionadas ao funcionamento do restaurante fictício "Coma Bem". Ontologias sobre comidas, e bebidas, por serem algo tão trivial certamente já foram criadas por alguém, e possivelmente poderão ser encontradas em alguma página Web. Para construir a ontologia do restaurante Coma Bem, estas ontologias podem ser reutilizadas de maneira a compor a ontologia mais abrangente. Ontologias sobre comidas e bebidas podem ser compartilhadas por todos os restaurantes, não só restaurantes como hotéis, entre outros usos. O restaurante Coma Bem poderia também utilizar uma ontologia básica sobre restaurantes, com aspectos que são comuns a todos os restaurantes. Espera-se que todos os restaurantes sirvam comida e bebida, e que os clientes paguem por isso, essas são características que estão presentes em todos os estabelecimentos deste tipo. Entretanto, a ontologia que define os processos internos do Coma Bem são aspectos bastante particulares do restaurante, e podem ser mantidas em uma ontologia à parte, por exemplo, que defina as regras de negócio do restaurante Coma bem. A reutilização do conhecimento pode eliminar a necessidade de replicar o processo de análise de conhecimento.

### 3.4.2 - Aspectos Práticos da Construção de Ontologias: Linguagens Descritivas para a Criação de Ontologias

Independente da linguagem utilizada, algumas suposições são comuns a todo projeto de ontologias, sendo assim, linguagens robustas de ontologias devem ser capazes de expressar as seguintes suposições (CHANDRASEKARAN *et al.*, 1999)

- Existem *objetos* no mundo
- Objetos possuem *propriedades* ou *atributos* que podem receber *valores*
- Objetos podem figurar em várias *relações* com outros objetos
- Propriedades e relações podem mudar com o *tempo*
- Existem *eventos* que podem ocorrer em diferentes *instantes de tempo*
- Existem *processos* dos quais os objetos participam, e que podem ocorrer durante um tempo determinado
- O mundo e seus objetos podem estar em diferentes *estados*

- Eventos podem *causar* outros eventos ou *mudança de estados* como efeitos
- Objetos podem possuir *partes*

Para que uma ontologia possa atender ao seu real objetivo, isto é, poder ser compreendida tanto por homens quanto por máquinas, foram sugeridos vários tipos de padronização para representação de ontologias. Atualmente destacam-se os padrões propostos pelo W3C (RDF, RDFS, OWL) e as linguagens propostas pela DARPA (DAML, DAML+OIL) e pelo grupo OntoKnowledge (OIL). Algumas possuem o objetivo apenas de acrescentar semântica em conjuntos de dados, enquanto outras são realmente linguagens de descrição de ontologias. Muitas das vezes essas linguagens são utilizadas em conjunto, de maneira a obter maior expressividade na ontologia. A seguir uma breve descrição de cada uma dessas linguagens:

**RDF e RDFS** – RDF *(Resource Description Framework)* (W3C, 2005a) é uma linguagem para representar informações sobre recursos na World Wide Web, mais especificamente, representar metadados sobre recursos Web. Podemos entender recursos Web como informações sobre páginas e documentos da Internet, recursos compartilhados, e até informações que podem ser identificadas na Web, mesmo que não sejam diretamente recuperadas por meio dela, como por exemplo, informações de perfis de navegação de usuários. RDF provê uma estrutura comum para expressar informações de forma que possam ser trocadas entre aplicações sem perda de significado. RDF em si não é uma linguagem para representação de ontologias. Porém, provê suporte para descrição de ontologias, concomitantemente com outras linguagens como DAML, OIL e OWL. O RDFS (RDF *Schema* ou RDF *Vocabulary Description Language*) (W3C, 2004d) está para o RDF assim como o XML Schema (W3C, 2005b) está para o XML (W3C, 2005c). Ele funcionará como um descritor de tipos para o RDF. O RDFS fornece facilidades para a descrição de propriedades e classes e relações de dependência entre as mesmas, sendo bastante análogo à filosofia das linguagens orientadas a objetos.

**DAML** - (DARPA *Agent Markup Language*) (DARPA, 2005) é uma linguagem desenvolvida como uma extensão do XML e do RDF, que fornece um conjunto de construções para a criação de ontologias e marcação (*markups*) de

informações de forma que sejam legíveis e compreensíveis por máquinas. Tem como objetivo fornecer linguagens e ferramentas para possibilitar o desenvolvimento da Web Semântica. Se comparado ao XML em sua forma mais simples, um conjunto de expressões DAML (e a especificação DAML) pode permitir inferir outra expressão DAML. Em XML não é possível executar tal tarefa. DAML oferece características de descrição de ontologias, como propriedades padrão, por exemplo, equivalência, significando que uma propriedade em uma língua terá o mesmo sentido semântico que em outra língua. DAML fornece também propriedades particulares como por exemplo um número de identificação que caracterizará exclusivamente um indivíduo.

**OIL** - (*Ontology Inference Layer*) (ONTOKNOWLEDGE, 2000) é uma proposta de uma camada de representação e inferência de ontologias baseado na Web. Ela é uma linguagem fundamentada em RDF/RDF Schema e XML/XML Schema, incluindo semântica para descrição de significados de termos. A arquitetura OIL permite camadas ("também conhecidas como dialetos") para descrição de ontologias com níveis variados de funcionalidade e complexidade, onde a compreensão de ontologias descritas em camadas inferiores permite a compreensão parcial de ontologias descritas em camadas superiores.

**DAML+OIL** – DAML+OIL (DARPA, 2002) é uma linguagem de marcação semântica para recursos Web. É uma linguagem derivada das linguagens DAML e OIL, e, portanto, também construída sobre os padrões RDF/RDF Schema. DAML+OIL permite expressar propriedades e classificações ainda mais sofisticadas de recursos que o RDFS. Junção DAML + OIL provê uma série de funcionalidades adicionais a de suas linguagens antecessoras, entre elas: facilidade na definição de propriedades, aumento dos recursos de tipagens de dados, possibilidade da definição de propriedades únicas, definição de coleções, definição de classes disjuntas, possibilidade do estabelecimento de propriedades como uniões disjuntas, transitividade e propriedades inversas e restrições sobre propriedades.

Por ser a linguagem de construção de ontologia escolhida neste trabalho, a OWL será tratada na seção a seguir, com o objetivo de explicitar melhor os conceitos básicos referentes à utilização desta linguagem, bem como os recursos utilizados na elaboração da ontologia aqui apresentada.

### 3.4.3 - OWL

A iniciativa da Web Semântica propõe a utilização do RDF para a representação dos dados. Além do RDF, a Web Semântica necessita também de uma linguagem que possa representar ontologias, para que os significados das terminologias utilizadas nos documentos possam ser descritos formalmente. A OWL (*Web Ontology Language*) (W3C, 2004a) proposta pelo W3C é grande candidata à linguagem padrão para definição de ontologias da Web Semântica.

A OWL é uma linguagem de marcação semântica, para publicar e compartilhar ontologia na Web. Ela foi desenvolvida como uma extensão do vocabulário RDF e é derivada da linguagem anteriormente descrita DAML + OIL (W3C, 2004c). A OWL tem como objetivo fornecer uma linguagem, que pode ser utilizada para descrever classes e relacionamentos entre essas classes. Possui um poder de expressividade bastante grande, capaz de representar relacionamentos complexos do mundo real.

De acordo com o critério de expressividade, a OWL é subdividida em três sublinguagens, onde cada uma é uma extensão da anterior (W3C, 2004b):

- OWL *Lite*: destinada àqueles que desejam somente uma classificação de hierarquias e descrição de restrições. Como exemplo podemos citar o fato de que, embora OWL Lite permita a utilização de restrições de cardinalidade, esta cardinalidade só pode ser 0 ou 1. É perfeitamente possível que um sistema de *reasoning* possa facilmente implementar todas as características do OWL Lite.
- OWL DL: destinada aos usuários que querem o máximo de expressividade, porém sem perder a completude computacional (garantia de que todas as expressões podem ser computadas) e poder de decisão em tempo finito. OWL DL possui este nome em referência à Lógica Descritiva (BAADER *et al.*, 2003), um campo de pesquisa que estuda os aspectos de decisão da lógica de primeira ordem. OWL DL possui propriedades computacionais bastante razoáveis para sistemas de *reasoning*.

- OWL *Full*: para aqueles que querem o máximo de expressividade liberdade sintática com o RDF, sem garantias computacionais. Um sistema de *reasoning* não será capaz de implementar todas as características do OWL Full.

A seguir um exemplo de um trecho de uma ontologia escrita em OWL:

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.owl-ontologies.com/unnamed.owl#"
  xml:base="http://www.owl-ontologies.com/unnamed.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="LinguagemOO"/>
  <owl:Class rdf:ID="Variavel"/>
  <owl:Class rdf:ID="LinguagemProgramacao">
    <rdfs:subClassOf rdf:resource="#LinguagemOO"/>
  </owl:Class>
  <owl:ObjectProperty rdf:ID="possui">
    <rdfs:domain rdf:resource="#LinguagemOO"/>
    <rdfs:range rdf:resource="#Variavel"/>
  </owl:ObjectProperty>
</rdf:RDF>
```

**Figura 3 – Exemplo de um trecho de ontologia escrito em OWL**

Na figura 3, através da propriedade "owl:Class" são declaradas três classes: LinguagemOO, Variavel e LinguagemProgramacao. A propriedade "owl:subClassOf" aplicada à classe LinguagemProgramacao, diz que a classe LinguagemProgramacao é uma sub-classe da classe LinguagemOO. E finalmente a propriedade "owl:ObjectProperty" define o atributo "possui" que relaciona as classes: LinguagemOO e Variavel, informando assim, que LinguagemOO possui Variavel.

### 3.4.4 - Utilizando Ontologias

Muitos trabalhos falam sobre teorias de construção de ontologias, e até sugerem a criação de ontologias para determinadas aplicações. Entretanto, são poucos os que mostram todos os aspectos, desde a construção de uma ontologia até a sua efetiva utilização associada à aplicação em questão, de maneira a possibilitar um processamento adicional da informação. Grande parte dos trabalhos existentes explicita a aplicação, associa ao uso da ontologia, entretanto, a maneira como a ontologia será utilizada na aplicação é na maioria das vezes nebulosa.

Uma ontologia, se bem projetada, e associada de maneira adequada ao propósito da aplicação pode fazer com que a qualidade dos resultados obtidos pela aplicação seja bastante maximizada. Não é à toa que nem sempre os trabalhos realizados com ontologias deixam claro como essas ontologias foram utilizadas. Nem sempre é fácil encontrar um algoritmo genérico que possa percorrer esta ontologia, de maneira a encontrar as respostas desejadas, para qualquer tipo de pergunta, ou informação que se deseja obter sobre o domínio em questão. Neste momento, a pergunta "para que a ontologia será utilizada?" deve assumir grande importância no projeto, pois é ela que definirá de que maneira os conceitos deverão ser agrupados de maneira que a ontologia possa corresponder ao objetivo desejado.

Em (MIZOGUCHI e IKEDA, 1996) são propostos oito níveis de utilização de ontologias:

- Nível 1: Utilizada como um vocabulário comum para comunicação entre agentes distribuídos
- Nível 2: Utilizada como um esquema conceitual de um banco de dados relacional. Informações estruturais de conceitos e relacionamentos entre os mesmos são utilizadas. A conceitualização em um banco de dados é nada mais que o esquema conceitual. A recuperação de dados em um banco de dados é facilmente realizada quando há um acordo com o esquema conceitual.
- Nível 3: Utilizada como pano de fundo para o usuário de uma certa base de conhecimento.
- Nível 4: Utilizada para responder questões de competência
- Nível 5: Padronização
    - 5.1 Padronização de terminologia (no mesmo nível do nível 1)
    - 5.2 Padronização do significado de conceitos
    - 5.3 Padronização de componentes de objetos-alvo (ontologia de domínio)
    - 5.4 Padronização de componentes de tarefas (ontologia de tarefa)
- Nível 6: Utilizada para transformação de bases de dados, considerando as diferenças do significado do esquema conceitual.

Isto requer não somente a transformação estrutural mas também a transformação semântica.

- Nível 7: Utilizada para reutilizar conhecimento de uma base de conhecimento utilizando recuperação de informação (*information retrieval*)
- Nível 8: Utilizada para reorganizar uma base de conhecimento baseada em recuperação de informação.

### 3.4.5 - Ontologias e Comunidades de Prática

Segundo a definição sobre comunidades de prática dada no capítulo 2, que diz que "comunidades de prática podem ser caracterizadas por grupos de pessoas que compartilham um conceito ou uma paixão por algo que fazem e que interagem regularmente para aprender a fazê-lo melhor" podemos perceber que em toda comunidade de prática existe um assunto que move a comunidade, ou seja, um domínio bastante definido que é a razão da existência da comunidade.

Comunidades de prática virtuais produzem uma enorme gama de conhecimento que muitas das vezes só é utilizado instantaneamente para resolver um dado problema de um membro da comunidade, e futuramente, quando um outro membro possui o mesmo questionamento, nem sempre esse conhecimento é reaproveitado da melhor maneira. A maior parte das comunidades virtuais possuem o recurso de armazenamento das mensagens trocadas, entretanto, mecanismos simples de busca são utilizados para a recuperação das mensagens que podem ser úteis a um membro com um problema já discutido anteriormente na comunidade. Sendo assim, a atitude de um membro mais experiente é procurar nas mensagens anteriores alguma que esteja relacionada à sua dúvida. Como em uma busca tradicional na web, ele informa um conjunto de palavras-chave, e o sistema lhe retorna um conjunto de mensagens que possuam aquelas palavras-chave, ou um subconjunto delas, de acordo com a consulta especificada pelo usuário, e com as capacidades de busca de informação fornecidas pelo sistema.

Sistemas clássicos de busca e recuperação de informação tradicionalmente enfocam a relação entre uma consulta e a informação armazenada. A exploração dos inter-relacionamentos entre pedaços selecionados de informação (que podem ser

facilitadas pelo uso de ontologias) pode colocar informações antes isoladas e sem significado em um contexto.

Utilizar uma ontologia que vá ao encontro das necessidades de uma comunidade de prática, faz com que ferramentas de gestão do conhecimento possam organizar as informações em classes conceituais pré-definidas da ontologia, permitindo um acesso ao conhecimento maior e mais natural. Vale ressaltar que é vital a preservação das relações sobre a informação armazenada. Estas relações podem cobrir diversos aspectos, como importância relativa, contexto, seqüência, significância, causalidade e associação (DAVIES *et al.*, 2002).

Sendo uma ontologia um recurso, que por definição, deve poder ser perfeitamente compreendido tanto por humanos quanto por computadores, parece razoável aceitar que uma ontologia se torne campo de trabalho comum tanto a membros da comunidade quanto a agentes de software que possam atuar nesta comunidade. Agentes de software serão definidos na seção a seguir.

## 3.5 - Agentes de Software

Uma vez estabelecidas as estruturas de armazenamento de conhecimento, é necessário que existam entidades capazes de processar, manipular e transmitir esse conhecimento.

Estas entidades estão representadas na Web Semântica como agentes de software.

Existem vários trabalhos sobre agentes de software na literatura. Estes trabalhos abordam tanto definições na área de inteligência artificial quanto na área de gestão do conhecimento. A seguir analisaremos alguns aspectos pertinentes à teoria de agentes de software, de maneira que possamos compreender melhor como estas entidades podem nos ajudar na proposta desta dissertação.

### 3.5.1 - O que são agentes de software?

Vários autores discutem o que realmente deve ser caracterizado como um agente. De acordo com ao grau de flexibilidade da definição, utilizando um pouco de criatividade, qualquer dispositivo que responda a um estímulo pode ser facilmente caracterizado como um agente. Como um exemplo prático (SHOHAM, 1990) relata que tudo pode ser caracterizado como um agente, mas nem sempre isto é vantajoso:

"É perfeitamente coerente tratar um interruptor de luz como um agente (bastante cooperativo), que possui a capacidade de transmitir corrente, e que, invariavelmente transmite corrente quando ele acredita que nós queiramos que ele o faça e quando não. Apertar o interruptor é nossa maneira de comunicar nossos desejos. Entretanto, enquanto é uma visão coerente, não nos traz nada de novo, uma vez que nós essencialmente entendemos o mecanismo o suficiente para termos uma descrição simples do seu comportamento".

Uma definição mais específica de agentes de software que muitos pesquisadores de agentes podem achar aceitável é: "uma entidade de software que funciona continuamente e autonomamente em um ambiente particular, freqüentemente habitado por outros agentes e processos" (SHOHAM, 1997). O requisito para continuidade e autonomia deriva do nosso desejo de que um agente seja capaz de executar nossas atividades de maneira flexível e inteligente, e que responda a mudanças no ambiente sem demandar por constantes intervenções humanas. Idealmente, um agente que funciona continuamente em um ambiente durante um grande período de tempo deveria estar apto aprender a partir de suas experiências. Adicionalmente, esperamos que um agente que habita um ambiente juntamente com outros agentes e processos, possa se comunicar e cooperar com eles, e possivelmente se mover de um lugar para outro para possibilitar tal comunicação.

Em geral, um agente de software age a serviço de alguém, de maneira que execute uma tarefa que lhe foi delegada. Mas, uma vez que ter que soletrar cada detalhe é uma tarefa bastante tediosa, é desejável que nossos agentes sejam capazes de inferir o que nós desejamos de acordo com o que lhe dizemos. Os melhores agentes então, não devem somente demonstrar alguma forma de conhecimento, mas também levar em conta as peculiaridades do usuário e a situação na qual está envolvido.

### 3.5.2 - Categorias de Agentes de Software

Por existir uma infinidade de trabalhos que tratam de agentes de software, foram criadas também uma infinidade de classificadores e categorias de tipos de agentes de software, alguns exemplos dessas classificações são: reatividade, autonomia (pró-atividade ou agência), colaboração ou sociabilidade, inteligência (intencionalidade, capacidade de inferência ou adaptatividade), mobilidade, agentes de interface, agentes de informação (e Internet) e personalidade (ou *believable*

*agents*). Estas características foram retiradas dos trabalhos de: Franklin, Graesser, (1996); Etizioni, Weld (1995); Moulin, Chaib-Draa (1996); Gilbert *et al.* (1995); Nwana (1996) e Wooldridge, Jennings (1995).

No entanto, para o propósito deste trabalho nos ateremos ao estudo de dois tipos de agentes: os agentes colaborativos e os agentes de informação e internet.

Nos tópicos a seguir veremos uma breve explicação sobre esses dois tipos de agentes.

### 3.5.3 - Agentes Colaborativos

A colaboração entre agentes é uma característica cuja definição possui acordo comum entre os autores. Alguns utilizam uma terminologia diferente na abordagem do assunto, mas nada que comprometa a definição objetivo.

Wooldridge e Jeenings (1995) fornecem uma definição para "habilidade social" que é a capacidade dos agentes de interagir com outros agentes (possivelmente humanos) através de algum tipo de linguagem de comunicação entre agentes (GENESERETH, KETCHPEL, 1994). Para Moulin e Chaib-draa (1996), um agente social é aquele que possui modelos explícitos de outros agentes. Etzioni e Weld (1995) e Franklin e Graesser (1996) utilizam a expressão "comportamento colaborativo" e a definem como o conjunto de agentes que podem trabalhar em conjunto com outros agentes de maneira a atingir um objetivo comum. Além disso, eles consideram uma outra característica da comunicação entre agentes, que seria a capacidade de comunicação ao nível de conhecimento. Neste atributo, os agentes devem ser capazes de comunicar-se entre si em uma linguagem que seria próxima a linguagem humana, e não somente através de padrões estabelecidos por protocolos. Segundo Nwana (1996), além das características base de colaboração, ele enfatiza que um agente colaborativo deve ter como característica marcante a autonomia, ou seja, a autonomia é pré-requisito para a colaboração. Em sua definição, agentes colaborativos devem dar ênfase a autonomia e cooperação (com outros agentes) de maneira a realizar tarefas solicitadas por seus proprietários. Além disso, ele considera que um agente colaborativo possua habilidade social, pró-atividade e várias outras características.

De maneira geral, podemos resumir as características desejadas em um agente colaborativo na seguinte lista:

- Capacidade de interagir com outros agentes (humanos e não humanos)
- Possuir alguma linguagem específica de comunicação
- Possuir modelos explícitos de outros agentes
- Ser capaz de agir em conjunto com outros agentes de maneira a atingir um objetivo comum

### 3.5.4 - Agentes de Informação e Internet

Um agente de informação é um agente que tem acesso a pelo menos uma, e potencialmente muitas fontes de informação, e está apto a reunir e manipular as informações obtidas destas fontes com o objetivo de responder a consultas propostas por usuários e outros agentes de informação (a rede de fontes inter-operantes de informação é freqüentemente citada como sistemas cooperativos e inteligentes de informação). Esta definição está nos trabalhos de Wooldridge e Jeenings (1995), e está diretamente relacionada ao papel que os agentes podem assumir na web. As fontes de informação podem ser bastante variadas, desde páginas web comuns, documentos, bancos de dados e até mesmo outros agentes.

A definição de Nwana (1996) também é bem próxima a de Wooldridge e Jeenings (1995), ressaltando que este tipo de agente executa o papel de gerenciar, manipular ou agregar informações de várias fontes distribuídas. Além disso, Nwana salienta que esta classificação está relacionada estritamente ao que o agente faz, e não ao que ele é, como acontece no caso de agentes colaborativos.

Este tipo de agente vem ganhando muita importância, já que a quantidade de informação a que somos expostos diariamente cresce exponencialmente. Agentes deste tipo serão responsáveis por analisar, filtrar e preparar o conteúdo que chegará até nós, sem que sejamos inundados pela grande quantidade de informação presente na web.

Bob Jonhson, um analista da Dataquest Inc cita:

"No futuro, agentes serão a única maneira de procurar na Internet, porque não importa o quão melhor organizada a Internet possa estar, não será possível lidar com o crescimento da informação".

## 3.6 - Ontologias, Agentes de Software e Comunidades de Prática

Percorremos uma série de definições, grande parte delas baseadas nos princípios da Web Semântica, até chegarmos na combinação que será o coração deste trabalho. Esta combinação é a associação de Ontologias e Agentes de Software aplicados a Comunidades de Prática.

A ontologia representará a estruturação do domínio e vocabulário utilizado na comunidade, a partir dela, agentes de software que se comportarão como membros da comunidade de prática serão capazes de fazer inferências, utilizando o conhecimento trocado na comunidade, combinado às associações presentes na ontologia de domínio. Esta combinação terá como objetivo a obtenção de um melhor reaproveitamento do conhecimento trocado pela comunidade, cuja combinação e inter-relacionamento pode ser responsável por gerar novo conhecimento. As definições de agentes de software fornecem um amplo campo de trabalho, e vários tipos de agentes podem ser utilizados para a implementação deste trabalho, partindo desde os agentes que trabalharão o conhecimento interno da comunidade, até possivelmente agentes que possam trazer conhecimento externo à comunidade.

## 3.7 - Conclusão

A Web Semântica trouxe não só uma maneira de repensar a web atual, como um conjunto de conceitos bastante interessantes que podem ser aplicados em várias outras áreas de manipulação digital do conhecimento. O objetivo deste capítulo foi estudar os principais conceitos utilizados pela Web Semântica, de maneira a aplicá-los a um novo ambiente de troca de conhecimento, o ambiente das comunidades de prática. Analisamos rapidamente as definições básicas de representação do conhecimento, nos aprofundamos no estudo de ontologias, analisando as principais linguagens de construção, metodologias e utilização de ontologias. Concluímos com as definições das teorias de agentes, que serão bastante úteis na construção da arquitetura do trabalho aqui proposto.

No próximo capítulo falaremos sobre a última parte teórica a ser utilizada no propósito deste trabalho. Analisaremos as principais técnicas para a construção de sistemas de busca e recuperação de informação, bem como as principais técnicas de avaliação desses sistemas. Em seguida utilizaremos os conceitos apresentados nos capítulos 2, 3 e 4, para construir uma arquitetura multiagente, que utilizando uma

ontologia de domínio, seja capaz de interpretar semanticamente o conhecimento trocado em uma comunidade de prática.

# Capítulo 4 – Avaliação de Sistemas de Busca de Recuperação de Informação

Descrevemos até aqui três dos principais componentes do nosso trabalho: comunidades de prática, que produzirá e absorverá todo o conhecimento aqui tratado, um pouco de web semântica, quando falamos de agentes de software que serão responsáveis respectivamente por capturar, organizar e distribuir o conhecimento, e a ontologia, que será utilizada para classificar e contextualizar o conhecimento.

Entretanto, ainda não definimos como esses agentes trabalharão este conhecimento armazenado. Neste trabalho não tratamos de agentes inteligentes, sendo assim, necessitamos estudar as técnicas que serão utilizadas para que estes agentes possam realizar o trabalho de indexação e localização de conteúdo.

Por ser um assunto bastante disseminado e de técnicas consagradas, optamos então por utilizar os conceitos clássicos de busca e recuperação de informação.

Os modelos clássicos de busca e recuperação de informação se mostraram bastante eficientes, uma vez que aliam a capacidade de armazenamento de grandes quantidades de informação e rápida recuperação. O tema é foco de muitos estudos, gerou e continua gerando muitos trabalhos, uma vez que a quantidade de informação é crescente, e se faz necessária a otimização de métodos para armazenamento e busca de informações de qualidade.

Neste capítulo abordaremos os principais modelos de busca de recuperação de informações, bem como medidas de avaliação que nos fornecerão mecanismos de mensurar os resultados obtidos em nosso trabalho.

## 4.1 - Sistemas de busca e recuperação de informação

Modelos de busca e recuperação de informação possuem em comum o fato de utilizarem índices para armazenar e recuperar documentos. De um modo geral, um índice é cada palavra que aparece em um documento da coleção. Índices são freqüentemente denominados "termos".

O trabalho da busca e recuperação de informação consiste em mapear um conjunto de termos pergunta a um conjunto de termos resposta que mais se

aproximam do conjunto de termos da pergunta. Esta medida de proximidade é que será discutida na análise de cada um dos modelos.

Além disso Baeza-Yates e Ribeiro-Neto (1999) ressaltam que o problema central da busca e recuperação de informação é decidir quais documentos são relevantes e quais não são. Este tipo de decisão dependerá de um algoritmo de ranqueamento que tentará estabelecer uma ordem dos documentos recuperados. Quanto menor for a ordem de um documento, mais relevante ele será para a consulta submetida.

## 4.2 - Modelos Clássicos

Nas próximas seções analisaremos alguns dos principais modelos clássicos de busca e recuperação de informação de maneira a compreender e analisar as principais vantagens e desvantagens de cada um desses modelos.

### 4.2.1 - Modelo Booleano

É o modelo mais simples de todos. Nele são utilizadas expressões de álgebra booleana. Foi utilizado pelos primeiros sistemas bibliográficos. No modelo booleano uma consulta pode ser expressa através de conectores como: and, or e not.

O modelo booleano é de fácil implementação, entretanto uma consulta complexa por meio de conectivos booleanos pode não ser tão trivial para um usuário menos experiente. Até hoje os mecanismos mais consagrados de busca utilizam consultas expressas por meios de conectores, a prova do quanto o modelo booleano foi importante para a busca e recuperação de informações.

Entretanto, embora de fácil implementação e com um mecanismo simples de formulação de consultas o modelo booleano não apresenta ordenação de resultados, o que pode ser bastante incômodo para a visualização do usuário. De acordo com a consulta do usuário são recuperados documentos que possuem ou não possuem os termos listados, fazendo-se assim uma atribuição binária de pesos.

### 4.2.2 - Modelo Vetorial

O modelo vetorial é baseia-se na premissa de que a utilização de pesos binários é limitante, e possui uma proposta onde é possível classificar parcialmente determinado documento. Isto é feito através da atribuição de pesos não binários aos

termos da consulta e do documento. Estes pesos são utilizados então para calcular o grau de similaridade entre a consulta do usuário e cada documento presente na coleção. Tendo um valor não binário como unidade de comparação, o modelo vetorial considera documentos que atendam parcialmente à consulta submetida. Além disso, é possível também ordenar esse conjunto de resultados, sendo assim, o conjunto de respostas se torna muito mais relevante do que o presente no modelo booleano (considerando que a representação dos resultados será mais proveitosa ao usuário).

O modelo vetorial considera que a consulta e o documento podem ser representados por vetores, e o co-seno do ângulo $\theta$ entre esses vetores determina a similaridade entre o documento e a consulta.

Seja $w_{id}$ o peso de um termo i em um documento d, e $w_{iq}$ o peso de um termo i em uma consulta q, a similaridade entre um documento e uma consulta pode ser calculada como (SALTON, BUCCKLEY, 1988):

$$sim(d,q) = \frac{\sum_{i=1}^{t} w_{id} \times w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{id}^2} \times \sqrt{\sum_{i=1}^{t} w_{iq}^2}}$$

Os pesos representam a importância de cada termo para cada consulta e documento. Para calcular os pesos $w_{id}$ e $w_{iq}$, deve-se considerar o conceito de freqüência de um termo em um documento.

O peso de um termo em um documento (ou consulta) é dado pela fórmula:

$$W_{id} = freq(t_i,d) \text{ x } idf_i$$

Onde :

$freq(t_i,d)$ = freqüência do termo i no documento

$idf_i$ = inverso da freqüência do termo na coleção de documentos

O $idf_i$ (*inverse document frequency*) é calculado da seguinte maneira:

$$idf_i = \log \frac{N}{n_i}$$

Sendo $N$ o número de documentos na coleção e $n_i$ a quantidade de documentos que possuem o termo i.

A partir da análise das fórmulas podemos concluir facilmente que: quanto maior for o número de vezes que o termo aparece em um documento, e o quanto menor for o número de documentos que apresentam o termo, maior o peso deste termo no documento.

O modelo vetorial, embora com resultados bastante eficientes é um modelo simples de ser implementado, computa facilmente a similaridade entre consultas e documentos além de ter um comportamento bem estável com coleções genéricas.

Não à toa o modelo vetorial é um dos modelos mais bem aceitos e populares entre a comunidade de busca e recuperação da informação. Dedicamos uma maior atenção a este modelo, uma vez que este foi o modelo escolhido para o desenvolvimento deste trabalho.

### 4.2.3 - Modelo Probabilístico

O modelo probabilístico tenta dar ao problema de busca e recuperação de informação uma visão com os conceitos de probabilidade. Utiliza-se de pesos binários que representam presença ou ausência de termos. O resultado gerado pelo modelo baseia-se na probabilidade de que um documento seja relevante para uma consulta.

Neste modelo é utilizado o teorema de Bayes (VAN, 1979) como fundamentação teórica. Além disso, se baseia no princípio probabilístico que diz que: dada uma consulta q e um documento $d_j$ na coleção, o modelo probabilístico tenta estimar a probabilidade de um usuário o documento $d_j$ interessante (i.e., relevante). O modelo assume que esta probabilidade de relevância depende somente da consulta e da representação dos documentos. O modelo assume também que existe um subconjunto de todos os documentos que o usuário prefere como o conjunto de respostas para a consulta $q$. Este conjunto é chamado de conjunto $R$ e deveria maximizar a probabilidade geral de relevância para o usuário. Documentos que pertencem ao conjunto $R$ são considerados relevantes à consulta. Documentos não presentes neste conjunto serão considerados não-relevantes.

A similaridade $sim(d,q)$ entre um documento e uma consulta corresponde a um fator $W_{d|q}$ que considera as probabilidades $P(+R_q|d)$ e $P(-R_q|d)$ respectivamente, as probabilidades de que um documento seja relevante a uma consulta e a probabilidade

de que um documento d não seja relevante a consulta. Sendo assim, um documento é considerado relevante para a consulta se $P(+R_q|d) > P(-R_q|d)$.

$$sim(d,q) = W_{d|q}$$

$$W_{d|q} = \frac{P(+R_q \mid d)}{P(-R_q \mid d)}$$

A partir de transformações matemáticas e da utilização do teorema de Bayes, (BAEZA-YATES, RIBEIRO-NETO, 1999) demonstra como chegamos na fórmula:

$$sim(d,q) = W_{d|q} = \sum_{i=1}^{t} x_i \times W_{qi}$$

Onde:

$x_i \in \{0,1\}$

$W_{qi} = \log r_{qi}(1\text{-}s_{qi})/s_{qi}(1\text{-}r_{qi})$;

$r_{qi}$ = probabilidade de que um termo de indexação i ocorra no documento, dado que o documento é relevante para uma consulta q.

$s_{qi}$ = probabilidade de que um termo de indexação i ocorra no documento, dado que o documento não é relevante para a consulta q.

A principal vantagem do modelo probabilístico, em teoria, é que os documentos são ordenados em ordem decrescente da probabilidade de serem relevantes. As desvantagens estão no fato da necessidade de se ter uma separação inicial entre documentos relevantes e não relevantes e principalmente por não considerar a freqüência com que um termo ocorre em um documento.

### 4.2.4 - Modelo de Indexação Semântica Latente

O modelo de indexação semântica latente difere dos outros modelos em relação ao fato de assumir que, um texto está mais relacionado aos conceitos intrínsecos do que aos termos índices utilizados na sua descrição (BAEZA-YATES, RIBEIRO-NETO, 1999). Sendo assim, o processo de combinação entre consulta e documentos é feito através do batimento dos conceitos presentes no documento ao invés do batimento termo de índice / documento. Isto permite a recuperação de

documentos, mesmo que seus termos não estejam indexados, bastando somente que compartilhe conceitos relevantes com outro documento que é relevante para a consulta em questão.

A principal idéia por traz do modelo de indexação semântica latente é mapear cada documento e vetor consulta em um espaço dimensional menor, que está associado a conceitos. Isto é obtido através do mapeamento dos termos de índice do vetor em seu espaço dimensional menor. A proposta é que a recuperação de informação neste espaço reduzido possa ser superior à recuperação no espaço de termos de índice.

## 4.3 - Avaliação de SRI

Quando analisamos avaliação de sistemas de busca e recuperação de informação primeiramente devemos analisar o mecanismo de recuperação que será avaliado. Um mecanismo de recuperação pode consistir simplesmente de uma consulta processada em *batch* (i.e. o usuário submete uma consulta e recebe uma resposta) ou um conjunto de sessões interativas (i.e. o usuário especifica suas necessidades de informação através de uma série de passos interativos com o sistema. Além disso, o mecanismo de recuperação pode também considerar uma combinação dessas duas estratégias. Entretanto, o processamento *batch,* e uma sessão interativa são estratégias bem distintas, e por isso devem ser avaliadas de maneiras diferentes, considerando variáveis diferentes. Por exemplo na sessão interativa são muito importantes as características de interface, o comportamento do sistema em relação à interação do usuário e a duração da sessão. Já no processamento *batch* nenhuma dessas características são relevantes, sendo levada em consideração somente a qualidade do conjunto de respostas gerado.

### 4.3.1 - Medidas de Desempenho

A maioria das técnicas utilizadas na avaliação de sistemas de busca e recuperação de informação considera dois índices que medem o quanto um SRI pode ser eficiente. Esses índices são: precisão (*precision*) e revocação (*recall*).

Para a avaliação de um sistema de busca e recuperação de informação em geral são necessários os seguintes requisitos:

- Uma coleção de documentos

- Um conjunto de consultas
- Um conjunto de respostas idéias a cada uma das consultas (que podem ser determinadas por especialistas dos assuntos em questão)

Dados estes requisitos é possível avaliar automaticamente se o conjunto de documentos retornados pelo sistema corresponde ou não ao que se espera, no caso o conjunto ideal de respostas.

Tendo essas informações podemos definir precisão e revocação.

Precisão é a fração de documentos retornados que é relevante.

Revocação é a fração dos documentos relevantes que foram recuperados.

Considerando |N| o conjunto de respostas ideal e R o vetor resultado recuperado pelo SRI, temos que:

$$Precisão = \frac{|N \cap R|}{|R|}$$

$$Revocação = \frac{|N \cap R|}{|N|}$$

Dados os dois índices, podem ser gerados gráficos semelhantes ao da figura 4.

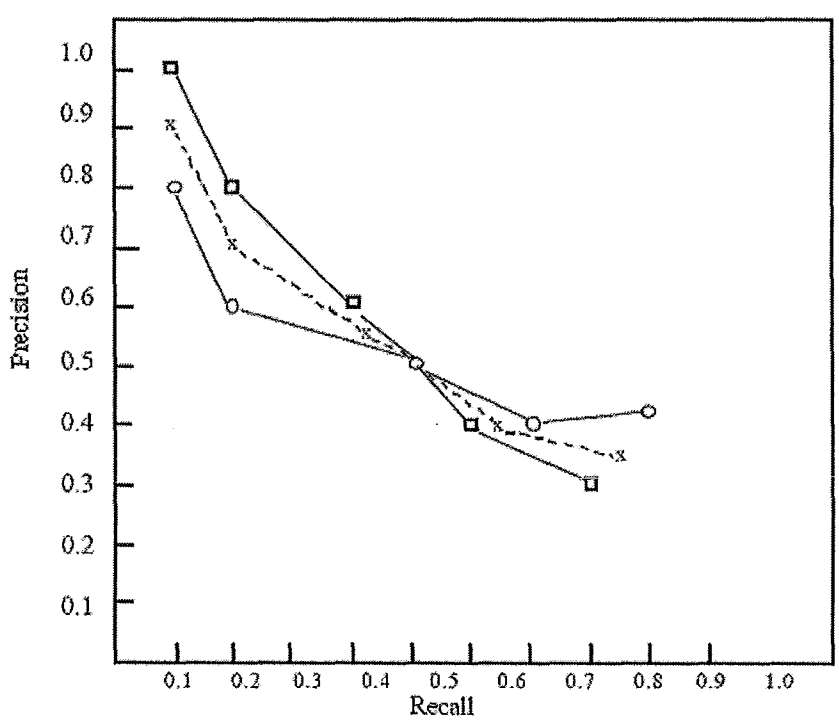



**Figura 4 – Exemplo de gráfico de precisão x revocação para três consultas (RIJSBERGEN, 1979)**

Em geral, para a comparação desses índices são utilizados níveis padrão de revocação 0%, 10%, 20%....100%. A fórmula de precisão acima corresponde à avaliação de uma única consulta. Entretanto, algoritmos de busca e recuperação de informação são avaliados utilizando-se uma série de consultas distintas. Neste caso, uma curva distinta de precisão x revocação é gerada, como pode ser observado na figura 4.

Para avaliar a performance de um algoritmo de busca e recuperação de informação sobre todas as consultas, pode ser calculada uma precisão média como mostrado a seguir:

$$\overline{P}(r) = \sum_{i=1}^{Nq} \frac{Pi(r)}{Nq}$$

Onde:

$\overline{P}(r)$ é a precisão média no nível de revocação r

$N_q$ é o número de consultas utilizadas

$Pi(r)$ = é a precisão no nível de revocação r para a i-ésima consulta

A precisão média juntamente com a revocação figuram um padrão de estratégia de avaliação de SRI e têm sido usados intensamente na literatura de busca e recuperação de informação.

### 4.3.2 - Padrões Internacionais de Avaliação - TREC

Como pudemos ver até aqui, os aspectos da efetividade dos resultados de uma busca dependem de quais resultados são relevantes ao usuário (precisão) se estes são apresentados em ordem de relevância, e a quantidade máxima de resultados relevantes que são trazidos em relação a um conjunto maior de documentos que o usuário consideraria relevante (revocação).

A comunidade de busca e recuperação dispõe então de uma metodologia de avaliação que é tanto cientificamente rigorosa quanto satisfatória do ponto de vista do usuário final. Esta metodologia está representada pela TREC (*Text Retrieval Conference*) um programa de avaliação suportado pelo Instituto Nacional Norte Americano de Padrões e Tecnologia (NIST, 2006). A metodologia da TREC é o

resultado de décadas de debate sobre avaliação na comunidade de busca e recuperação de informação.

A TREC possui uma grande coleção de dados de jornais e documentos do governo. Anualmente, a conferência é realizada, e são inscritos para análise sistemas de busca e recuperação de informação. A estes sistemas são submetidas consultas cujos resultados são enviados ao NIST (*US National Institute of Standards and Technology*) para avaliação de juízes humanos.

Nos últimos anos a TREC tem dado especial atenção à busca na Web, uma vez que este tipo de busca difere bastante da busca realizada em uma coleção controlada com é o caso da coleção típica da TREC. É muito difícil realizar avaliações efetivas de mecanismos de busca na grande rede, uma vez que os dados mudam diariamente, sendo assim, novos métodos de avaliação de busca na Internet têm sido propostos (HAWKING *et al*,1999).

### 4.3.3 - Outras técnicas de sistemas de busca e recuperação de informação

Até aqui demonstramos algumas das principais técnicas de implementação de SRI bem como seus mecanismos de avaliação. Entretanto, existe um outro conjunto de técnicas que tem como objetivo aumentar ainda mais o potencial dos SRI. Estas técnicas misturam reformulação de consultas e análise comportamental de usuários de SRI. A seguir uma breve descrição de cada uma dessas técnicas.

#### 4.3.3.1 - Filtragem de Informação

A filtragem de informação tem como premissa oferecer ao usuário aquilo que é de seu interesse. Antes de mais nada é necessário que o SRI possua informações sobre o perfil do usuário que realiza a consulta, seus assuntos de interesse, suas buscas realizadas anteriormente, documentos que costuma ler e até mesmo as páginas visitadas por este usuário.

A Google (GOOGLE, 2007) possui um arsenal de ferramentas para filtragem de informação. Com o google desktop é possível saber o conteúdo dos documentos do usuário, comunidades do Orkut também podem ajudar na identificação de interesses, troca de emails e chat são uma grande fonte de informação sobre os costumes do

usuário, além do mais recente google busca personalizada, onde todas as buscas são armazenadas em um histórico, assim como os links em que o usuário clicou após realizar a busca, assim o sistema possui retorno do que agradou ao usuário em determinada busca.

Segundo Belkin e Croft (1992), filtragem de informação e recuperação de informação são dois lados da mesma moeda, trabalham para ajudar pessoas a obter informações necessárias para executar suas tarefas.

#### 4.3.3.2 - Categorização de Informação

Categorização é o processo de classificar documentos em categorias pré-definidas. Sua maior aplicação tem sido para atribuir categorias a documentos e depois utilizar essas categorias para suportar recuperação e filtragem de informação.

As categorias podem ser definidas através de um pequeno conjunto de características e tendem a ser mais estáticas que os perfis de filtragem de informação. Sistemas de recuperação de informação apresentam baixo desempenho no contexto de categorização, principalmente devido ao vocabulário restrito que descreve as categorias e o vocabulário irrestrito dos documentos (YANG, 1994).

Entretanto, a categorização é o precursor de um movimento bastante interessante na história recente da Internet, a folksonomia.

#### 4.3.3.3 - Folksonomia

A folksonomia (*folksonomy*, classificação do povo) pode ser definida como o processo de categorização colaborativa de conteúdo de Internet. Nos últimos dois anos, a necessidade de novos mecanismos de organização pessoal e as interfaces web cada vez mais interativas proporcionadas pela Web 2.0 (WEB20, 2007) fizeram com que uma série de ferramentas que utilizam o conceito de folksonomia se tornassem bastante populares na Internet.

Marcadores sociais consistem basicamente em associar etiquetas em forma de palavras para classificar determinado conteúdo. Delicious (DELICIOUS, 2007) e Flickr (FLICKR, 2007) são duas ferramentas de bastante projeção na Internet e que utilizam marcadores sociais, com propósitos diferentes.

A descrição do Delicious é a seguinte: "Um gerenciador social de endereços de Internet. Ele lhe permite adicionar facilmente os sites que você gosta à sua coleção

pessoal de links, para categorizar estes sites com palavras-chave e compartilhar sua coleção não somente com seus navegadores e máquinas, mas também com outras pessoas" (SCHACHTER, 2004). Uma idéia simples: adicione seus sítios preferidos, classifique-os, e compartilhe-os, mas esta idéia tem mudado a forma de pensar em busca de informações na Internet.

Uma mesma página de Internet é classificada por centenas, milhares de usuários, cada um com sua classificação pessoal, e suas impressões do que aquele conteúdo significa para ele. A classificação daquele conteúdo pode ser exatamente o que outro usuário necessita.

O Flickr utiliza o mesmo mecanismo para alcançar outro objetivo: classificar fotos de álbuns pessoais. Com o aumento da capacidade de armazenamento das máquinas digitais, cada vez mais e mais conteúdo fotográfico é produzido, em um ano, uma única pessoa pode produzir gigabytes de fotos, contendo suas viagens, família e vida pessoal. Gerenciar este conteúdo e localizar determinada foto é bastante trabalhoso, caso não se disponha de mecanismos práticos de classificação. Enquanto os mecanismos de busca por conteúdo de imagens (busca por cores, formatos, contornos) ainda engatinham, mecanismos de classificação de imagens são uma boa saída para a localização de imagens baseadas em descrição.

A folksonomia carece de inúmeras características já extensamente estudadas em trabalhos sobre taxonomia: não possui hierarquia, também não apresenta suporte a sinônimos, é ambígua e também não possui um vocabulário controlado (MATHES, 2004). Entretanto, representa um enorme avanço no que diz respeito a atribuição de meta-dados e classificação de conteúdo web. Além disso, ferramentas encontraram uma maneira bastante efetiva para motivar os usuários a classificarem: classifique seu próprio conteúdo e compartilhe, se cada um fizer isso, estará ajudando a classificar um pedaço do conteúdo da Internet.

## 4.4 - Expansão de Consultas

Sem o conhecimento detalhado das informações contidas em uma coleção de documentos disponíveis para busca, a maioria dos usuários considera difícil formular consultas de boa qualidade para os propósitos de busca e recuperação. Mesmo os usuários mais experientes em geral utilizam entre dois e três ciclos de reformulação da consulta original para obter os resultados desejados.

Ainda assim, um problema que persiste é o fato de que: mesmo reformulando a consulta, a terminologia utilizada na coleção de documentos pode ser diferente da terminologia utilizada na formulação da consulta (KLINK, 2001).

Em Ruthven (2003) existe uma interessante comparação entre expansão de consulta interativa (realizada por humanos) e expansão de consulta automática. Os resultados deste trabalho mostram que na média, os usuários de sistemas de busca e recuperação de informação optam por piores alternativas de expansão de consultas do que os sistemas de informação. Um argumento em favor da expansão automática de consultas é que o sistema tem acesso a mais informações estatísticas à qualidade relativa da informação para a expansão de termos e pode fazer uma seleção melhor de quais termos adicionar à consulta. O principal argumento da expansão de consultas interativa é que ela dá maior controle ao usuário.

### 4.4.1 - Histórico

Os estudos sobre expansão de consultas não são recentes. Os trabalhos mais antigos datam das décadas de 60 e 70, como os de Maron, Kuhns (1960) e Rocchio (1971). Entretanto, com a explosão de informações do nosso século e a necessidade de lidar com elas, o assunto continua atual, e é consenso entre os pesquisadores que, embora com resultados comprovados de eficiência, o tópico ainda é uma técnica que é pouco explorada comercialmente.

### 4.4.2 - Técnicas de Expansão de Consultas

O problema de expansão de consultas resume-se em duas partes principais:

1. Como os termos da expansão serão selecionados
2. Quais serão os parâmetros estimados para estes termos

Para a análise desses problemas são sugeridas as técnicas de análise local e de análise global.

As técnicas de análise local trabalham com o conjunto de informações contidas nos documentos retornados, e a avaliação do usuário sobre os documentos retornados (*relevance feedback*). Após submeter uma consulta, o usuário recebe o primeiro conjunto resposta de documentos. Após avaliar estes documentos entre relevantes e não relevantes, o sistema submete-se a uma nova consulta, considerando a classificação dos documentos, retornando outros documentos que são similares aos

documentos classificados como relevantes pelo usuário e que não contenham as informações dos documentos classificados como irrelevantes pelo usuário.

Enquanto as técnicas de análise local extraem informações do conjunto local de documentos recuperados para expandir a consulta, as técnicas de análise global utilizam todo o conjunto de documentos para a expansão da consulta, exemplos de aplicação desta técnica é a construção de tesauros automáticos baseada no conjunto de todos os documentos presentes na coleção.

Além dessas técnicas, com a proposta da Web Semântica, alguns trabalhos recentes têm obtido bons resultados com a utilização de ontologias para a expansão de consultas. Trabalhos como Pinheiro (2004) e Navigli, Velardi (2003) mostram a aplicação de um processo de expansão de consultas utilizando ontologias, submetendo à consulta expandida à API de busca do Google (GOOGLEAPI, 2006).

### 4.4.3 - Atribuição de pesos aos termos expandidos

Como visto na seção anterior, o primeiro desafio na expansão de consultas é escolher os mecanismos pelos quais os termos serão selecionados. Estes mecanismos podem ser: utilização de tesauros, ontologias, termos de documentos recuperados pela consulta original entre outros. Porém, embora com resultados comprovados de aumento da eficiência em mecanismos de buscas, se a expansão de consultas não for realizada de forma controlada e criteriosa, ao invés de trazer melhores resultados, o retorno da busca pode culminar em uma coleção de resultados até mesmo pior do que a que seria trazida através da consulta original.

No capítulo seguinte, analisaremos um trabalho bem sucedido de atribuição de pesos, o qual, baseando-nos nas conclusões do autor, fizemos algumas adaptações e obtivemos resultados bastante interessantes com o protótipo desta dissertação.

### 4.4.4 - Seleção de termos e o Léxico Gerativo de Pustejovsky (LGP)

Um dos grandes desafios deste trabalho foi identificar quais relacionamentos na ontologia deveriam ser considerados para efeito de expansão de consultas. O objetivo era selecionar determinadas categorias de relacionamentos que fizessem com que os termos presentes na consulta expandida fossem realmente relevantes, e trouxessem o mínimo possível de termos que poderiam ser prejudiciais na obtenção

dos resultados. Buscamos nas técnicas de processamento de linguagem natural algumas respostas às nossas perguntas, no que diz respeito a mecanismos de captação de similaridade semântica entre conceitos.

Em Gonzales (2000) é feita uma análise sobre o conjunto de técnicas provenientes da área de processamento de linguagem natural a serem utilizadas na busca e recuperação de informações. Em um sistema de processamento de linguagem natural (PLN) um léxico pode ser definido como "uma relação de palavras com suas categorias gramaticais e seus significados" (ALLEN, 1995). Ao contrário dos dicionários, que devem ser lidos por pessoas, um léxico computacional deve ser formalizado e legível por computador (WILKS *et al*,1996).

Aqui não estaremos interessados nas categorias gramaticais de um item lexical, mas nos seus significados e nos relacionamentos que apresentam com outros itens, através de "relações semânticas lexicais" e operações gerativas de um campo semântico.

Algumas das principais relações semânticas lexicais são: sinonímia (termos de mesmo significado), homonímia (termos de mesma grafia, porém com significados diferentes), polissemia (mesmo termo com mais de um significado), antonímia (termos de significados opostos ou complementares) e hiponímia (relação de um termo com significado genérico e um termo com significado específico), meronímia (relação todo-parte, incluindo predicados como parte-de, feito-de).

Gonzales (2000) propõe ainda as seguintes operações gerativas e suas definições:

- Especialização: Dados os itens lexicais a e b, um conjunto de itens lexicais B e uma relação de hiponímia Hn, diz-se que oEsp é uma operação de especialização, tal que:

$$\forall a\ (oEsp(a)=B) \leftrightarrow \forall b \in B\ (Hn(b,a))$$

  Ex: Os itens lexicais "gato" e "cachorro" são obtidos através da operação de especialização sobre o item lexical "animal".

- Co - Herança: Dados os itens lexicais a, b e c, um conjunto de itens lexicais B, e uma relação de hiponímia Hn, diz-se que oCoh é uma operação de co-herança, tal que:

$$\forall a\ (\textit{oCoh}(a)=B) \leftrightarrow \exists c\ (\forall b \in B\ (\textit{Hn}(a,c) \wedge \textit{Hn}(b,c)))$$

Ex: O item lexical "animal" é obtido através da operação de co-herança sobre os itens lexicais "gato" e "cachorro".

- Associação: Dados os itens lexicais a e b, um conjunto de itens lexicais B e uma relação de implicatura/pressuposição IP, diz-se que oAss é uma operação de associação, tal que:

$$\forall a\ (\textit{oAss}(a)=B) \leftrightarrow \forall b \in B\ (\textit{IP}(a,b))$$

Ex: A partir do item lexical "assar" podem ser obtidos por associação os seguintes itens lexicais: "calor, assadura, assador".

- Equivalência: Dados os itens lexicais a, b, c e f, o conjunto de itens lexicais B, uma relação de hiponímia Hn e as operações de co-herança oCoh e de associação oAss, diz-se que oEqv é uma operação de equivalência, tal que:

$$\forall a\ (\textit{oEqv}(a)=B) \leftrightarrow (\forall b \in B\ (\ B \subseteq \textit{oCoh}(a)=b) \wedge \exists c\ ((c \notin \textit{oAss}(f)\ (\textit{Hn}(a,f)))$$
$$\wedge\ c \in \textit{oAss}(a) \wedge c \in \textit{oAss}(b)\ )))$$

Ex: A equivalência apresenta relações "um tipo de" sendo assim, podemos obter por equivalência itens lexicais que se encaixam neste preceito. "casa" é um "tipo de" moradia, em outras palavras, uma casa equivale à uma moradia.

- Decomposição: Dados os itens lexicais a e b, um conjunto de itens lexicais B e uma relação de meronímia Mn, diz-se que oDec é uma operação de decomposição, tal que:

$$\forall a\ (\textit{oDec}(a)=B) \leftrightarrow \forall b \in B\ (\textit{Mn}(a,b))$$

Ex: Através da operação de decomposição do item lexical "apartamento" obtemos os itens lexicais "porta" e "janela" pois, apartamento possui portas e janelas.

- Agregação:Dados os itens lexicais a e b e uma relação de meronímia Mn, diz-se que oAgr é uma operação de agregação, tal que:

$$\forall a\ \forall b\ (\textit{oAgr}(a)=b) \leftrightarrow (\textit{Mn}(b,a))$$

Ex: Através da operação de agregação sobre o item lexical porta, obtemos o item lexical "apartamento", pois uma porta está contida em um apartamento ou um "apartamento" contém uma "porta". É o oposto da operação de decomposição.

Expansão de consultas utilizando ontologias é um dos grandes tópicos deste trabalho, e será abordado em maiores detalhes no capítulo seguinte, que além da arquitetura do trabalho aqui realizado, também sugere mecanismos de seleção de termos na ontologia para a expansão e atribuição de pesos aos termos selecionados.

## 4.5 - Conclusão

Ao longo deste capítulo analisamos as principais técnicas de busca e recuperação de informação. Analisamos também os mecanismos de avaliação utilizados para avaliar estes sistemas. Fizemos um rápido panorama sobre novos paradigmas que podem auxiliar na busca e recuperação de informações, estes paradigmas estão diretamente relacionados na evolução da Internet e a necessidade de categorização de informação. Chegando em um dos tópicos mais importantes deste capítulo, falamos sobre o processo de expansão de consultas e atribuição de pesos, citando conceitos amplamente utilizados nos trabalhos sobre processamento de linguagem natural, que, embora não seja o tópico principal desta dissertação, foi fundamental no momento em que decidimos como utilizaríamos a ontologia na expansão da consulta. Este assunto, dada tamanha importância será mais uma vez abordado no próximo capítulo, que descreve a arquitetura proposta por esta dissertação.

# Capítulo 5 – COOPRACTICE – Arquitetura e ferramenta

Nos capítulos anteriores introduzimos os principais conceitos que envolvem agentes, ontologias, comunidades de prática e busca e recuperação de informações. Estes conceitos formam a base teórica da proposta apresentada por esta dissertação. A partir deste capítulo, começaremos a analisar a criação de uma arquitetura de busca e recuperação de informação, que, apoiada por uma ontologia de domínio terá como objetivo trazer resultados mais precisos e significativos do que os sistemas clássicos de busca e recuperação de informação. Nossa teoria propõe que: dado um sistema de busca e recuperação de informação de um domínio de informação pré-definido, se este sistema possuir conhecimento do domínio, o próprio sistema pode colaborar com as consultas submetidas a ele.

Para a base do nosso estudo analisamos as comunidades de prática virtuais por serem um ótimo exemplo de um domínio de informação definido. Também deve-se levar em consideração o fato de que consultas em forma de perguntas são submetidas diariamente, nos fornecendo um grande material de estudo na avaliação de sistemas de busca e recuperação de informação baseados em um domínio de informação.

Este capítulo tem como objetivo propor a arquitetura de um sistema de busca e recuperação de informação, que, atuando em uma comunidade de prática e apoiado por uma ontologia será responsável por minerar a base de conhecimento da comunidade de prática, atuando como um membro da comunidade e fornecendo indicações de documentos e assuntos anteriormente abordados, com a meta de conseguir responder a perguntas levantadas na comunidade.

## 5.1 - Arquitetura do sistema

O COOPRACTICE é uma arquitetura multi-agente de manipulação do conhecimento. A arquitetura é composta de um conjunto de agentes que são responsáveis por filtrar, analisar e indexar o conteúdo trocado na comunidade de prática virtual, bem como o conteúdo externo relacionado ao domínio do assunto.

A arquitetura é formada por dois módulos principais: o módulo de organização de conteúdo, e o módulo de solução de problemas. Estes dois módulos não são

completamente independentes, possuindo como estruturas em comum a base de conhecimento da comunidade e a ontologia de domínio.

O módulo de organização de conteúdo (MOC) é responsável pela captação e indexação de todo o conteúdo que fará parte da base de conhecimento da comunidade. Os agentes pertencentes a este módulo deverão captar, organizar o conhecimento de maneira semi-estruturada, descartar informações irrelevantes e também buscar informações externas, que possam ser úteis à biblioteca de conhecimento da comunidade de prática.

O módulo de solução de problemas (MSP) conterá os agentes que farão a interface com os membros da comunidade de prática. Esses agentes terão como funcionalidade receber os questionamentos enviados à comunidade, fazendo análises no conhecimento pré-estruturado já organizado pelo MOC, e apoiado pela ontologia de domínio, selecionando as melhores mensagens ou documentos relacionados à mensagem originalmente submetida.

A figura 5 mostra o modelo de interação entre o MOC e o MSP.

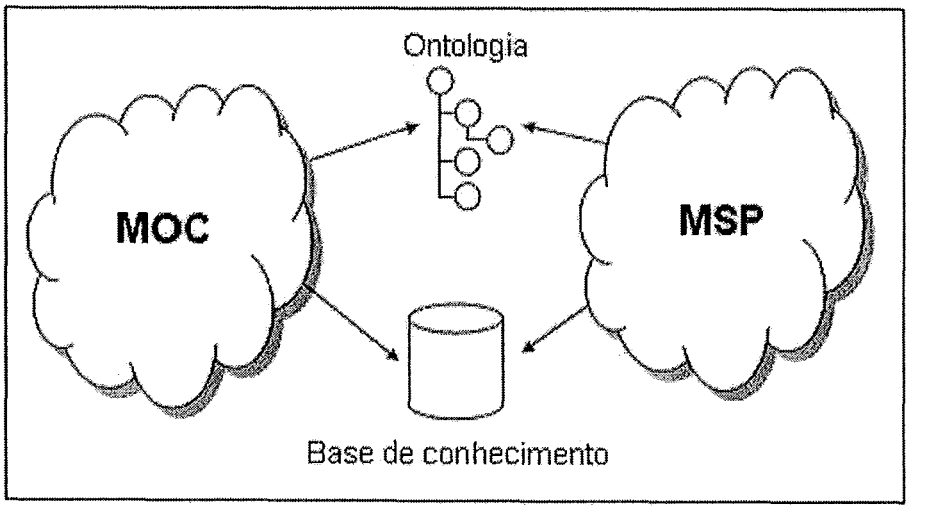



**Figura 5 – Modelo de interação do MOC e do MSP**

Além do MOC e do MSP existe um agente externo, que não se encontra em nenhum desses módulos. Ele será o responsável por receber o questionamento da comunidade e enviar a resposta obtida pelo MSP. Este agente é o "agente de interface com a comunidade" (AIC).

Nas próximas seções analisaremos as responsabilidades do agente de interface com a comunidade bem como os módulos de solução de problemas e de indexação de conteúdo.

### 5.1.1 - Agente de Interface com a Comunidade (AIC)

O AIC é a porta de entrada do sistema, e onde se inicia a atividade principal a ser realizada, que é a solução de problemas. Ele pode ser instanciado de duas maneiras principais: na forma de receptor e remetente de e-mails (no caso de comunidades mediadas por e-mail) ou através de uma interface web, para comunidades cuja interação é feita através de páginas da Internet (por exemplo, comunidades com fórum de mensagens).

O AIC é ativado basicamente por dois eventos: recebimento de uma mensagem da comunidade e recebimento da resposta do MSP a uma pergunta enviada para a comunidade. A atuação do AIC pode ser melhor representada na figura 6. Em linhas gerais, o AIC recebe uma mensagem enviada para comunidade. Ele duplica esta mensagem, e envia uma cópia para o MOC, e outra para o MSP. O MOC se encarregará das tarefas de indexação desta nova mensagem, para que esta conste na base de conhecimento da comunidade no COPPRACTICE. O MSP iniciará o processo de análise da pergunta e à seleção de mensagens relacionadas e documentos relevantes, que serão retornados ao AIC, para que ele possa enviar o conteúdo resultante à comunidade.

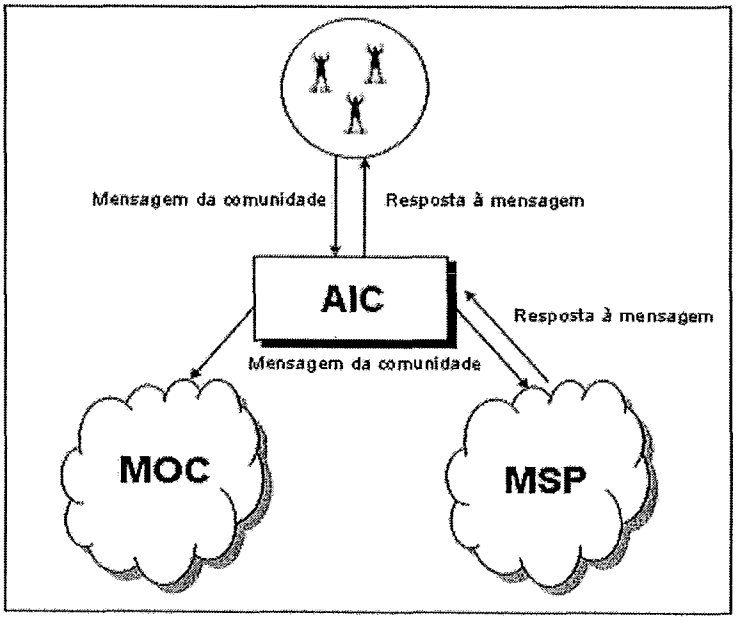


**Figura 6 – Modelo de interação do AIC com os outros módulos de agentes**

## 5.1.2 - Módulo de Organização de Conteúdo (MOC)

O Módulo de Organização de Conteúdo, o MOC será responsável pelo conjunto de atividades que farão com que a base de conhecimento da comunidade possa ser estruturada, com o objetivo de fornecer acesso efetivo e rápido às informações requisitadas. O MOC é composto por vários agentes que serão responsáveis por executar tarefas específicas de análise, indexação, seleção e organização de conteúdo. O final de cada etapa de manipulação do conhecimento pelo MOC resultará em uma visão do conhecimento, que estará representada através de várias dimensões, como pode ser visto na figura 7.

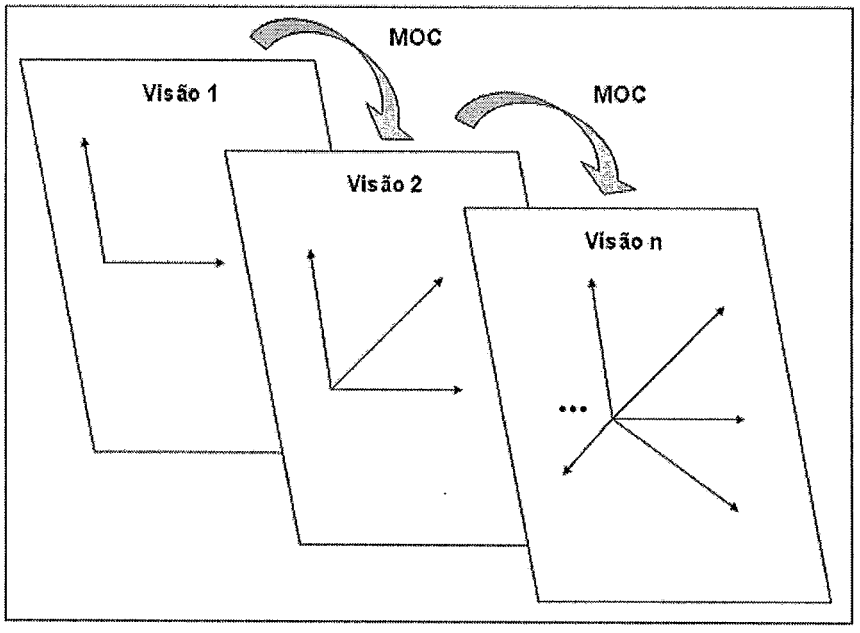



**Figura 7 – Resultado do trabalho do MOC: Visões e Dimensões**

Na figura 8 podemos observar em um diagrama de seqüência a interação dos agentes presentes no MOC. Nas seções seguintes temos uma breve descrição do comportamento de cada um desses agentes.

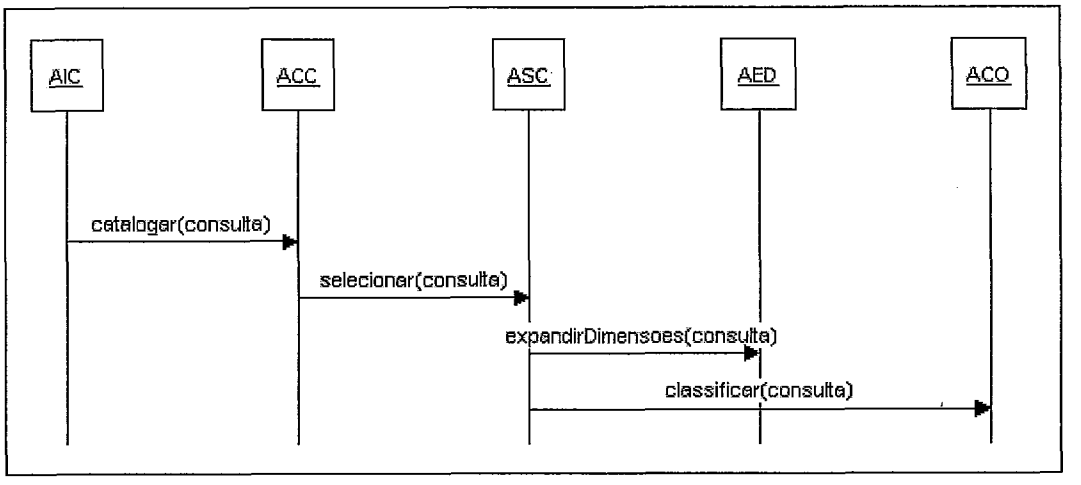



**Figura 8 – Diagrama de seqüência do módulo de organização de conteúdo**

### 5.1.2.1 - Agente de Captação de Conteúdo (ACC)

O agente de captação de conteúdo (ACC) é o primeiro agente a executar uma atividade dentro do MOC. É também a atividade mais simples. Ele é responsável simplesmente por receber as mensagens enviadas pelo AIC (descrito na seção acima) e catalogá-las. Nenhum processamento adicional é efetuado. É também do ACC a responsabilidade de colocar um "flag" indicando que aquela mensagem ainda não foi completamente processada pelo MOC. A visão gerada pelo ACC é a primeira visão de conhecimento da base de conhecimento da comunidade. Ela possui as dimensões mais básicas da mensagem, que são: autor da mensagem ou remetente, assunto, corpo da mensagem e a data de recebimento.

### 5.1.2.2 - Agente de Seleção de Conteúdo (ASC)

A partir da atuação do ASC começam realmente as atividades de processamento das mensagens pelo MOC. O ASC será responsável por identificar aquelas mensagens que são relevantes ao assunto da comunidade e as que não são. O ASC indicará também se aquela mensagem deve ser respondida ou não. Para que sejam identificadas mensagens cuja relevância permita que a mensagem seja indexada ou descartada, o ASC pode utilizar uma série de heurísticas. Um exemplo bastante simples de um desses critérios utilizados é a presença do texto "off-topic" no assunto da mensagem. Em geral mensagens com este assunto abordam assuntos não

pertencentes ao domínio principal da comunidade, membros da comunidade utilizam esta denominação justamente para indicar que o assunto deles não está relacionado ao escopo ali tratado, em uma comunidade de linguagem Java poderia ser, por exemplo, o anúncio sobre cursos de Java.

### 5.1.2.3 - Agente de Expansão de Dimensões (AED)

Uma vez selecionadas as mensagens que devem ser indexadas, será necessário fazer uma análise mais profunda de cada uma dessas mensagens. Até o momento possuímos as quatro dimensões iniciais: remetente, assunto, corpo da mensagem e data de recebimento. O objetivo do AED será o de expandir essas dimensões, mais especificamente, de expandir a dimensão "corpo da mensagem" para que esta possa ser desmembrada em trechos relevantes, e comuns a várias mensagens, permitindo-se assim que seja possível estabelecer critérios mais específicos de busca, além da atribuição de peso (relevância) da busca em cada uma das dimensões.

Da análise de mensagens trocadas em listas de discussão sobre linguagens de programação, em geral, podemos reconhecer alguns dos seguintes trechos:

- Trechos de texto corrido, onde o membro explica o seu problema, ou fornece uma resposta a uma pergunta de um outro membro.
- Trechos de código fonte, onde o usuário lista trechos do seu código problemático, ou um trecho de código que pode representar a solução a um problema de um outro membro.
- Referências a endereços de Internet que possam conter a resposta ao questionamento enviado.

Desta maneira, podemos referenciar estes três trechos como três novas dimensões de indexação.

A utilização destas novas dimensões se dará da seguinte maneira:

1. Dimensão de texto: esta possuirá um peso maior em relação às outras dimensões, é nela que se espera encontrar a maior parte das soluções aos questionamentos enviados pela comunidade. Referências à dimensão de código fonte também serão consideradas como fortes indícios da relevância do documento recuperado.
2. Dimensão de código fonte: o peso de busca nesta dimensão será a metade do peso correspondente à dimensão de texto. Isto porque

termos presentes em trechos de código-fonte nem sempre serão tão relevantes já que funções da linguagem são utilizadas repetidamente. No entanto, como dito na dimensão acima, é importante considerar a referência à um código-fonte, uma vez que muitas das vezes a resposta à pergunta de um membro de uma comunidade de linguagem de programação é um trecho de um código-fonte.

3. Dimensão de referência: A dimensão de referência freqüentemente será caracterizada por seu texto iniciar por "http" ou "www". Ela não será considerada como índice, no entanto, o conteúdo apontado por esta referência poderá ser bastante útil para a atualização do conteúdo da biblioteca de conhecimento da comunidade.

O resultado final do trabalho do AED será a construção de uma nova visão com as dimensões originais: remetente, data, assunto e as recém criadas: texto, código-fonte e referência.

### 5.1.2.4 - Agente de Classificação Ontológica (ACO)

Este agente será responsável por determinar onde cada mensagem deve ser mapeada de acordo com os conceitos presentes na ontologia.

A construção da ontologia utilizada nesta arquitetura será demonstrada nas próximas seções. A ontologia Java possuirá um conjunto de conceitos e relacionamentos de grande relevância e abordados insistentemente nas comunidades Java.

O trabalho do ACO será mapear cada mensagem em um conjunto de conceitos presentes na ontologia. A atividade de mapeamento de mensagens em conceitos da ontologia está presente em Pereira et. al (2006). Assim, mensagens classificadas em um mesmo conjunto de conceitos poderão ser agrupadas por similaridade de assuntos.

### 5.1.2.5 - Agente de Indexação de Referências (AIR)

Além das atividades de organização do conteúdo que trafega pela comunidade, também é desejável que esta disponha de uma biblioteca particular de informações, onde os agentes possam pesquisar com o objetivo de encontrar informações adicionais além daquelas que constam na comunidade.

A dimensão de referências será o ponto de partida do trabalho deste agente. Cada referência sugerida por um membro da comunidade em relação a determinado assunto é um grande indício de conteúdo de qualidade, que poderá ser aproveitado pelos outros membros da comunidade. Com uma referência recomendada evita-se a necessidade de um "*crawling*" sem heurística pela web, mas partindo de uma informação consistente e direcionada.

O AIR partirá de uma referência mencionada em determinada mensagem, captará o conteúdo desta referência (que pode ser uma página web, livros, ou artigos em variados formatos) e indexará este conteúdo em uma nova dimensão, que será a dimensão de conteúdo de referências. Esta dimensão será a "biblioteca particular" da comunidade. Este agente não interage com nenhum outro agente da arquitetura, trabalhando de forma independente, sendo assim, não foi demonstrado em nenhum dos diagramas aqui apresentados.

O ACO (agente de classificação ontológica) também mapeará o conteúdo destas referências de maneira a identificar conceitos presentes neste conteúdo.

### 5.1.3 - Módulo de Solução de Problemas (MSP)

Após descritas as funcionalidades necessárias à seleção e indexação de mensagens, prosseguiremos à descrição do módulo que é a atividade fim desta arquitetura.

Como explicado nas seções anteriores, o AIC (agente de interface com a comunidade) enviará uma cópia da mensagem recebida ao MSP, aguardando por um conjunto de respostas que serão enviadas à comunidade.

O MSP, apesar de conter somente três agentes, é o módulo mais complexo e mais interessante deste sistema pois é nele que serão realizadas as atividades diferenciadas para a obtenção de melhores resultados na busca de informações.

Se no MOC o produto final era um conjunto de dimensões para posterior busca, no MSP um outro produto final possui quase tanta importância quanto os resultados retornados: uma mensagem expandida a partir da mensagem original submetida pelo membro da comunidade. É esta mensagem expandida que fará com que os resultados sejam mais significativos e próximos dos desejos do membro da comunidade ao submeter a mensagem.

Na figura 9 podemos observar o diagrama de seqüência da interação entre os agentes do MSP.

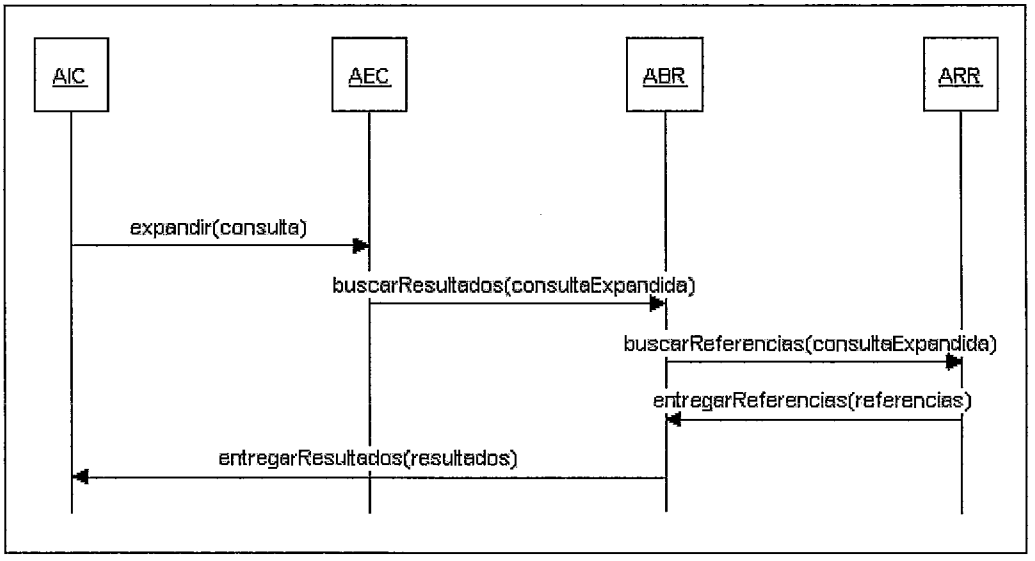



**Figura 9 – Diagrama de seqüência do MSP**

### 5.1.3.1 - Agente de Expansão de Consultas (AEC)

É no trabalho deste agente que uma mensagem se transformará em uma consulta e será a qualidade desta consulta que será determinante na relevância dos resultados retornados.

Por ter sido um assunto tão importante na realização deste trabalho e ter sido objeto de grande parte dos estudos aqui executados, o trabalho do AEC será tratado em uma sessão à parte, ao fim da apresentação dos demais agentes do MSP.

### 5.1.3.2 - Agente de Busca de Resultados (ABR)

O ABR é uma interface de entrada de consulta e saída de resultados. Dada uma consulta ele deve poder ser conectado a qualquer mecanismo de busca que possua uma interface para recebimento de consultas e fornecimento de respostas.

Na arquitetura aqui apresentada o ABR receberá a consulta expandida pelo AEC (agente de expansão de consultas) e submeterá a uma base de dados utilizando o modelo vetorial para busca e recuperação de informações.

Os documentos serão retornados em ordem de relevância e submetidos ao AIC (agente de interface com a comunidade) para que possam ser exibidos na interface apropriada.

#### 5.1.3.3 - Agente de Recuperação de Referências (ARR)

Além das respostas catalogadas na comunidade também retornaremos um conjunto de referências obtidas através do conteúdo de referências externas catalogadas pelo AIR (agente de indexação de referências). A consulta expandida será submetida da mesma maneira à base de referências indexadas. O conjunto de referências retornadas será também submetido ao AIC, para que encapsule na mesma mensagem o conjunto de respostas retornadas da comunidade, e as referências que podem ser úteis na abordagem do assunto em questão.

## 5.2 - Construção da Ontologia

A ontologia aqui utilizada foi construída de uma maneira semi-automática. Semi-automática porque inicialmente foi criada manualmente, e em uma segunda fase alimentada automaticamente por informações da web. São poucos os trabalhos que demonstram iterativamente a construção de uma ontologia. Optamos por detalhar aqui a criação da ontologia utilizada por este trabalho, de maneira que o processo, associados a modificações pertinentes ao problema em questão possa servir de base a outros trabalhos realizados.

### 5.2.1 - A ferramenta

Para a criação da ontologia utilizamos a ferramenta Protegé (PROTEGE, 2007). O Protegé é uma ferramenta que permite a construção de ontologias bastante completas, permitindo a definição visual de classes, propriedades, e instâncias, bem como restrições sobre propriedades entre outros aspectos comuns de uma ontologia. Além da definição de componentes visuais ele fornece também várias APIs (*application program interfaces*) para visualização da ontologia de outras maneiras, além de APIs de geração de código. Para a codificação da nossa ontologia, optamos pela utilização da OWL, definida e explicada no capítulo 3. A interface do Protege pode ser vista na figura 10.

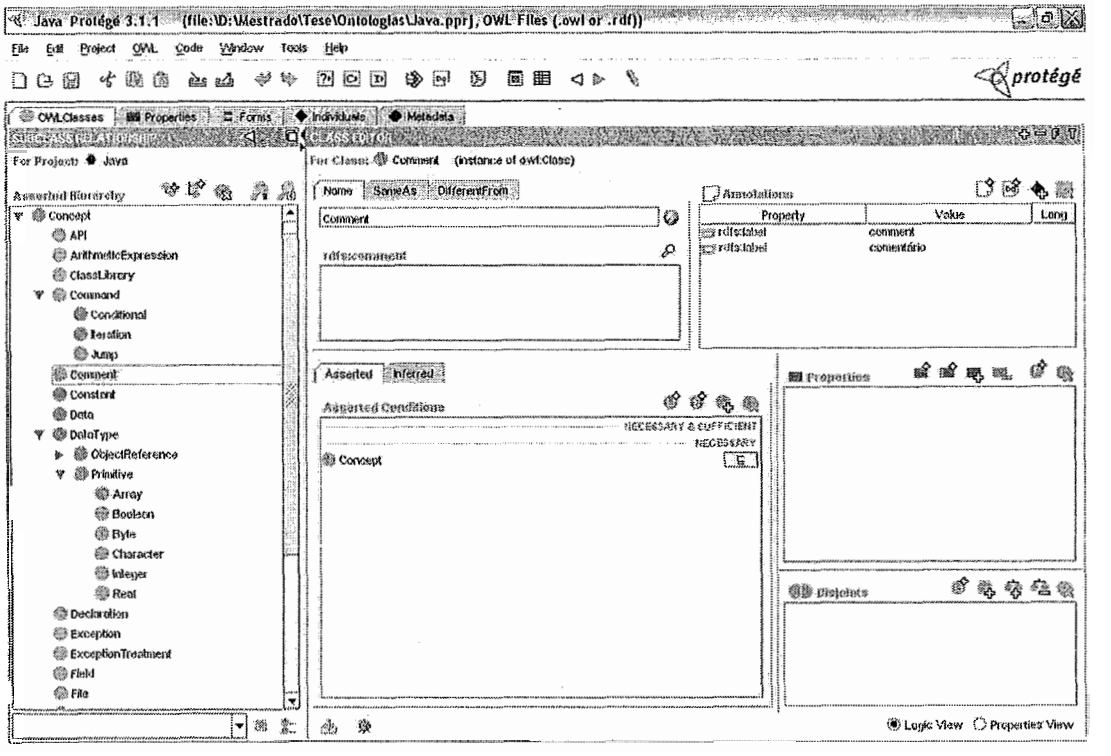



**Figura 10 – Tela do Protege**

A figura 11 mostra um trecho de ontologia gerada pelo Protegé.

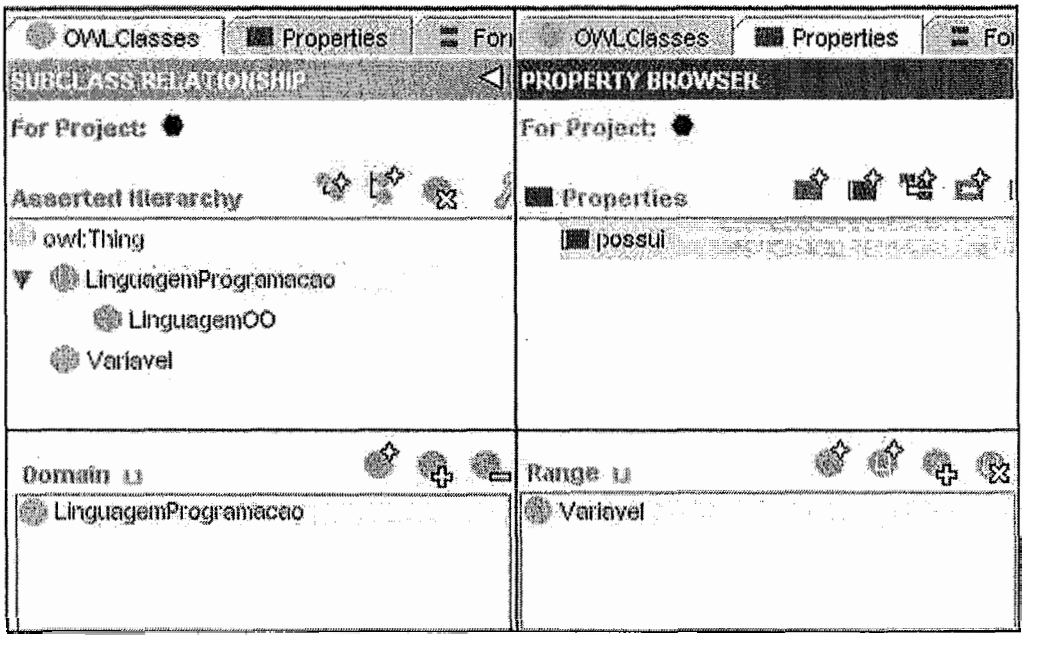



**Figura 11 – Exemplo de ontologia gerada com o Protege**

No primeiro quadrante temos a definição das classes (conceitos) da ontologia. Nesta ontologia possuímos três conceitos: LinguagemProgramacao,

LinguagemOO e Variavel, onde LinguagemOO é um subtipo de LinguagemProgramacao, ou seja, LinguagemOO herda atributos de LinguagemProgramacao.

No segundo quadrante, visualizamos o conjunto de propriedades desta ontologia. Neste caso só possuímos a propriedade "possui". Nos quadrantes inferiores podemos visualizar o domínio (*Domain*) e o alcance (*Range)* da propriedade "possui". Lendo-se que LinguagemProgramacao possui Variavel. Como visto no capítulo 3, é imediata a inferência que LinguagemOO possui Variavel, uma vez que LinguagemOO herda as propriedades de LinguagemProgramacao.

A representação OWL desta ontologia é exatamente o trecho demonstrado na figura 2 (capítulo 3).

### 5.2.2 - A criação manual

Uma vez demonstrada a ferramenta utilizada, podemos proceder aos mecanismos utilizados para alimentar a ontologia.

Para a construção da nossa ontologia, utilizamos como base a seqüência de passos sugerida por Noy e McGuinness (2001), que consiste de perguntas e atividades a serem realizadas durante a construção da ontologia. A seguir, as respostas do nosso problema dadas as perguntas e tarefas sugeridas em Noy e McGuinness (2001).

#### 5.2.2.1 - Passo 1 - Determinação do domínio e escopo da ontologia

Perguntas:

1) Qual o domínio a ontologia cobrirá?

R.: Conceitos básicos da linguagem Java

2) Para que iremos utilizar a ontologia?

R.: Para apoiar a busca e recuperação de informação em uma base de dados de comunidades de prática virtuais sobre a linguagem Java.

3) Para que tipos de questões a informação na ontologia deve fornecer respostas?

R.: A ontologia não responderá diretamente aos questionamentos enviados para a comunidade, no entanto ela servirá como base de conhecimento adicional aos

conceitos tratados na comunidade, fornecendo conceitos similares e resultados da combinação de conceitos em uma mesma mensagem.

4) Quem utilizará e manterá a ontologia?

R.: A ontologia será diretamente utilizada pelos agentes de software responsáveis pelo fornecimento de informações relacionadas aos questionamentos enviados pelos membros da comunidade. Inicialmente a ontologia possuirá uma versão inicial constituída pela alimentação manual e automática (analisada na próxima seção), no entanto, para que os resultados de busca fossem cada vez mais satisfatórios, seria necessário que um especialista do domínio na comunidade, neste caso da linguagem Java, fosse o responsável pela manutenção da ontologia.

#### 5.2.2.2 - Passo 2 – Considerar a reutilização de ontologias

Inúmeras pesquisas foram feitas com o objetivo de localizar ontologias semelhantes, fossem da linguagem Java, linguagens orientadas a objetos ou simplesmente uma ontologia de linguagem de programação. Nada parecido foi encontrado, desta maneira optou-se por construir uma ontologia do zero, que possivelmente poderá servir de base à construção de novas ontologias.

Este ponto foi bastante importante na elaboração da ontologia. Durante todo o processo de análise do sistema, o foco era na linguagem Java, com o aprimoramento dos conhecimentos sobre construção de ontologias pudemos perceber que seria bastante elegante e útil construir uma ontologia que possuísse maior reusabilidade. A partir deste ponto o foco da construção da ontologia mudou, da construção de uma ontologia de Java, para construção de uma ontologia de linguagens de programação orientadas a objeto. Tendo uma ontologia deste tipo, facilmente a linguagem Java poderia ser instanciada, assim como outras linguagens orientadas a objeto. Assim, os aspectos pertinentes à linguagem Java passaram a ser instâncias da ontologia principal.

#### 5.2.2.3 - Passo 3 – Enumerar termos importantes da ontologia

Aqui começa a elaboração da ontologia propriamente dita. Os passos anteriores serviram de análise inicial ao processo. O objetivo inicial da definição dos termos importantes da ontologia é definir um ponto de partida. No momento da

execução de outras tarefas de construção da ontologia, vários outros termos são reconhecidos, alguns são descartados, durante o processo de elaboração da ontologia podem ocorrer mudanças bastante significativas, em virtude do conhecimento cada vez maior dos conceitos envolvidos e dos objetivos desejados.

A seleção dos termos importantes ao domínio é de relevância essencial à construção da ontologia, pois a partir deste ponto é que será feito o reconhecimento dos conceitos e propriedades presentes na estrutura. Na figura 12 vemos a primeira listagem, de termos básicos relacionados a linguagens orientadas a objetos, que foi gerada, com o objetivo de fornecer embasamento inicial à ontologia. Não é uma lista completa, no entanto possui muitos termos importantes que posteriormente transformaram-se em conceitos na ontologia.

| | | | |
|---|---|---|---|
| classes | objetos | herança | polimorfismo |
| variáveis | métodos | atributos | funções |
| instância | alocação | encapsulamento | constantes |
| operador lógico | desvio condicional | estrutura de controle | recursão |
| tipos de dados | sobrecarga de operador | ponteiros | comentário |
| exceção | operador relacional | qualificador | declaração |

**Figura 12 – Termos importantes da ontologia**

### 5.2.2.4 - Passo 4 – Definir classes e hierarquias de classes

Noy e McGuinness (2001) consideram três tipos de abordagem para a definição da hierarquia de classes: abordagem top-down, abordagem bottom-up e uma combinação das duas abordagens. Na abordagem *top-down* a determinação das primeiras classes começa com a definição da classe mais genérica e a subseqüente especialização dos conceitos. Na abordagem *bottom-up* é realizado o procedimento inverso, partindo-se de conceitos mais especializados para conceitos mais genéricos. A abordagem combinada é uma mistura das duas metodologias, onde o engenheiro de ontologias ora parte no sentido generalização especialização, ora vai do conceito mais especializado para o mais genérico.

A abordagem combinada se mostrou mais útil ao nosso procedimento de construção. Começamos definindo uma classe "Concept" que é a classe mãe de todas as outras classes, de maneira que caso fosse necessário a classe Concept abrigaria

propriedades que seriam comuns a todas as demais classes da ontologia. A partir deste ponto a criação de classes foi bastante natural, a cada termo analisado, a classe correspondente era identificada, e inserida na ontologia. Se fosse verificada uma existência de herança, a classe era inserida como uma subclasse da classe pai correspondente. O mesmo acontecia de maneira oposta, caso uma classe mais especializada tivesse sido inserida primeiro, uma nova classe que fosse uma generalização da anterior era inserida como classe pai da mesma. Classes com relação de herança são as classes cuja relação é estabelecida pela expressão "A classe B é um tipo de classe A", onde B herda as características de A. Neste momento, também já é possível visualizar relação entre classes onde a relação não é algo do tipo generalização especialização, no entanto essas classes possuem algum relacionamento, que na ontologia é representado por propriedades, ou *slots*. Este é o próximo passo da construção da ontologia, a determinação de propriedades das classes.

Optamos por definir os nomes das classes em inglês por ser uma língua mais amigável à nomeação de variáveis, e por definir melhor expressões com um menor número de letras, além de ser uma língua que pode ser compreendida pela comunidade científica, e ao ser disponibilizada a ontologia na internet, esta possa ser mais facilmente reutilizada.

#### 5.2.2.5 - Passo 5 – Definir propriedades das classes

O Protegé permite que sejam definidas propriedades de dois tipos: propriedades cujo valor é um tipo de dado (string, booleano, numérico...) e propriedades cujo valor é um objeto. Como exemplo de propriedades de tipos de dados, podemos citar por exemplo a propriedade número de filhos da classe pessoa. A propriedade numeroFilhos, espera receber como valor um dado numérico. As propriedades que recebem um objeto como valor, são as de maior importância para este trabalho. São elas que estabelecem as relações entre classes que não são do tipo herança. Como exemplo de propriedades que recebem um objeto como valor podemos citar o exemplo já mencionado, de linguagens de programação e variáveis. Afirmar que "Linguagens de programação possuem variáveis" pode ser representado em uma ontologia com duas classes: LinguagemProgramacao e Variavel, relacionadas pela propriedade "possui".

### 5.2.2.6 - Passo 6 – Definir as características das propriedades

Cada propriedade da ontologia que recebe um objeto como valor, possui um domínio e um alcance. Tanto o domínio quanto o alcance estão representados por classes existentes na ontologia. Então, um exemplo de domínio e alcance para a propriedade "has" são as classes ProgrammingLanguage e Variable. ProgrammingLanguage é um dos domínios de "has", enquanto que Variable é um dos alcances de "has".

Em nossa ontologia, a propriedade "has" possui uma característica bastante peculiar. Ela relaciona várias classes a várias outras. Ou seja, tanto o domínio quanto o alcance de "has" possuem várias classes. Exemplo:

"API has Interface"

"ClassLibrary has Package"

"Class has Method"

O domínio e o alcance de uma propriedade são representados por dois conjuntos, onde cada conjunto é uma lista de conceitos da ontologia. A figura 13 apresenta alguns conceitos dos conjuntos de domínio e alcance da propriedade "has".

| Domínio |
|---|
| Class |
| OOLanguage |
| ClassLibrary |
| Package |
| Api |
| Object |
| Method |
| Variable |
| Field |

| Alcance |
|---|
| Field |
| Heritage |
| Method |
| Package |
| Command |
| Declaration |
| Comment |
| DataType |
| Data |

**Figura 13 – Domínio e Alcance da propriedade "has"**

O conteúdo da figura 13 em relação à propriedade "has" significa que, um dos termos do quadro de Domínio está relacionado a um ou mais termos do quadro de alcance. Em OWL, na definição da propriedade, estes conjuntos são representados através da união de classes do domínio, e da união de classes do alcance, através da

propriedade "unionOf". Porém, como identificar na ontologia, quais termos do quadro de alcance estão relacionados aos termos do quadro de domínio? A resposta está na definição de restrições sobre propriedades, mais um passo importante na construção da ontologia.

A definição de restrições sobre propriedades é feita através de quantificadores e operações sobre conjuntos. Assim, podemos definir, que, na existência de uma propriedade que possua alcances múltiplos, o alcance só será válido para conceitos pré-definidos. As restrições sobre propriedades são definidas em cada um dos conceitos. Como exemplo analisemos a relação entre "Class" e "Method". Sabemos que o conceito "Class" está relacionado através da propriedade "has" com os conceitos "Method" e "Field", pois classes podem possuir métodos e campos. Assim, o domínio "Class" de "has" está relacionado a dois conceitos do alcance de "has". Para indicar que "Class" está relacionado somente com "Method" e "Field" deveremos criar uma restrição, que indique que "Class" só se relaciona com esses dois conceitos e mais nenhum outro. Isto pode ser feito através da restrição "allValuesFrom" ou ∀. Assim, podemos dizer que a propriedade "has" se aplica ao conceito "Class" para toda classe "Method" ou para toda classe "Field". A relação com o conceito "Method" está demonstrada na expressão OWL da figura 14.

```
<owl:Restriction>
   <owl:onProperty>
      <owl:ObjectProperty rdf:about="#has"/>
   </owl:onProperty>
   <owl:allValuesFrom>
      <owl:Class rdf:about="#Method"/>
   </owl:allValuesFrom>
</owl:Restriction>
```

**Figura 14 – Exemplo de restrição sobre propriedades em OWL**

### 5.2.2.7 - Passo 7 – Criar as instâncias

O último passo é a criação das instâncias da ontologia. No nosso caso, as instâncias serão formadas por aspectos pertinentes a uma linguagem de programação específica, aqui a linguagem Java. Entretanto, como dito nos passos anteriores, a ontologia está preparada para receber a instanciação de qualquer linguagem de programação orientada a objetos. É na criação das instâncias que entram as

construções automáticas utilizadas neste trabalho. Parte das instâncias foram inseridas manualmente, assim como os conceitos da ontologia. As instâncias inseridas manualmente estão basicamente representadas pelos comandos básicos da linguagem Java. Exemplos: "if", "while", "for", "break" são instâncias da classe "command" e foram inseridos manualmente. Estes são comandos nativos da linguagem, e não mudarão, a não ser com o advento do lançamento de uma nova versão. No entanto, com a expansão da linguagem Java, e a grande diversidade de bibliotecas distribuídas livremente pela Internet, a cada dia surgem novas APIs para propósitos específicos. Em geral, uma API é descrita por uma documentação chamada JavaDoc (JAVADOC, 2007). O JavaDoc possui a listagem de todos os pacotes, classes e métodos que a API possui, entre outros componentes, descrevendo a funcionalidade de cada um desses itens. Reconhecemos no JavaDoc uma ótima maneira de criar instâncias automaticamente, para que um maior conteúdo pudesse ser adicionado à ontologia de maneira rápida e prática.

Na seção seguinte demonstraremos como o JavaDoc foi utilizado para o enriquecimento da ontologia através da criação de instâncias automaticamente.

### 5.2.3 - A criação automática - JavaDoc

A linguagem Java dispõe de uma poderosa ferramenta de documentação, o JavaDoc. O JavaDoc é uma aplicação que permite catalogar todas as informações sobre um programa Java. JavaDoc é também o nome do documento gerado por esta aplicação. Informações sobre pacotes, classes e métodos são geradas automaticamente, mas o programador também pode adicionar meta-informações ao seu código fonte, tornando a documentação mais rica. Além disso, o JavaDoc é gerado em formato próprio para a Internet, sendo assim, se torna bastante acessível a outros programadores que desejem utilizar a documentação.

Freqüentemente, quando um programador Java possui alguma dúvida em relação à determinada funcionalidade de uma biblioteca Java, ele utiliza o JavaDoc da biblioteca como fonte de consulta.

O JavaDoc é um documento estruturado e de fácil compreensão por máquina e suas informações estão estruturadas nos seguintes níveis:

1. API - *Application Program Interface*, o nível macro de organização do JavaDoc, uma API é formada por vários pacotes (*packages*)

2. Pacotes - Um pacote é um conjunto de classes e interfaces
3. Interfaces - Uma Interface é um protocolo de comportamento de classes, sendo assim suas características serão bastante semelhantes às de uma classe
4. Classes - Conjunto de métodos, variáveis e constantes

Sendo um documento de tamanha importância para a comunidade de usuários Java, optamos então por incorporá-lo também ao protótipo aqui desenvolvido. Assim, quando a busca fosse realizada poderia também se beneficiar das informações presentes no JavaDoc de cada API.

Um novo agente foi criado com o propósito de "alimentar" a ontologia com o JavaDoc de cada API que se desejasse. O AJD (Agente Java Doc) não é acionado automaticamente, mas, por uma solicitação direta, informando-se ao agente o endereço da API na Internet. O agente lê todo o conteúdo documental da API, inserindo informações sobre pacotes, interfaces, classes e métodos pertinentes à mesma.

Por serem informações bastante específicas, optamos por inserir o conteúdo do JavaDoc nas instâncias da ontologia. Já constavam na ontologia os conceitos: API, Interface, Classe, Método... entretanto, cada JavaDoc apresenta inúmeros pacotes, classes, entre outros. Consideramos então, que cada JavaDoc de uma API, seria uma instância do conceito "API". Na tabela 1 vemos alguns exemplos de conceitos e suas respectivas instâncias.

| *Conceito (Classe da ontologia)* | *Instância* |
|---|---|
| API | Java 1.4.2 API |
| Pacote | java.applet, java.awt, java.beans... |
| Interface | AppletContext, AppletStub, AudioClip.. |
| Classe | Applet, Beans, String... |
| Método | getInstanceOf(), instantiate(), create()... |

**Tabela 1 – Conceitos e instâncias decorrentes da interpretação do JavaDoc**

Devido ao fato de serem inseridas como conceitos e instâncias da ontologia, as informações presentes no JavaDoc se tornaram disponíveis para consulta e análise no processo de busca. Como no processo de expansão de consultas os conceitos e instâncias são analisados de maneira semelhante e transparente, não houve

diferenciação no tratamento das informações inseridas manualmente, ou automaticamente na ontologia.

Através de uma consulta que utilizasse instâncias que foram inseridas pelo processamento do JavaDoc, pode-se saber por exemplo que o pacote java.applet é um pacote da API básica do Java, a Java 1.4.2 API, que possui a classe Applet, as interfaces AppletContext, AppletStub, entre outras valiosas informações.

## 5.3 - Expansão da consulta

Como visto no capítulo anterior, vários trabalhos analisam propostas de expansão de consultas com o objetivo de melhorar a relevância dos documentos retornados.

Tendo o nosso domínio de assunto tão bem definido e dispondo de uma estrutura de conceitos inter-relacionados sobre este domínio, nada mais natural do que utilizar a ontologia criada para expandir a consulta original. Outros métodos de expansão de consultas são descritos no capítulo 4.

A pergunta que norteia a pesquisa para descoberta de quais mecanismos devem ser utilizados para expandir uma consulta é: "Como posso expandir uma consulta de maneira a aumentar a relevância dos resultados retornados, porém sem prejudicar a precisão dos meus resultados em relação à consulta original".

O processo de descoberta da consulta expandida ideal está claramente entre dois limites, onde, aumentar a quantidade de termos em uma consulta faz com que mais resultados sejam trazidos, porém, pode fazer também com que informação desnecessária seja recuperada. Sendo assim, a qualidade de uma consulta expandida dependerá diretamente da seleção de seus novos termos.

### 5.3.1 - A utilização da ontologia

Uma vez estabelecido que a ontologia seria nosso artefato para a expansão da consulta, a pergunta que se segue é: Como utilizar a ontologia para a expansão?

O trabalho presente em Gonzales (2000) elucida uma série de propriedades que relacionam termos e que podem ser utilizadas na formação de um Léxico Gerativo de sistemas de busca e recuperação de informação. Como visto e definido no capítulo anterior, selecionamos as seguintes propriedades para compor o nosso mecanismo de geração:

- Sinonímia
- Especialização
- Co-Herança
- Associação
- Equivalência
- Decomposição
- Agregação

Para obter os termos equivalentes por sinonímia, pegaremos os sinônimos dos termos definidos na ontologia. Na especialização serão recuperados as subclasses ou "filhos" dos conceitos selecionados. Na co-herança selecionaremos os conceitos que possuem em comum a mesma superclasse com o conceito analisado, desta maneira selecionaremos os "irmãos" do conceito atual. Através da associação serão trazidos todos os conceitos que possuem algum relacionamento com o conceito analisado através de propriedades diretas. Na equivalência a superclasse do conceito será considerada na expansão da consulta. A decomposição é obtida através da recuperação de conceitos que são relacionados com o conceito original através de uma propriedade do tipo "contém", e a agregação é o oposto a esta propriedade. Podemos notar que decomposição e agregação são especializações da propriedade de associação (que possui todos os tipos de relacionamento). Sendo assim, não se faz necessário considerar aqui os conceitos de decomposição e agregação como um tipo a parte.

Chegamos então a cinco propriedades principais: sinonímia, especialização, co-herança, associação e equivalência.

Em uma visão bastante geral, um algoritmo de expansão de consulta utilizando ontologia seria o demonstrado na figura 15.

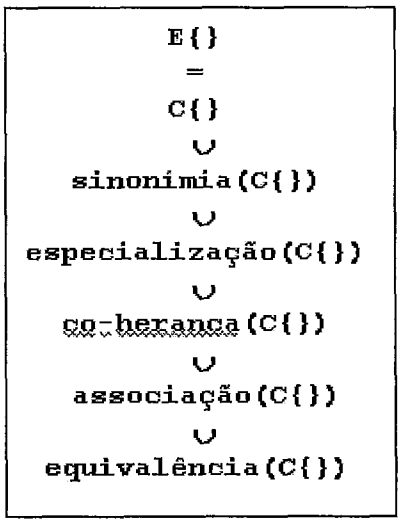



**Figura 15 – Algoritmo de expansão de consulta**

Onde C{ } é o conjunto de termos da consulta original e E{ } é o conjunto de termos da consulta expandida.

Esta abordagem, embora bastante prática, pode trazer resultados indesejáveis. Isto porque termos que foram recuperados do relacionamento podem acabar apresentando-se mais importantes do que os termos originais da consulta, fazendo com que os documentos recuperados pela consulta expandida apresentem resultados que não são compatíveis com os anseios originais daquele que submeteu a consulta.

Em um exemplo prático, a consulta na ontologia dado o conceito "Array" retornaria através das operações descritas acima, os seguintes termos relacionados:

```
Boolean, Byte, Character, Integer, Real, array, matriz, matrix,
vector, vetor, Primitive
```

Podemos observar claramente que documentos que contivessem quaisquer um dos termos acima seriam trazidos na mesma proporção que os documentos que contivessem o termo "Array", o que deixaria o membro da comunidade que só queria saber sobre o tipo de dados Array bastante decepcionado.

Mas então como expandir a consulta de maneira a ajudar o usuário, e deixá-lo satisfeito, procurando trazer o mínimo de resultados irrelevantes?

A resposta está no fato de que os termos expandidos na consulta não podem ser tratados com a mesma importância que os termos presentes na consulta original.

A solução para este problema é aplicar pesos nos termos da consulta. Os termos da consulta original deverão sempre ter peso superior aos pesos dos termos descobertos por expansão, sendo assim os termos descobertos por expansão não se sobreporão aos termos originais da consulta.

### 5.3.2 - Determinação de pesos dos termos da consulta

Como vimos no capítulo anterior, a expansão de consulta sem atribuição de pesos aos termos expandidos pode ao invés de trazer benefícios, prejudicar a consulta, tornando-a distante da consulta original e trazendo resultados que estão longe do que seria aceitável pelo usuário.

O problema da atribuição de pesos é: como determinar pesos que façam com que os termos digitados pelo usuário sejam importantes, ao mesmo tempo que a consulta expandida ajude a recuperar mais documentos relevantes que constam na coleção, e que não seriam recuperados caso não houvesse expansão de consulta.

Em Gonzalez, Strube de Lima (2001) é feita uma análise sobre a expansão de consultas em vários níveis e com a utilização de um tesauros. A expansão de consultas em vários níveis é feita da seguinte maneira: a partir do conjunto $C_0$ dos termos originais da consulta é obtido um conjunto $C_1$ de termos expandidos. Iterativamente são gerados conjuntos Ci a partir dos termos anteriormente obtidos. Cada iteração i obtida é determinada como sendo um nível de expansão. Uma matriz de co-ocorrência de termos é utilizada para atribuir pesos a cada um dos termos obtidos. Se um termo é obtido pela expansão de mais de um termo, este termo apresentará peso maior em relação aos outros termos.

Nos resultados do trabalho de Gonzales e Lima, pôde ser observado que as expansões de nível 1 apresentaram qualidade de resultados superior às configurações de nível zero (sem expansão) e nível três (após três expansões) como pode ser observado na figura 16.

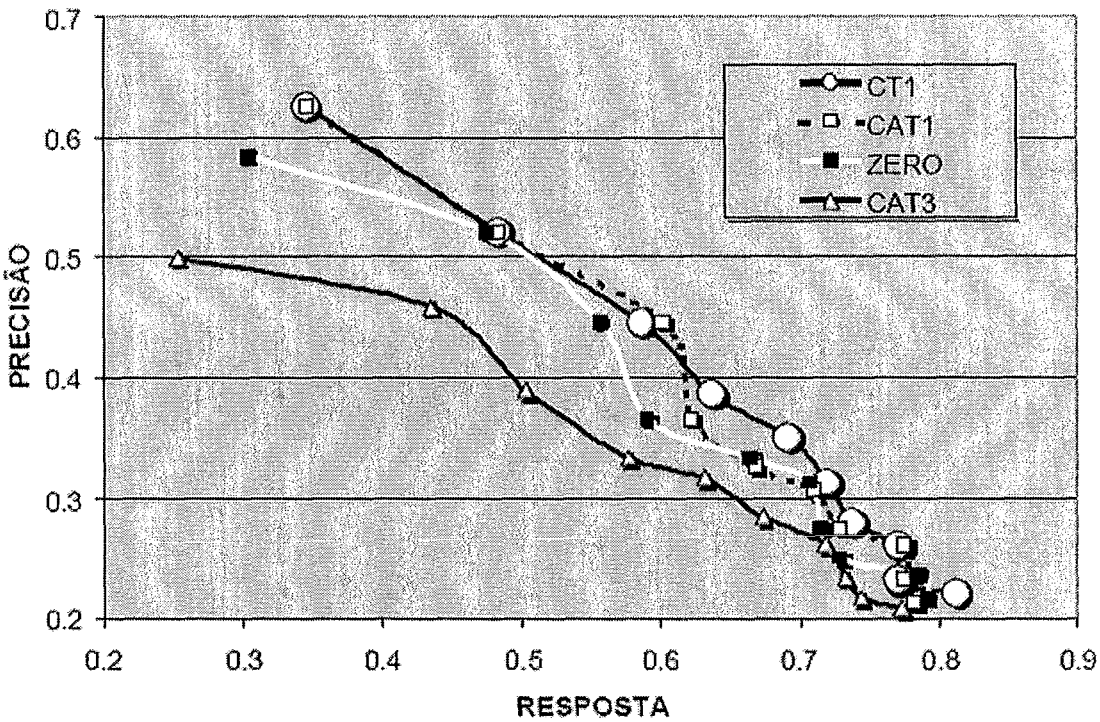



**Figura 16 – Níveis de precisão para expansão de consulta em vários níveis (GONZALEZ, STRUBE DE LIMA, 2001)**

Na figura 16 as configurações CT1 e CAT1 apresentam nível de expansão 1 (com pequenas diferenças inerentes ao experimento de Gonzales e Lima), as configurações ZERO e CAT3 apresentam respectivamente níveis de expansão 0 (sem expansão) e 3 (3 níveis de expansão).

Podemos observar que a expansão de nível 3 apresenta o pior desempenho se comparada às outras configurações de expansão. A configuração CT1 (1 nível de expansão) é a que apresenta melhor resultado, sobre os níveis considerados de resposta.

Como na conclusão do trabalho, a expansão de nível 1 apresentou o melhor resultado, adotaremos apenas um nível de expansão. Consideramos este um resultado bastante intuitivo, uma vez que muitos níveis de expansão podem fazer com que a consulta expandida fique muito distante da consulta original do usuário.

Partindo então de uma busca em um único nível optamos por adaptar os princípios de atribuição de pesos do trabalho de Gonzales, de maneira que os pesos não sejam dados em função do nível de expansão, mas da importância (prioridade) que o termo apresenta de acordo com a maneira pelo qual ele foi obtido.

No trabalho de Gonzales, cada nível que se expande, representa um distanciamento dos termos originais da consulta, sendo assim, termos no nível zero possuem peso maior do que termos no nível 1 que possuem peso maior que termos no nível 2 e assim por diante.

Utilizando o paralelo com os níveis, na nossa proposta de expansão, podemos identificar qual é o peso relativo de cada termo de acordo com o seu mecanismo de expansão.

A consulta original do usuário é o nível zero, sendo assim, deve possuir maior peso em relação a qualquer outro termo de expansão (isto em linhas gerais, uma vez que um termo expandido pode ser mencionado por mais de uma expansão, fazendo com que este fique com um peso alto podendo até se igualar ao peso dos termos originais). Em nossa proposta, a consulta é expandida por dois mecanismos: expansão por sinonímia e a identificação de outros relacionamentos semânticos através da ontologia (especialização, co-herança, associação e equivalência).

Ao invés dos níveis de Gonzales e Lima, utilizaremos ordens de prioridade. Sendo assim, os termos originais e expandidos terão ordem de prioridade de acordo com a tabela 2.

| | *Tipo de Expansão* | *Ordem de prioridade* |
|---|---|---|
| antes da expansão | Termos originais que representam conceitos na ontologia (zero expansão) | 1 |
| | Termos originais que não estão presentes na ontologia | 2 |
| após expansão | Sinônimos | 2 |
| | Relacionamentos semânticos | 3 |

**Tabela 2 – Relação entre tipo de expansão e graus de importância**

De acordo com a tabela 2 podemos observar que os termos que estão presentes na consulta e que representam conceitos na ontologia serão mais importantes de acordo com a escala de prioridade. Ainda sem expansão, além dos termos originais da consulta e que figuram na ontologia, temos também aqueles termos que foram utilizados pelo usuário, porém que não constam na ontologia. Estes termos podem ser verbos, outros substantivos, palavras que não foram reconhecidas como *stopwords* ou

até mesmo um termo importante, que deveria constar na ontologia, mas que atualmente não figura como conceito. É importante dar valor a estes termos, uma vez que o usuário não tem controle sobre a ontologia, e independentemente do conceito constar na ontologia ou não, ele desejará um bom resultado em sua busca. O fato dos termos digitados pelo usuário serem mais importantes fará com que, mesmo que um conceito não seja reconhecido, o resultado da busca não fique prejudicado. Assim, estes termos possuirão ordem de prioridade dois.

Na atribuição dos pesos aos termos obtidos após a expansão, serão obtidos termos com maior importância e menor importância. Os sinônimos possuem evidentemente grande importância, uma vez que são uma extensão do vocabulário utilizado pelo usuário. Uma dos grandes entraves da expansão de consultas é a possibilidade dos conceitos expandidos distorcerem a informação original da consulta. Consideramos que a expansão por sinonímia possui um baixo poder de distorcer o sentido original, uma vez que conceitualmente, os termos apresentarão os mesmos significados dos termos originais. Sendo assim, os sinônimos podem ser considerados praticamente como se tivessem sido digitados pelo usuário, o que fez com que atribuíssemos a eles a mesma ordem de prioridade dos termos originais que não constam na ontologia, ou seja, ordem dois.

Por último, temos a obtenção dos relacionamentos semânticos. Estas sim, por apresentarem operações de especialização e generalização, têm o poder de distorcer o objetivo original da consulta. Sendo assim, apresentarão ordem de prioridade três, prioridade relativa menor do que qualquer outro tipo de termo.

Munidos então das ordens de prioridade de cada termo, construiremos então uma matriz, onde serão calculados os pesos relativos de cada termo na consulta.

A figura 17 mostra uma matriz de T colunas por T + E linhas. Onde T é o número de termos que representam conceitos na ontologia, e E o número de termos expandidos.

Em uma consulta hipotética onde os termos T1, T2, T3 e T4 foram reconhecidos como conceitos na ontologia, os termos de T5 a T11 foram expandidos a partir dos termos originais de acordo com a tabela 3.

| | *T1* | *T2* | *T3* | *T4* |
|---|---|---|---|---|
| Sinônimos | | T5 | T6 | T7 |
| Especialização | T8 | | | |
| Co-herança | | T9 | T9 | |
| Associação | T10 | | | |
| Equivalência | | | T11 | |

**Tabela 3 – Termos expandidos e seus termos de origem**

Na matriz da figura 17 cada elemento é considerado um peso parcial do termo na consulta. Nos termos originais, este peso é dado pela fórmula $p_{i,j} = C$, onde C é uma constante, à qual foi atribuído o valor 10 por simplicidade de cálculo. Nos termos expandidos cada peso parcial será calculado pela fórmula $p_{i,j} = C/n\text{-}1$, caso a coluna indique o termo original que deu origem ao termo expandido. A variável n é a ordem de prioridade de cada termo. Caso o termo original não tenha qualquer relação com o termo expandido, o valor da célula será 0/n, ou simplesmente 0.

Nesta matriz podemos observar que T9 é obtido tanto através de T2 quanto através de T3, sendo assim, possuirá peso final maior em relação à outros termos de mesma categoria.

O peso total de cada termo será dado pelo somatório das linhas da matriz, e será normalizado para que o menor peso seja equivalente a um peso de valor 1.

| | | | T1 | T2 | T3 | T4 | peso total | peso norm |
|---|---|---|---|---|---|---|---|---|
| T termos originais | Termos orig. ontologia | T1 | p(1,1)=10 | p(1,2)=10 | p(1,3)=10 | p(1,4)=10 | 40 | 8 |
| | | T2 | p(2,1)=10 | p(2,2)=10 | p(2,3)=10 | p(2,4)=10 | 4 | 8 |
| | | T3 | p(3,1)=10 | p(3,2)=10 | p(3,3)=10 | p(3,4)=10 | 40 | 8 |
| | | T4 | p(4,1)=10 | p(4,2)=10 | p(4,3)=10 | p(4,4)=10 | 40 | 8 |
| E termos expandidos | Sinônimos | T5 | p(5,1)=0/1 | p(5,2)=0/1 | p(5,3)=10/1 | p(5,4)=0/1 | 10 | 2 |
| | | T6 | p(6,1)=0/1 | p(6,2)=0/1 | p(6,3)=0/1 | p(6,4)=10/1 | 10 | 2 |
| | | T7 | p(7,1)=0/1 | p(7,2)=0/1 | p(7,3)=0/1 | p(7,4)=10/1 | 10 | 2 |
| | Especializ. | T8 | p(8,1)=10/2 | p(8,2)=0/2 | p(8,3)=0/2 | p(8,4)=0/2 | 5 | 1 |
| | Co-herança | T9 | p(9,1)=0/2 | p(9,2)=10/2 | p(9,3)=10/2 | p(9,4)=0/2 | 10 | 2 |
| | Associação | T10 | p(10,1)=10/2 | p(10,2)=0/2 | p(10,3)=0/2 | p(10,4)=0/2 | 5 | 1 |
| | Equivalência | T11 | p(11,1)=0/2 | p(11,2)=0/2 | p(11,3)=10/2 | p(11,4)=0/2 | 5 | 1 |

**Figura 17 – Matriz de atribuição de pesos aos termos da consulta**

Uma vez expandida nossa consulta, e os pesos de seus termos determinados, teríamos que providenciar uma maneira para que ela fosse submetida exatamente como calculamos, caso contrário, o trabalho teria sido desperdiçado.

Como nossa proposta está focada na utilização da ontologia para expansão da consulta, e não no mecanismo de busca em si, optamos por utilizar nossa consulta em algum mecanismo de busca e armazenamento já conhecido. Estaria fora do escopo desta dissertação desenvolver uma ferramenta própria de armazenamento.

Com o objetivo de não comprometer a compreensão do assunto aqui abordado, a descrição detalhada dos mecanismos de recuperação de informações da ferramenta de armazenamento está descrita no Apêndice D.

## 5.4 - Conclusão

Neste capítulo descrevemos todos os componentes que formam a arquitetura proposta por esta dissertação. Iniciamos com uma arquitetura multi-agente que é a responsável pelas ações tomadas pelo sistema. Em seguida mostramos a construção de uma ontologia desde o início, analisando desde a seleção de termos iniciais manualmente, até mecanismos automáticos de alimentação da ontologia. Mostramos também como a ontologia foi utilizada para expandir a consulta inicial submetida pelo usuário, gerando uma nova consulta. E finalmente fizemos uma extensa análise de atribuição de pesos aos termos expandidos da nova consulta.

No próximo capítulo analisaremos os mecanismos de avaliação que foram utilizados para testar os resultados produzidos pelo nosso protótipo.

# Capítulo 6 – COOPRACTICE – Avaliação

Nos capítulos anteriores fizemos uma descrição de todo o embasamento teórico que sustentava nossa proposta, assim como a descrição da arquitetura que utilizou estes fundamentos até culminarmos na criação do protótipo desta dissertação.

Neste capítulo avaliaremos o comportamento do nosso protótipo através de um experimento, com o objetivo de comparar os resultados obtidos através da avaliação de voluntários de respostas a consultas enviadas ao sistema, considerando a utilização e não utilização da ontologia.

## 6.1 - Descrição do experimento

Para os procedimentos de avaliação do COOPRACTICE foi realizado um experimento, que contou com a colaboração de onze alunos de mestrado da cadeira de Busca e Recuperação de Informações da linha de Banco de Dados do Programa de Engenharia de Sistemas da COPPE – UFRJ.

Para a realização do experimento, os participantes dispunham de quatro horas, sendo assim, a quantidade de perguntas e avaliações que os participantes fizeram teve que ser adaptada a este período.

Os participantes contaram com uma interface, que os possibilitava:

- Enviar perguntas ao agente
- Avaliar perguntas respondidas pelo agente

A massa de dados do experimento foi obtida através de um processo de leitura das mensagens trocadas no grupo de discussão sobre Java Básico do Grupo de usuários Java (GUJ, 2006). O GUJ é uma comunidade de Java bastante conceituada na Internet. Criado em 2002 o GUJ conta com mais de dez mil usuários, um total de cento e cinqüenta artigos sobre Java e mais de cento e noventa mil mensagens trocadas em seu fórum de mensagens sobre Java, divididas em várias categorias.

Para este experimento foram catalogadas um total de 19183 mensagens, sendo 2649 tópicos de mensagens. Neste conjunto de mensagens constam as dúvidas mais freqüentes em iniciantes na linguagem Java, caracterizando um conjunto bastante significativo para que se pudesse obter resultados satisfatórios no que diz respeito às perguntas submetidas pelos participantes.

## 6.2 - Etapas do experimento

O experimento foi realizado em cinco etapas. Em cada uma dessas etapas foi solicitado aos participantes que realizassem as tarefas determinadas. A seguir a relação de cada uma dessas etapas.

## 6.3 - Primeira Etapa – Cadastro de participantes

Foi solicitado aos participantes que fizessem um cadastro básico para que pudessem acessar o sistema. Os usuários informaram seus nomes completos, e-mail e senha, para que pudessem ter acesso exclusivo às funcionalidades do sistema, e que suas avaliações e informações inseridas pudessem ser unicamente identificadas. A figura 18 mostra o cadastro de usuários

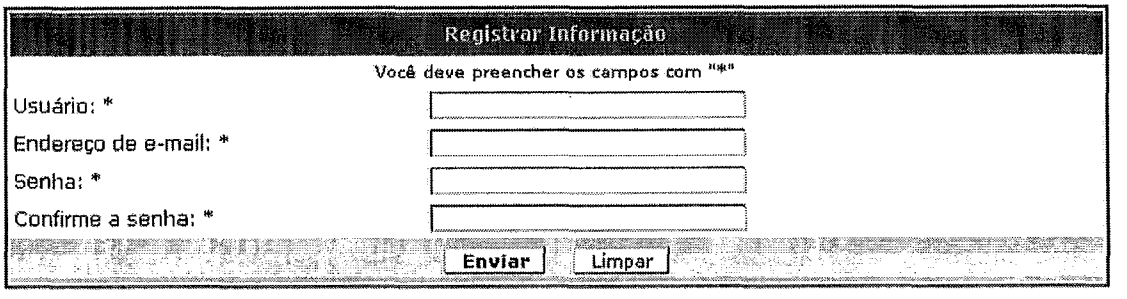



**Figura 18 – Cadastro de participantes**

## 6.4 - Segunda Etapa – Perfil dos participantes

Após o cadastro inicial, foi pedido a cada um dos participantes que informassem sua proficiência em Java como pode ser visto na figura 19. Como cada um dos participantes iria analisar informações sobre o domínio da linguagem Java, esta etapa foi de extrema importância, para avaliar qual era o conhecimento de cada avaliador sobre a linguagem. A informação inicial que tínhamos era que todos os participantes possuíam algum conhecimento sobre Java.

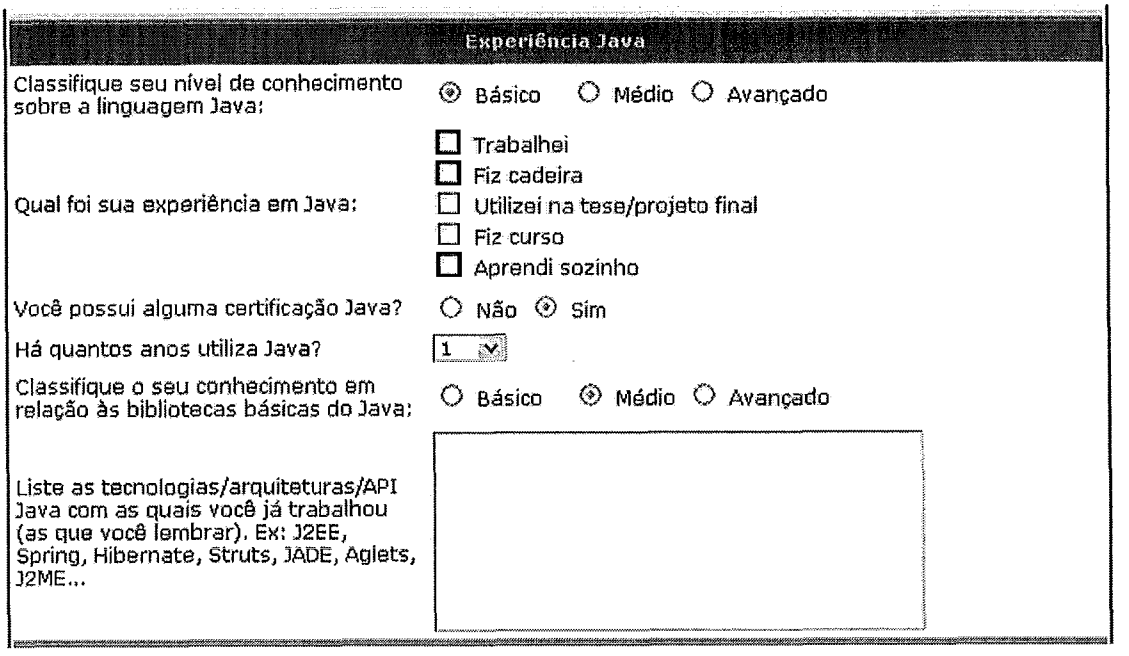



**Figura 19 – Perfil dos participantes**

Foram feitas então as seguintes perguntas:

1. Classifique seu nível de conhecimento sobre a linguagem Java(Básico/Médio/Avançado)
2. Qual foi sua experiência com Java? (Múltiplas opções)
    a. Trabalhei
    b. Fiz cadeira
    c. Utilizei na tese/projeto final
    d. Fiz curso
    e. Aprendi sozinho por livre e espontânea vontade
3. Há quantos anos você utiliza Java?
4. Classifique o seu conhecimento em relação às bibliotecas básicas do Java (Básico/Médio/Avançado)
5. Liste as tecnologias/arquiteturas/API Java com as quais você já trabalhou (as que você lembrar). Ex: J2EE, Spring, Hibernate, Struts, JADE, Aglets, J2ME...

Para cada uma das perguntas foram obtidos os seguintes resultados:

1. **Nível de conhecimento sobre a linguagem Java, tabela 4:**

| *Conhecimento* | *Número de Participantes* | *Porcentagem* |
|---|---|---|
| Básico | 4 | 36.36% |
| Médio | 6 | 54.54% |
| Avançado | 1 | 9.09% |

**Tabela 4 - Nível de conhecimento dos participantes**

2. **Experiência com Java, tabela 5:**

| *Experiência* | *Número de Participantes* | *Porcentagem* |
|---|---|---|
| Aprendeu sozinho | 3 | 27.27% |
| Fez curso | 3 | 27.27% |
| Usou em cadeira | 8 | 72.72% |
| Usou na tese | 3 | 27.27% |
| Usou no trabalho | 6 | 54.54% |

**Tabela 5 - Experiência com Java**

3. **Há quantos anos utiliza Java, tabela 6:**

| *Anos que trabalha com Java* | *Número de participantes* | *Porcentagem* |
|---|---|---|
| 0 | 1 | 9.09% |
| 1 | 2 | 18.18% |
| 2 | 3 | 27.27% |
| 3 | 3 | 27.27% |
| 4 | 2 | 18.18% |

**Tabela 6 - Anos que trabalha com java**

4. **Conhecimento das bibliotecas básicas da linguagem Java, tabela 7:**

| *Conhecimento* | *Número de Participantes* | *Porcentagem* |
|---|---|---|
| Básico | 4 | 36.36% |
| Médio | 6 | 54.54% |
| Avançado | 1 | 9.09% |

**Tabela 7 - Conhecimento das bibliotecas básicas**

5. **Tecnologias com as quais trabalhou em Java**

As tecnologias mencionadas foram: J2EE, J2ME, Struts, Hibernate, AspectJ, Swing, EJB. Dos onze participantes, dez mencionaram já ter utilizado a arquitetura J2EE.

A partir destes resultados, pudemos constatar que os participantes possuíam um conhecimento bastante razoável da linguagem Java, e que as avaliações dadas seriam coerentes com as avaliações de profissionais da área.

## 6.5 - Terceira Etapa – Perguntas dos participantes

Nesta etapa, foi solicitado que no fórum: "Suas perguntas sobre Java Básico", cada participante cadastrasse cinco perguntas. Foi explicado aos usuários que estas perguntas deveriam ser relativas aos primeiros conhecimentos que um iniciante na linguagem Java procura obter. Os participantes não possuíam qualquer informação sobre o que constava na ontologia de domínio. Esta fase, além de fazer com que os participantes pegassem intimidade com o sistema, fez com que um conjunto de teste fosse gerado por pessoas que não possuíam conhecimento do que existia na base de dados do sistema, além do fato já citado de não saberem o que existia na ontologia. Ao todo foram informadas cinqüenta e cinco perguntas sobre os tópicos mais variados da linguagem Java. O usuário deveria informar um título para sua pergunta, e a pergunta propriamente dita. A seguir alguns exemplos de perguntas informadas pelos participantes:

**Herança**

Como eu implemento herança em java ?

**Try catch**

Como usar try catch?

**Visibilidade**

Um método protected de uma classe pode ser visualizado por um método de outra classe, dentro do mesmo pacote?

**Arquivo**

Como faço para abrir arquivos?

**Integer para int**

Como consigo pegar o tipo primitivo int de um objeto Integer?

## 6.6 - Quarta Etapa – Participantes Avaliam Respostas às suas Perguntas

A cada uma das cinco perguntas efetuadas pelos participantes, o Agente Java forneceu dez respostas classificadas em ordem de relevância. Na figura 20 podemos ver a pergunta efetuada pelo participante, e duas das respostas dadas pelo agente.

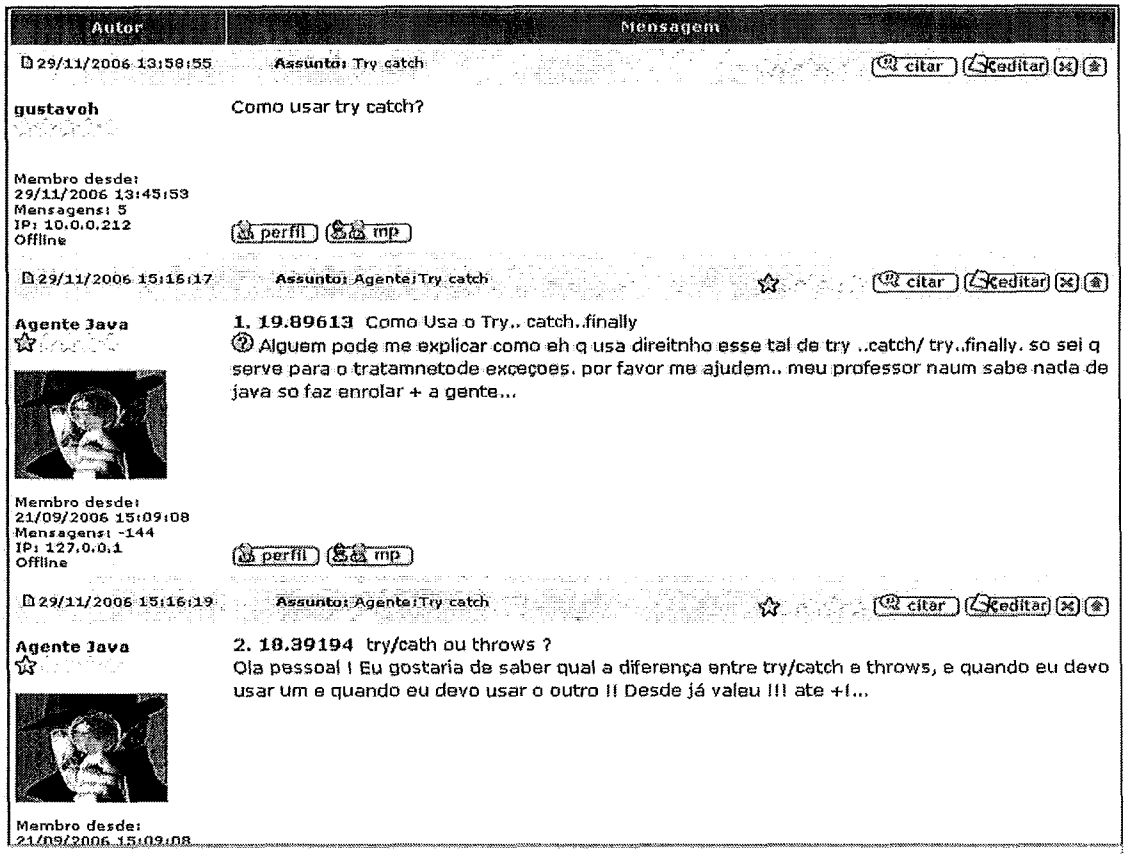



**Figura 20 – Respostas do agente**

Podemos observar, que dentre os resultados retornados pelo agente não constam respostas propriamente ditas, mas perguntas semelhantes às enviadas pelos usuários. Clicando nas perguntas retornadas pelo agente é possível verificar se existe alguma informação que ajude ao usuário solucionar sua dúvida. Este é o objetivo do sistema: localizar outros usuários com dúvidas, ou questionamentos semelhantes a questionamentos anteriormente enviados. Foi solicitado aos participantes que clicassem nos *links* que direcionavam à outras conversas, para que avaliassem se estas conversas o ajudariam a solucionar suas dúvidas.

A cada um dos participantes foi pedido que avaliassem as respostas fornecidas a cada uma de suas perguntas. Cada participante avaliou então as dez respostas de cada uma de suas cinco perguntas, totalizando cinqüenta avaliações por participante.

A interface utilizada para a avaliação das respostas do agente é mostrada na figura 21.

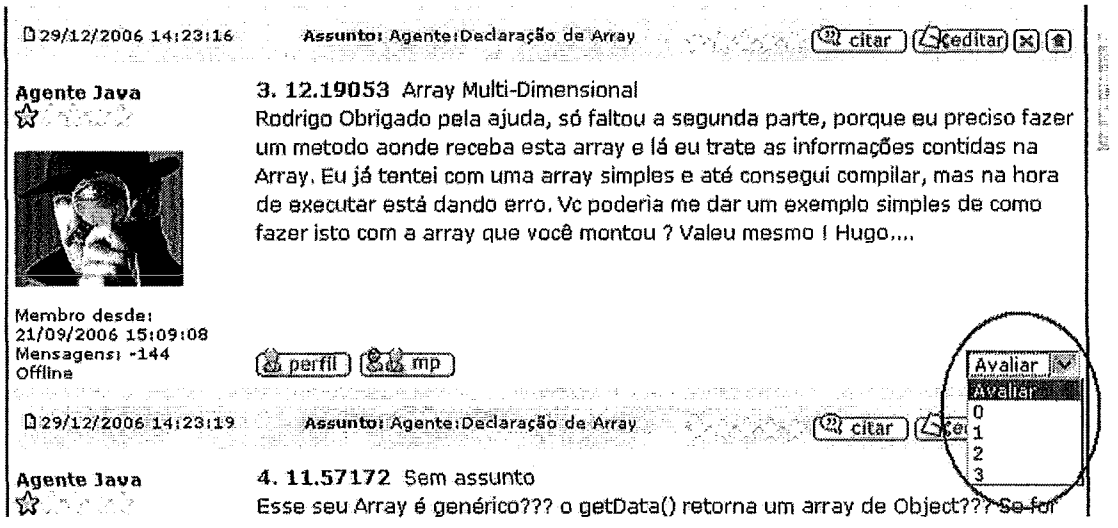



**Figura 21 – Interface para avaliação das respostas do agente**

A avaliação de cada resposta deveria ser dada de acordo com os critérios presentes na tabela 8.

| *Nota* | *Descrição* |
|---|---|
| 3 | A resposta responde exatamente à minha pergunta |
| 2 | A resposta tem algo que me ajuda a solucionar minha dúvida (um link por ex.) |
| 1 | A resposta tem a ver com o assunto, mas não é exatamente isso que eu quero |
| 0 | A resposta não tem nada a ver com o que eu perguntei |

**Tabela 8 - Notas para avaliação**

Neste caso, cada resposta do agente obteve uma única avaliação, a avaliação do autor da pergunta.

## 6.7 - Quinta Etapa – Participantes Avaliam Respostas às Perguntas pré Selecionadas

Nesta etapa, foi aberto um novo fórum aos participantes. Este fórum se chamou "Minhas perguntas sobre Java básico". Na figura 22 podemos observar os dois fóruns distintos, o primeiro onde os usuários fizeram suas perguntas, e o segundo, com perguntas selecionadas de acordo com conceitos presentes na ontologia.

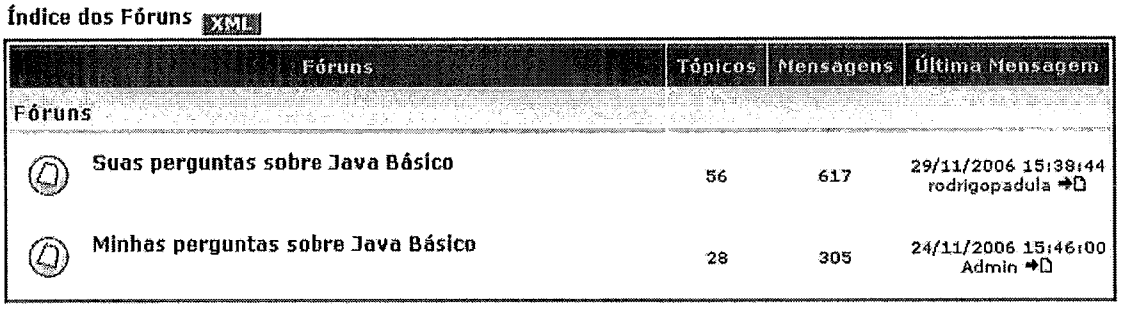



**Figura 22 – Fóruns do experimento**

No fórum "Minhas perguntas sobre Java Básico" foram cadastradas vinte e oito perguntas, com alguns aspectos diferenciados em relação às perguntas originalmente submetidas pelos participantes. Todos os conceitos presentes nas perguntas do segundo fórum constavam na ontologia. Sendo assim, todos os recursos de expansão e atribuição de pesos seriam utilizados. Entretanto, para cada pergunta o agente forneceu dois conjuntos de dez respostas: um conjunto utilizando os recursos da ontologia e um segundo conjunto sem utilização da ontologia. Para facilitar a comunicação e freqüente citação dos dois conjuntos de perguntas, chamaremos o conjunto de perguntas dos participantes de QA e o conjunto de perguntas pré selecionadas de QB. Como o conjunto de perguntas só teve um tipo de resposta (com utilização da ontologia) chamaremos o conjunto de resposta de A de RA1. Aos cojuntos de respostas de QB, daremos os nomes de RB1 (com ontologia) e RB2 (sem ontologia). Para facilitar a compreensão, podemos visualizar na tabela 9 uma síntese da nomenclatura utilizada.

| *Sigla* | *Descrição* |
|---|---|
| QA | Conjunto de perguntas enviadas pelos participantes |
| QB | Conjunto de perguntas selecionadas para o experimento (todas com conceitos presentes na ontologia) |

| | |
|---|---|
| RA1 | Conjunto de respostas dadas às perguntas de QA (com utilização de ontologia) |
| RB1 | Conjunto de respostas dadas às perguntas de QB (com utilização da ontologia) |
| RB2 | Conjunto de respostas dadas às perguntas de QB (sem utilização de ontologia) |

**Tabela 9 – Nomenclatura utilizada para os conjuntos de respostas**

Foi solicitado aos participantes, que avaliassem as respostas dadas pelo agente às perguntas de QB da mesma maneira que eles avaliaram as respostas de suas próprias perguntas (tabela 8). Como o tempo era restrito, e para que um maior número de perguntas pudessem ser avaliadas, os participantes foram divididos em quatro grupos A, B, C e D. Foram realizadas quatro rodadas de avaliação. O grupo A avaliava as sete primeiras perguntas enquanto o grupo B avaliava as perguntas de número 8 a 14 e assim sucessivamente. Quando os grupos terminavam um conjunto de avaliações, seguiam para um novo conjunto de perguntas não avaliadas por eles. Repare que nesta etapa, diferentemente da etapa de avaliação de QA, cada uma das respostas às perguntas de QB recebeu quatro avaliações. Porém, duas dessas avaliações eram às respostas que foram obtidas com a utilização de ontologia (RB1) e outras duas sem a utilização da ontologia (RB2). Os participantes não sabiam quais resultados foram recuperados com a utilização da ontologia e quais não foram, para que sua avaliação não fosse influenciada.

Assim como no conjunto de perguntas QA, as perguntas em QB possuíam um título e um corpo contendo a pergunta. A seguir exemplos de algumas das perguntas contidas em QB:

**Declaração de Array**

Como declaro um array em java?

**HashMap**

Como faço para utilizar um HashMap?

**Formatar casas decimais**

Que comando devo utilizar para formatar as casas decimais de um double?

**Conversão de string para Timestamp**

Como faço para converter uma string em timestamp?

**Métodos private e protected**

Qual a diferença entre um método private e um protected?

## 6.8 - Resultados

Na análise das avaliações para cada um dos conjuntos de teste, vários resultados interessantes foram obtidos.

Nas tabela 10 podemos observar alguns dados obtidos após a comparação dos resultados encontrados na análise das avaliações de RA1, RB1 e RB2

| *Dados* | *Com ontologia* | | *S/ onto.* |
|---|---|---|---|
| | *RA1* | *RB1* | *RB2* |
| Total de participantes | 11 | 11 | 11 |
| Total de perguntas | 55 | 28 | 28 |
| Número de perguntas feitas por participante | 5 | 0 | 0 |
| Número de avaliações por pergunta | 1 | 2 | 2 |
| Média geral da nota obtida pelas avaliações de cada pergunta (intervalo entre 0 e 3) | 0.69741697 | 1.07843148 | 0,63432836 |
| Média considerando somente a avaliação da nota da melhor resposta de uma pergunta | 1.9455 | 2.9630 | 2,3704 |
| Porcentagem de perguntas com avaliação máxima de resposta entre 2 e 3 | 69,81% | 100% | 71,4286% |
| Porcentagem de perguntas com avaliação máxima de resposta entre 0 e 1 | 30,19% | 0% | 28,5714% |

**Tabela 10 – Comparação de resultados de Q1 e Q2**

Após a análise dos resultados mostrados na tabela 10 comparando os resultados de RA1 e RB1 podemos observar que os resultados obtidos para o conjunto RB1 são sempre melhores quando comparados ao conjunto RA1. Isto porque parte das perguntas submetidas pelos participantes não possuía qualquer compromisso com

a ontologia. Como as consultas são submetidas em forma de pergunta, os participantes incluem vários outros termos que podem confundir o sistema de busca caso nenhum conceito presente na consulta seja mapeado na ontologia (ou alguns conceitos sejam mapeados e outros não). Um exemplo deste comportamento foram as respostas dadas à seguinte pergunta feita pelo usuário: "Como criar uma classe com um método *main*?". Neste caso a usuária gostaria de mais informações sobre o método "*main*", mais especificamente sobre o conceito "*main*". Entretanto, este conceito não estava presente na ontologia. Os conceitos reconhecidos para esta pergunta foram: classe e método. Sendo assim, a participante obteve vários resultados que falavam sobre a utilização de classes, métodos entre outros conceitos relacionados a estes, mas nenhuma das respostas falava especificamente sobre o método "*main*".

No conjunto RB1 toda pergunta possuía pelo menos um conceito presente na ontologia. No quesito média geral, o conjunto RA1 ficou bem abaixo da média de RB1. Entretanto, quando consideramos somente as melhores notas atribuídas a cada uma das respostas fornecidas, o resultado é surpreendente. Neste quesito, foram considerados os seguintes parâmetros: para cada uma das dez respostas fornecidas, é selecionada a maior nota que uma resposta obteve. Sendo assim, se a maior nota atribuída à resposta de uma determinada pergunta for três, este será o parâmetro utilizado para o cálculo da média. Caso todas as respostas de determinada pergunta recebam o valor zero, zero será a nota máxima das respostas de determinada pergunta. Em RA1 este valor se aproxima de dois, o que é superior a 50% de satisfação no que diz respeito à qualidade de pelo menos uma das respostas de cada pergunta. No resultado de RB1 esse valor aumenta mais ainda, indicando que em média para cada uma das perguntas, pelo menos uma das respostas está absolutamente satisfatória.

Nos dois últimos quesitos confirmamos o resultado mencionado no parágrafo acima. Enquanto em RA1 69.81% das respostas obtiveram avaliação máxima entre 3 e 2, 100% das respostas em RB1 obtiveram pelo menos uma avaliação máxima entre 3 e 2. Ou seja, para o conjunto de perguntas cujos conceitos estão presentes na ontologia, os participantes consideraram que pelo menos uma resposta satisfazia à pergunta proposta.

O terceiro conjunto de respostas, RB2, possuía avaliações às mesmas perguntas de QB (conseqüentemente de RB1) porém não foi obtido com a utilização

da ontologia. Ao compararmos os resultados de RB1 e RB2 comparamos diretamente a avaliação de respostas que foram utilizadas com e sem a utilização da ontologia.

Assim como em RA1, todos os resultados de RB2 são inferiores a RB1. Em parte pelo mesmo motivo, de que, sem a utilização da ontologia, o sistema "se perde" na identificação do que deve ser importante para determinada pergunta. Por outro lado, se compararmos os resultados de RB2 com RA1 podemos observar que os resultados de RB2 são ligeiramente superiores à RA1, mesmo com RA1 tendo sido recuperado com a utilização de ontologia e RB2 não. Esses valores podem ser justificados pelo fato de que, mesmo utilizando ontologia, em RA1 nem toda pergunta teve seu conceito mapeado, e que, o conjunto de perguntas em RB2 era mais controlado do que RA1. Por ser um conjunto controlado e cuidadosamente selecionado, de perguntas mais freqüentes sobre iniciantes na linguagem Java. A superioridade de RB2 em relação à RA1 se deu pelo fato da escolha de perguntas e da recuperação do MySQL, e não em função da ontologia propriamente dita.

### 6.8.1 - Análise do percentual de pontuação atribuída e cálculo de precisão

O gráfico da figura 23 mostra o percentual de respostas avaliadas com cada uma das notas (entre 0 e 3) em cada um dos conjuntos. O grande percentual de respostas avaliadas com zero em RA1 e RB2 mostra a quantidade de resultados irrelevantes que podem ser trazidos sem a utilização da ontologia. Isto devido ao peso errado que determinado termo pode ter quando a consulta é submetida. RB1 apresenta um percentual bem menor de notas 0.

A proximidade de percentual nos valores 1 e 2 para os três sugere que independente da utilização da ontologia, um numero semelhante de respostas nos resultados trazidos tem de certa forma algo a ver com o que foi pedido, ainda assim, o conjunto RB1 (em que a ontologia foi utilizada plenamente) apresenta desempenho ligeiramente superior para os dois níveis de pontuação.

Já para o nível de pontuação 3, podemos observar que a avaliação de RB1 é bem superior a dos outros conjuntos, isto porque com a utilização da ontologia, foram atribuídos pesos aos termos, que tendem a demonstrar o que é realmente importante na pergunta.

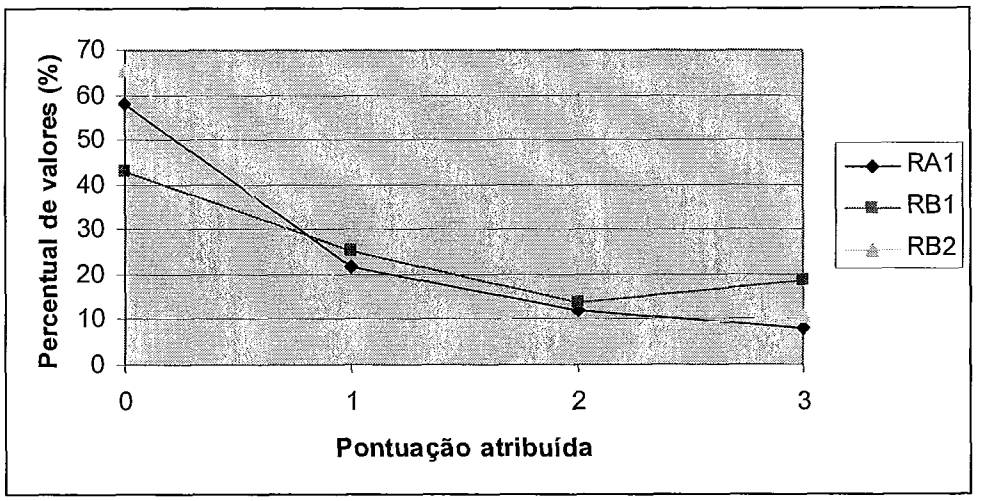



**Figura 23 – Gráfico de pontuação atribuída e suas porcentagens**

O cálculo da precisão considera a proporção de resultados relevantes nos resultados recuperados. Como visto no capítulo 4, o conceito clássico de precisão considera todos os resultados trazidos. Neste protótipo consideramos apenas os 10 primeiros resultados retornados. Como em Hawking *et al* (1999), utilizaremos o conceito de precisão dado um número de resultados retornados. Para a precisão em 10 resultados retornados, é utilizada a expressão: P@10.

Hawking *et al* (1999), também fala sobre a utilização do cálculo da revocação. Segundo ele, no contexto da Web é freqüentemente dito que as pessoas não estão interessadas nos valores de revocação. Sendo assim, caso isto seja verdade, a avaliação deve focar na dimensão de precisão. Na verdade esta perspectiva é bastante oportuna uma vez que o cálculo da revocação exige que sejam avaliados todos os documentos da coleção, para classificá-los como relevantes ou irrelevantes. Tal tarefa seria impossível na Web, uma vez que é extensa a coleção de documentos, e esses documentos mudam diariamente. Mesmo no caso do nosso protótipo, com algo em torno de 19 mil documentos seria bastante custoso o processo de avaliação de todos esses documentos.

Para calcular os valores de precisão P@10 precisamos primeiramente definir o que deve ser classificado como relevante e não relevante.

No nosso experimento foi atribuída uma escala de avaliação que varia entre 0 e 3. Do menos relevante ao mais relevante. Como seria esta escala se mapeássemos para o conceito binário de relevante e não relevante? 3 é relevante e 0 não relevante?

Como o participante teria avaliado os documentos que receberam avaliação 1 e 2? A avaliação 2 não é boa o suficiente para que o documento seja considerado relevante? O conceito é muito abstrato, sendo assim, para determinado participante, se fosse pedido para classificar um documento apenas como relevante ou irrelevante talvez sua atribuição de pontos fosse diferente. Sendo assim, optamos por calcular a precisão em todas as combinações do que poderia ser julgado como relevante e irrelevante de acordo com a atribuição de pontos efetuada, como pode ser observado no gráfico da figura 24. Este gráfico mostra qual seria o P@10 caso relevantes fossem os documentos com pontuação 3 e os demais relevantes, ou, relevantes seriam os documentos com pontuação 3 e 2 e finalmente a hipótese de que seriam relevantes quaisquer documentos que não obtivessem pontuação zero (3 ou 2 ou 1).

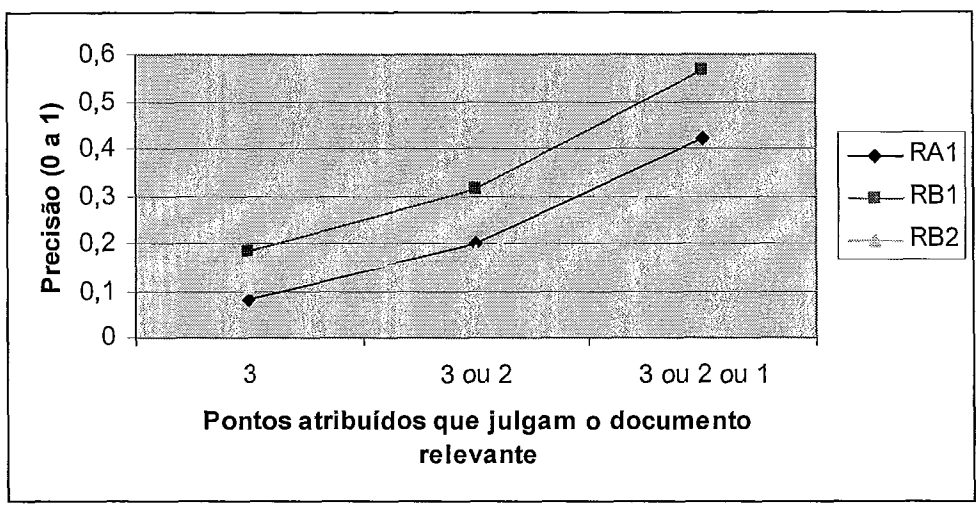



**Figura 24 – Gráfico de P@10 dadas hipóteses diferentes de relevância**

## 6.9 - Conclusão

No penúltimo capítulo de nossa dissertação, avaliamos os resultados produzidos por um experimento ao qual o protótipo aqui desenvolvido foi submetido.

Foram analisadas configurações de busca de resultados com e sem utilização de ontologia, e pôde ser observado que as configurações que utilizaram a ontologia apresentaram resultado superior em relação às que não utilizaram.

Pudemos observar o quão importante é a utilização de uma ontologia, que será utilizada para contextualizar a informação submetida ao sistema. Isto permitirá que o usuário tenha mais liberdade na especificação da consulta, de maneira que o sistema

identifique prontamente os termos que devem ser mais valorizados em detrimento de outros.

# Capítulo 7 – Conclusão

Comunidades de prática são objeto de estudo de inúmeros trabalhos na área de gestão do conhecimento. Com a evolução da Internet, essas comunidades ganharam sua faceta virtual, e passaram a produzir uma enorme quantidade de conhecimento devido às informações trocadas diariamente via e-mails e fóruns de mensagens.

Graças aos meios de armazenamento digital, o conhecimento dessas comunidades é mantido em bases de dados, guardando a memória da comunidade. Entretanto, dadas as crescentes taxas de geração de informação, buscar determinada informação em um repositório de conhecimento de uma comunidade virtual pode não ser tarefa fácil ao membro iniciante. O mesmo assunto pode ser abordado de diversas maneiras, e com a utilização de diversos termos, que ao interpretador humano teriam o mesmo significado. Além disso, a busca clássica por palavras-chave pode ser um limitante às inúmeras possibilidades de obtenção de informação que se poderia ter caso o assunto a ser procurado pudesse ser contextualizado em uma visão mais ampla de acordo com conceitos abordados na comunidade.

Esta dissertação propôs a criação de uma arquitetura multi-agente, que atuando em uma comunidade de prática virtual, teria como objetivo armazenar, filtrar, categorizar e recuperar conhecimento sob o enfoque da Web semântica aliada às técnicas já consagradas de busca e recuperação de informação.

Nesta arquitetura, um agente atua como um membro da comunidade, procurando por oportunidades de colaboração ao responder perguntas enviadas por outros membros da comunidade. Como memória do agente, são utilizadas as próprias mensagens trocadas na comunidade de prática, ou de outras comunidades sobre o mesmo domínio, considerando também a indicação de material externo sobre o assunto tratado, como livros e endereços com conteúdo escrito que possa ser catalogado pelo agente.

Os mecanismos de associação de informações da arquitetura multi-agente são proporcionados por uma ontologia de domínio, que tem como objetivo contextualizar a informação trocada, além da possibilidade de aprimoramento da pergunta enviada pelo membro da comunidade através da expansão dos termos associados a conceitos da ontologia.

Iniciamos nossa dissertação explicando as principais teorias que envolvem comunidades de prática. Falamos sobre as opiniões divergentes dos autores sobre o conceito de comunidades de prática virtuais até chegarmos nas comunidades de prática de Java, que foram o assunto aqui abordado.

Em seguida fizemos uma breve descrição dos princípios da Web semântica. Falamos sobre a importância da representação do conhecimento em um contexto semântico, e fornecemos uma detalhada visão sobre os propósitos, a construção e utilização de ontologias, bem como suas linguagens de construção e sua aplicação no domínio de comunidades de prática. Ainda sob o enfoque da Web semântica analisamos os agentes de software, as principais definições dos autores da área, as vantagens de utilização de agentes colaborativos que interagem com seres humanos e a aplicação de agentes de software na área de busca e recuperação de informação.

Analisamos também as principais técnicas de recuperação de informação clássica. Demonstramos brevemente os principais modelos utilizados, suas vantagens e desvantagens além de algumas técnicas complementares a estes modelos. Falamos também da expansão de consultas, uma das principais técnicas utilizadas no protótipo implementado por esta dissertação, para que a pergunta original enviada por um membro da comunidade pudesse ser enriquecida com conceitos encontrados na ontologia. Abordamos as principais técnicas de expansão de consultas, analisando as vantagens e desvantagens na sua utilização, aproveitando também para definir alguns conceitos que foram utilizados a partir de um trabalho sobre um léxico gerativo.

A viabilidade da arquitetura proposta é então verificada com a implementação de um protótipo. Demonstramos os principais aspectos dos módulos utilizados em nossa arquitetura multi-agente. A atividade de cada agente é explicitada, bem como a interação entre eles, suas informações de entrada e saída.

Após descrever a arquitetura do COOPRACTICE, mostramos passo a passo a construção da ontologia a ser utilizada, desde o processo de seleção de termos até o estabelecimento de relações entre eles, culminando na inclusão de cerca de cem conceitos via inclusão manual e inúmeros conceitos via inclusão automática sobre a linguagem Java. Na demonstração do processo de construção de ontologia, utilizamos os passos sugeridos por Noy e McGuinness (2001) utilizando como ferramenta o aplicativo Protege (PROTEGE, 2007). Definida a construção da ontologia, analisamos qual foi seu papel na expansão das perguntas enviadas pelos membros da comunidade.

Estabelecidos os critérios de seleção de termos para expansão, fizemos uma adaptação ao algoritmo de Gonzales e, Strube de Lima (2001) para calcular os pesos que seriam atribuídos aos termos expandidos, uma vez que a expansão de termos sem atribuição de pesos poderia prejudicar a qualidade dos resultados ao invés de melhorá-la. Na conclusão da arquitetura do protótipo, falamos sobre as ferramentas de armazenamento, e os critérios utilizados para a escolha da ferramenta adotada.

Na parte final desta dissertação, avaliamos o protótipo implementado através da realização de um experimento. A avaliação contou com a colaboração de onze alunos da turma de busca e recuperação de informação do mestrado da linha de Banco de Dados do Programa de Engenharia de Sistemas da COPPE – UFRJ.

Na avaliação foi utilizada uma interface de perguntas e respostas, onde os participantes deveriam avaliar perguntas submetidas por eles, e uma série de perguntas pré-selecionadas para o experimento. O escopo das perguntas deveria ser restrito às dúvidas de iniciantes na linguagem Java, ou, Java básico. Através de um questionário inicial, pudemos concluir que todos os participantes possuíam conhecimentos de Java entre um nível intermediário e avançado, sendo assim considerados aptos a darem avaliações confiáveis no experimento.

Foram avaliadas respostas com e sem a utilização de ontologia. Nos resultados demonstrados pudemos observar a superioridade dos resultados que foram trazidos com a utilização da ontologia. Como as perguntas eram livres, nos casos em que a ontologia não era utilizada, era freqüente que assuntos relacionados a outros termos não tão importantes fossem trazidos como resposta as perguntas submetidas. Quando a resposta era dada com o auxílio da ontologia, era possível ao sistema identificar o que deveria ser considerado como importante, uma vez que os conceitos mencionados e reconhecidos na ontologia adquiriam maior peso na consulta. Além do reconhecimento dos conceitos principais, era importante também que os termos obtidos por expansão não "atrapalhassem" o propósito original da consulta, o que foi contornado com a atribuição de pesos. Qualquer termo expandido teria peso inferior aos termos originais da consulta.

Concluímos então, que a utilização de ontologias para expansão de uma consulta submetida a um sistema e busca de recuperação de informação pode trazer resultados significativos em termos de qualidade dos resultados, e na obtenção de

resultados relacionados a consulta original de uma maneira semântica e não só por sua sintaxe.

Pudemos observar também que ao utilizar um sistema que se valha de uma ontologia de domínio para dar pesos aos termos inseridos pelo usuário, consultas mais livres podem ser submetidas, uma vez que é facilmente identificável o que é e o que não é importante na consulta, tirando o usuário assim do modelo rígido de consultas booleanas. Tudo isso pode ser obtido a um baixo custo de processamento, uma vez que não são utilizadas análises estruturais da consulta ou análises mais complexas como processamento de linguagem natural.

## 7.1 - Trabalhos futuros

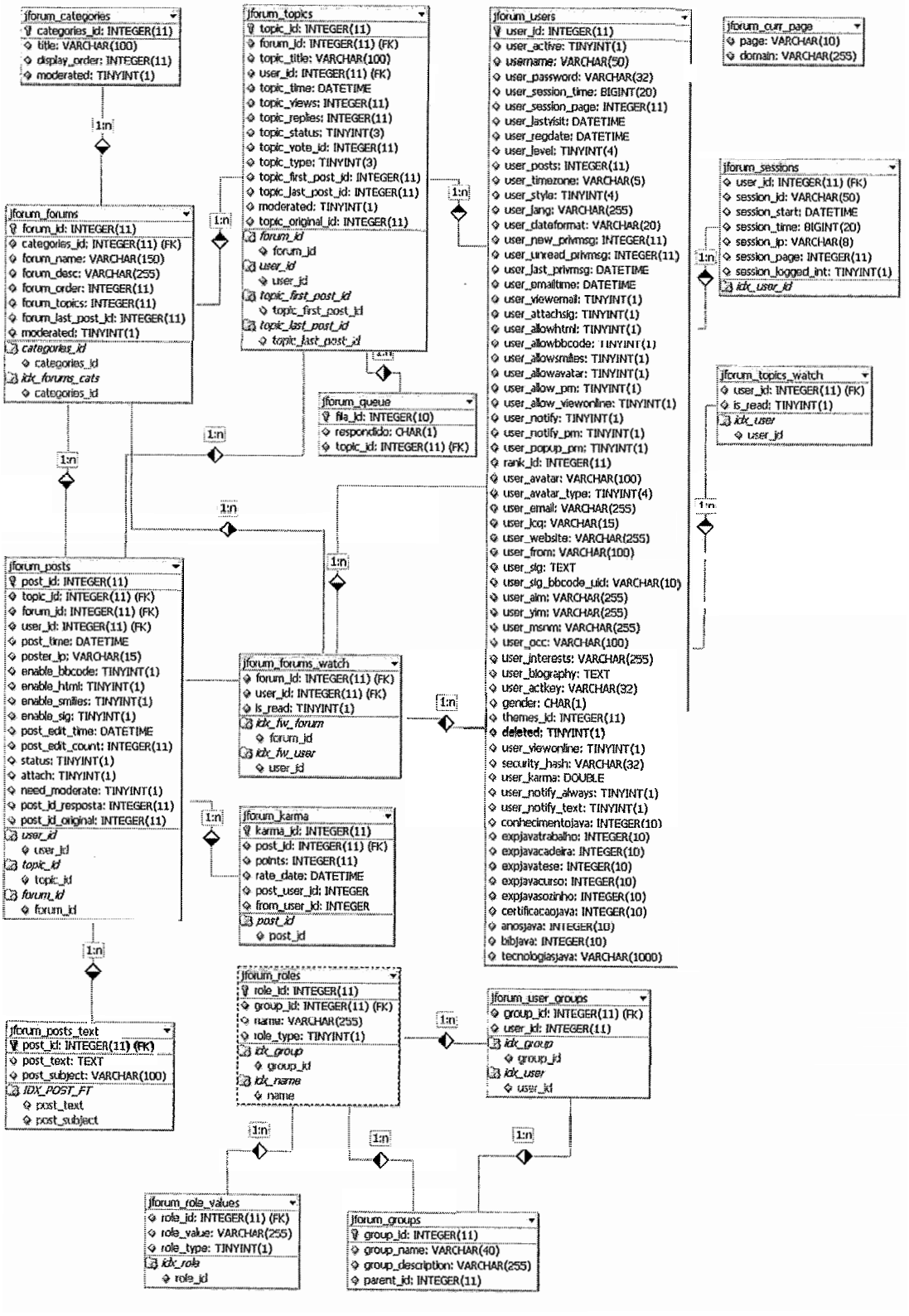
Podemos vislumbrar uma série de possibilidades na utilização de uma arquitetura multi-agente apoiada por uma ontologia para diferentes domínios de aplicação. Neste trabalho, abordamos o tema comunidades de prática por possuírem um repositório de conhecimento controlado, facilmente manipulável e de fácil obtenção para testes. Entretanto, os princípios aqui utilizados podem ser aplicados em várias outras aplicações, como portais de busca, bibliotecas digitais entre outros.

Nem todos os agentes da nossa arquitetura foram implementados no protótipo, mas somente um subconjunto essencial para que pudéssemos avaliar a viabilidade do sistema.

A implementação dos agentes de classificação ontológica (ACO) e de recuperação de referências (ARR) trariam enormes vantagens no que diz respeito à recomendação, uma vez que as mensagens poderiam ser classificadas de acordo com os conceitos da ontologia, e referências externas de sítios de Internet e ou livros poderão ser sugeridos aos membros da comunidade.

A proposta desta dissertação é apenas uma instância do que pode ser realizado com o potencial das idéias propostas pela terceira geração da Internet. Os princípios da Web semântica poderão ser utilizados nas mais variadas áreas de aplicação. Munidos de contextualização de informação e uma arquitetura de agentes responsáveis por realizar o trabalho árduo, os membros de comunidades de prática e usuários da Internet ficarão mais livres para trocar e procurar por conhecimento de alto nível que somente humanos sabem trocar, e que talvez, nenhum sistema seja capaz de interpretá-lo.

# Apêndice A – O modelo de dados



# Apêndice B – JForum, ferramenta de gerenciamento de fóruns em Java

Para a realização do nosso experimento, utilizamos o JForum (JFORUM, 2007). O JForum é um sistema para construção e gerenciamento de fóruns criado por brasileiros. Ele foi desenvolvido na linguagem Java, e suporta todos os bancos de dados comerciais e gratuitos, através da conexão com a API Java JDBC (JDBC, 2007).

Sendo o JForum uma proposta de código aberto, foi possível fazer várias alterações no sistema de maneira que este se adaptasse às nossas necessidades. Durante o desenvolvimento do protótipo do COOPRACTICE encontramos algumas falhas no JForum e as reportamos à comunidade de desenvolvimento. Os responsáveis foram bastante solícitos e atenderam às nossas solicitações liberando versões com as falhas corrigidas. O próprio JForum, possui o seu fórum de mensagens (feito no JForum) onde a comunidade de desenvolvedores troca informações sobre o andamento do projeto, sugestões, e evoluções do sistema.

Dotado de uma interface altamente amigável, o JForum possui uma série de recursos de configuração, e adaptação às necessidades de cada usuário. Com esta ferramenta é possível criar rapidamente um fórum de mensagens, moderado ou não e customizado de maneira bastante flexível.

A utilização do JForum fez com que pudéssemos nos desligar dos aspectos de interface, para nos ater somente às questões inerentes à proposta da dissertação.

Uma ferramenta bastante robusta e com muitos recursos, onde a qualidade de seu código-fonte mostra a seriedade do projeto. Um projeto de código aberto, entretanto com qualidade profissional que traz orgulho à comunidade de desenvolvedores brasileiros.

# Apêndice C – JADE, ferramenta para construção de arquiteturas multi-agente

Ao propor a criação de uma arquitetura multi-agente precisaríamos de uma plataforma onde nossos agentes pudessem ser criados.

Ao se pesquisar por plataformas multi-agente na Internet inúmeras opções são sugeridas, entretanto, as de maior expressividade são: Aglets (AGLETS, 2007) e JADE (JADE, 2007).

As duas plataformas foram analisadas com o objetivo de verificar qual se adequaria melhor aos nossos propósitos.

A plataforma *Aglets* é uma plataforma de agentes móveis desenvolvida em Java 1.1 pela IBM/Japão (AGLETS, 2007). Posteriormente, o código fonte da plataforma foi aberto (*open source*) em uma tentativa de obter interessados em trabalhar nesse processo de adaptação do código para versões posteriores de Java.

Embora com uma boa proposta, a plataforma Aglets carece de boa documentação, o que torna o trabalho do desenvolvedor bastante árduo. Além disso, as funcionalidades não são simples de utilizar, exigindo um grande esforço de compreensão no início do aprendizado das bibliotecas.

Ao analisarmos a plataforma JADE, constatamos que boa parte dos problemas existentes nos Aglets não existiam. Sendo assim, optamos por sua utilização. JADE conta com uma boa documentação, além de praticidade na utilização das funcionalidades, proporcionando um aprendizado rápido, sendo possível construir uma arquitetura simples de agentes com conhecimentos básicos da biblioteca.

Jade (JADE, 2007) (Java Agent DEvelopment framework) é um ambiente para desenvolvimento de aplicações baseada em agentes conforme as especificações da FIPA (FIPA, 2007) (Foundation for Intelligent Physical Agents) para interoperabilidade entre sistemas multiagentes totalmente implementado em Java. Foi desenvolvido e suportado pela Universidade de Parma na Itália, e é uma aplicação de código aberto. Segundo Jade (2007), o principal objetivo do Jade é simplificar e facilitar o desenvolvimento de sistemas multiagentes, garantindo um padrão de interoperabilidade entre esses sistemas, através de um abrangente conjunto de agentes de serviços de sistema, os quais, tanto facilitam como possibilitam a comunicação entre agentes, de acordo com as especificações da FIPA: serviço de nomes (*naming*

*service*), páginas amarelas (*yellow-page service*), transporte de mensagens, serviços de codificação e decodificação de mensagens e uma biblioteca de protocolos de interação (padrão FIPA) pronta para ser usada (AYRES, 2003).

A plataforma JADE conta com as seguintes características: plataforma distribuída de agentes, interface gráfica para criação de agentes, ferramentas de *debug,* suporte à execução de múltiplas, paralelas e concorrentes atividades de agentes, transporte de mensagens, biblioteca de protocolos FIPA, serviço de nomes, e suporte à integração com outras aplicações Java.

Esta última característica nos permitiu construir nossa aplicação sobre a plataforma.

Ao instanciarmos um agente JADE definimos um comportamento, através do pacote *jade.core.behaviors*. Os principais comportamentos apresentados por um agente são: *CyclicBehavior* (agentes que ficam executando tarefas continuamente), *TickerBehavior* (executado periodicamente após um intervalo de tempo), *WakerBehavior* (executado uma vez após um intervalo de tempo) e *OneShotBehavior* (executado uma vez assim que é chamado).

Além das classes de comportamento, existem também as classes que são utilizadas exclusivamente para a troca de mensagens entre inúmeras outras funcionalidades.

Não é objetivo deste trabalho explicar todas as características sobre a plataforma JADE. Um trabalho bastante completo e na língua portuguesa pode ser encontrado Ayres (2003), uma de nossas fontes de aprendizado sobre os principais conceitos desta eficiente arquitetura multi-agente.

# Apêndice D – Ferramenta de Armazenamento e Recuperação de Informações

Após estabelecer mecanismos de expansão de consultas, e determinar pesos para os termos expandidos, necessitaríamos de um repositório para nossos dados, cujas informações pudessem ser recuperadas através da submissão das consultas obtidas.

Ao analisarmos a popular ferramenta de código aberto MySQL (MYSQL, 2006), observou-se que além do modo clássico de busca de informações, o MySQL dispunha de um meio de indexação denominado *full-text search*. O banco de dados Postgres (POSTGRES, 2007) também dispõe de um modo *fulltext*, entretanto o MySQL foi escolhido por apresentar maior documentação disponível na Internet, e um esclarecimento melhor de suas fórmulas utilizadas. Além disso, pôde ser facilmente utilizado em conjunto com o JForum, ferramenta de gerenciamento de fóruns de Internet descrita no Apêndice B.

O modo *full-text search* do MySQL, nada mais é que a implementação do clássico modelo vetorial, apresentado no capítulo anterior. Analisando as fórmulas implementadas por este módulo do MySQL pudemos então perceber que a consulta, poderia ser submetida de tal maneira que refletisse os pesos calculados para cada termo.

Para a utilização deste modo, basta criar um índice *fulltext* nos campos onde se deseja fazer este tipo de busca. Um índice *fulltext* utiliza os seguintes parâmetros padrão:

- Termos com 4 letras ou menos não são considerados
- Palavras que aparecem na lista de *stopwords* do MySQL não serão consideradas
- Termos que estão presentes em mais de 50% dos documentos na coleção também não são considerados.

Todos estes parâmetros podem ser configurados. Para este protótipo configuramos o tamanho mínimo do termo para três letras, pois termos importantes inerentes a linguagens de programação poderiam ficar de fora (ex: termo "int", qualificador para variáveis do tipo inteiro). Também alteramos a lista de *stopwords*, já

que originalmente a lista era em inglês, para que constasse dos termos mais freqüentes na língua portuguesa.

Os cálculos do modo *fulltext* se baseiam na clássica fórmula de busca e recuperação:

```
w = tf * idf
```

cujo significado é: o peso w de um documento em uma coleção, é igual a freqüência do termo vezes o inverso do termo na coleção de documentos.

Partindo desta fórmula e com algumas alterações com o objetivo de normalização o MYSQL usa a seguinte fórmula (MYSQLFULLTEXT, 2006):

```
w = (log(dtf)+1)/sumdtf * U/(1+0.0115*U) * log((N-nf)/nf)
```

Onde:

| | |
|---|---|
| dtf | número de vezes que o termo aparece no documento |
| sumdtf | soma de (log(dtf)+1) para todos os termos no mesmo documento |
| U | número de termos únicos no documento |
| N | número total de documentos |
| nf | número de documentos que contém o termo |

A fórmula tem 3 partes: base, fator de normalização e multiplicador global

A parte base é o lado esquerdo da fórmula "(log(dtf)+1)/sumdtf".

O fator de normalização é a parte do meio da fórmula. A idéia de normalização é: se um documento é menor do que o tamanho médio, então seu peso sobe, se o seu tamanho está na média, então o peso permanece o mesmo, se ele é maior que a média então seu peso desce. (um documento menor significa um maior número de palavras únicas, um documento maior significa um menor número de palavras únicas). O MYSQL utiliza um fator de normalização único chamado de pivot. A palavra "único" significa que a medida do tamanho de um documento é baseada nos termos únicos de um documento. Foi escolhido o valor 0.0115 para o valor pivot. A teoria e justificativa da utilização de valores pode ser vista em SINGHAL *et al* (1996). O valor pivot padrão pode ser alterado modificando o valor da variável PIVOT_VAL do código fonte do MYSQL, no arquivo header myisam/ftdefs.h.

Multiplicando-se a parte base e o fator de normalização conseguimos o peso do termo no documento. O MYSQL armazena o peso do termo no índice.

O multiplicador global é a parte final da fórmula. Na fórmula do modelo clássico vetorial, a parte final seria o inverso da freqüência dos documentos que contém o termo ou simplesmente:

$$\log(N/nf)$$

No MYSQL a fórmula utilizada é:

$$\log((N-nf)/nf)$$

Esta variante é mais freqüentemente utilizada em fórmulas "probabilísticas". Estas fórmulas tentam fazer uma melhor aposta da probabilidade de que um termo seja relevante. Também é possível alterar a utilização desta fórmula pela primeira, bastando no arquivo myisam/ftdefs.h substituir a declaração #define GWS_IN_USE GWS_PROB por #define GWS_IN_USE GWS_IDF.

Por fim, temos que a relevância final do termo para o documento dada a consulta é:

$$r = w * qf$$

Onde:

w é o peso do termo no documento

qf é o número de vezes que o termo aparece na consulta

Esta fórmula será calculada para cada termo da consulta. A relevância final do documento nos resultados retornados, será o somatório das relevâncias de cada termo.

$$R = w_1 * qf_1 + w_2 * qf_2 + w_3 * qf_3 + \ldots w_t * qf_{t(\text{número de termos da consulta})}$$

Podemos concluir a partir desta fórmula final, que: quanto maior a freqüência de um termo na consulta, maior será o peso deste termo para o cálculo da relevância final, sendo assim, quanto mais vezes um termo aparecer na consulta, maior será seu peso para aquela consulta.

A grande vantagem na utilização desta fórmula é: a atribuição de pesos que demonstramos nesta dissertação, representarão a freqüência do termo na consulta. Sendo assim, replicaremos o termo quantas vezes for necessário até que seja atingido o peso calculado.

Na figura 17, ao termo T1 foi atribuído o peso final 8. Sendo assim, o termo T1 deverá ser repetido 8 vezes na consulta para que ele seja considerado com peso 8. Na matriz, cada termo é considerado como um único termo. Mesmo que o usuário repita seu termo na consulta, estes termos serão considerados como se fossem termos únicos, sendo assim, o termo apresentará um peso maior ainda (no caso do termo T1 ele apareceria 16 vezes) o que reflete a intenção do usuário, se o termo foi citado mais de uma vez, deve ter seu peso refletido como tal.

Ainda citando o exemplo da figura 17, a consulta original:

"T1 T2 T3 T4"

expandida, se transformaria em:

"T1 T1 T1 T1 T1 T1 T1 T1 T2 T2 T2 T2 T2 T2 T2 T2 T3 T3 T3 T3 T3 T3 T3 T3 T4 T4 T4 T4 T4 T4 T4 T4 T5 T5 T6 T6 T7 T7 T8 T9 T9 T10 T11"

# Apêndice E – Tabela de média de avaliações

| Título da Pergunta | # | RB1 | RB2 |
|---|---|---|---|
| Arredondamento de Double | 1 | 0,666667 | 0 |
| | 2 | 1,166667 | 0,333333 |
| | 3 | 1,833333 | 0,666667 |
| Como arredondar um Double? | 4 | 0,666667 | 0 |
| | 5 | 0,666667 | 0 |
| | 6 | 0,5 | 0,142857 |
| | 7 | 0,333333 | 0 |
| | 8 | 2,5 | 1,833333 |
| | 9 | 0,833333 | 0 |
| | 10 | 0 | 0 |
| Classes abstratas | 1 | 1,166667 | 1,285714 |
| | 2 | 1,166667 | 0 |
| | 3 | 2,333333 | 1,142857 |
| O que é uma classe abstrata? | 4 | 1,166667 | 0,142857 |
| | 5 | 3 | 2,5 |
| | 6 | 2,833333 | 0 |
| | 7 | 1,833333 | 0 |
| | 8 | 0,666667 | 0,666667 |
| | 9 | 1 | 0 |
| | 10 | 0 | 0,333333 |
| Collection | 1 | 0,857143 | 1 |
| | 2 | 0,857143 | 1,142857 |
| | 3 | 1,285714 | 0,285714 |
| O que é uma Collection? | 4 | 0,142857 | 1,333333 |
| | 5 | 0 | 0 |
| | 6 | 2,857143 | 1,142857 |
| | 7 | 2,571429 | 1,142857 |
| | 8 | 1,142857 | 0,666667 |
| | 9 | 0,285714 | 0 |
| | 10 | 1 | 1 |
| Como faço para declarar uma constante? | 1 | 0,333333 | 0,142857 |
| | 2 | 1 | 0,333333 |
| | 3 | 1,166667 | 1 |
| Como faço para declarar uma constante? | 4 | 1,333333 | 0 |
| | 5 | 0,333333 | 0 |
| | 6 | 0 | 0 |
| | 7 | 0,166667 | 0,333333 |
| | 8 | 0,166667 | 0,166667 |
| | 9 | 0 | 0,142857 |
| | 10 | 0,25 | 0,714286 |
| Comparação de Strings | 1 | 2,142857 | 0 |
| | 2 | 1,285714 | 0,666667 |
| | 3 | 0,714286 | 1 |
| Como faço para comparar duas strings? | 4 | 0,428571 | 1,5 |
| | 5 | 0,285714 | 0,857143 |
| | 6 | 0,142857 | 0 |
| | 7 | 0,285714 | 0,5 |
| | 8 | 1 | 0 |
| | 9 | 0,142857 | 1,166667 |
| | 10 | 2,714286 | 1,166667 |
| Conteúdo de arquivo | 1 | 1 | 0,833333 |
| | 2 | 0,833333 | 0,833333 |
| | 3 | 0,833333 | 0,666667 |
| | 4 | 1,166667 | 0 |
| Como ler o conteúdo de um arquivo texto? | 5 | 0,666667 | 0 |
| | 6 | 0,333333 | 0 |
| | 7 | 0,666667 | 1 |
| | 8 | 1 | 0,333333 |
| | 9 | 1,166667 | 0,333333 |
| | 10 | 0,666667 | 0 |

| | | | |
|---|---|---|---|
| Conversão de inteiro para String | 1 | 2 | 2,5 |
| | 2 | 1,5 | 0,5 |
| | 3 | 0,5 | 0,5 |
| | 4 | 0,5 | 1,166667 |
| Olá,<br>como faço para converter inteiro para string? | 5 | 2 | 2 |
| | 6 | 0 | 1 |
| | 7 | 0,5 | 1 |
| | 8 | 0,5 | 0 |
| | 9 | 0 | 0,5 |
| | 10 | 0 | 0 |
| Conversão de String para Data | 1 | 0,857143 | 0,2 |
| | 2 | 1,857143 | 0,833333 |
| | 3 | 0,285714 | 1 |
| | 4 | 0,285714 | 1,166667 |
| Como converter String para Date? | 5 | 0,571429 | 0 |
| | 6 | 1,428571 | 0,285714 |
| | 7 | 1 | 0,285714 |
| | 8 | 1,428571 | 0,666667 |
| | 9 | 1,142857 | 0 |
| | 10 | 1,285714 | 0,571429 |
| Conversão de String para Timestamp | 1 | 2,4 | 1,333333 |
| | 2 | 2,4 | 1,8 |
| | 3 | 0,2 | 1,166667 |
| | 4 | 3 | 1,833333 |
| Como faço para converter uma string em Timestamp? | 5 | 2,6 | 1,2 |
| | 6 | 1,2 | 0 |
| | 7 | 1 | 1,2 |
| | 8 | 0,6 | 2,4 |
| | 9 | 0,8 | 0,6 |
| | 10 | 2,4 | 1 |
| Data e Hora | 1 | 2,333333 | 1,166667 |
| | 2 | 1,8 | 1,833333 |
| | 3 | 0,666667 | 2,4 |
| Como pegar a data e hora atual do sistema? | 4 | 2 | 0 |
| | 5 | 1,166667 | 0,5 |
| | 6 | 0,5 | 0,5 |
| | 7 | 0,5 | 0,666667 |
| | 8 | 0,833333 | 1,2 |
| | 9 | 1,833333 | 0,142857 |
| | 10 | 1,833333 | 0 |
| Declaração de Array | 1 | 0,833333 | 0,142857 |
| | 2 | 0,8 | 0,6 |
| | 3 | 2,4 | 0 |
| Como declaro um array? | 4 | 1,2 | 0 |
| | 5 | 1,2 | 0,6 |
| | 6 | 2,8 | 0,2 |
| | 7 | 0,6 | 0,666667 |
| | 8 | 2 | 1,833333 |
| | 9 | 1,8 | 0 |
| | 10 | 1 | 0 |
| Enviando e recebendo emails com Java | 1 | 0,5 | 0 |
| | 2 | 0,5 | 0 |
| | 3 | 1 | 0,5 |
| Como enviar e receber emails em Java? | 4 | 0 | 0 |
| | 5 | 0,25 | 0,666667 |
| | 6 | 0 | 0,25 |
| | 7 | 0,5 | 0 |
| | 8 | 2 | 0 |
| | 9 | 0 | 0 |
| | 10 | 1 | 0 |
| Expressões regulares | 1 | 0,5 | 1 |
| | 2 | 2 | 0 |
| | 3 | 0 | 1,8 |
| Como faço para utilizar uma expressão regular em Java? | 4 | 0 | 0 |
| | 5 | 0 | 0 |
| | 6 | 0,5 | 0 |
| | 7 | 2 | 0 |
| | 8 | 0,5 | 0,5 |
| | 9 | 0 | 0 |
| | 10 | 0,5 | 0,2 |

| | | | |
|---|---|---|---|
| Formatar casas decimais | 1 | 0,2 | 1,666667 |
| | 2 | 1,8 | 0,8 |
| | 3 | 1,8 | 0,8 |
| Que comando devo utilizar para formatar as casas decimais de um double? | 4 | 2 | 0 |
| | 5 | 1,8 | 0 |
| | 6 | 2 | 1,666667 |
| | 7 | 1,8 | 1,333333 |
| | 8 | 1,4 | 1,666667 |
| | 9 | 2,6 | 0 |
| | 10 | 1,75 | 0 |
| HashMap | 1 | 0,4 | 0,2 |
| | 2 | 0,8 | 0 |
| | 3 | 2,6 | 0 |
| Como faço para utilizar um HashMap? | 4 | 0,6 | 0,6 |
| | 5 | 0,2 | 0 |
| | 6 | 0 | 2,666667 |
| | 7 | 2,6 | 0 |
| | 8 | 0,2 | 1,333333 |
| | 9 | 0,8 | 0 |
| | 10 | 2,6 | 0 |
| Inner Class | 1 | 2 | 2 |
| | 2 | 2,666667 | 0,714286 |
| | 3 | 1,333333 | 1 |
| O que é uma Inner Class? | 4 | 1 | 1 |
| | 5 | 1,666667 | 1,333333 |
| | 6 | 0,333333 | 0 |
| | 7 | 0,333333 | 1,666667 |
| | 8 | 0 | 0,333333 |
| | 9 | 0,333333 | 1,5 |
| | 10 | 1,666667 | 0 |
| InstanceOf | 1 | 2,166667 | 1,875 |
| | 2 | 2,166667 | 2 |
| | 3 | 1,5 | 1 |
| Para que serve o comando instanceof? | 4 | 2 | 2 |
| | 5 | 1,5 | 2,5 |
| | 6 | 1,5 | 1,5 |
| | 7 | 1,833333 | 1,5 |
| | 8 | 0,333333 | 0 |
| | 9 | 3 | 0,333333 |
| | 10 | 0,166667 | 0 |
| int e Integer | 1 | 0,833333 | 0,5 |
| | 2 | 0,5 | 0 |
| | 3 | 0,5 | 0 |
| Qual a diferença entre int e Integer? | 4 | 0,333333 | 0 |
| | 5 | 0 | 0 |
| | 6 | 0 | 0,666667 |
| | 7 | 0,333333 | 0,333333 |
| | 8 | 0,5 | 0,333333 |
| | 9 | 1 | 0 |
| | 10 | 0 | 0,666667 |
| Iterator | 1 | 2,25 | 2,5 |
| | 2 | 2,5 | 1,875 |
| | 3 | 1,5 | 0 |
| Para que serve um Iterator? | 4 | 0,5 | 0 |
| | 5 | 1,875 | 1,5 |
| | 6 | 0,875 | 0 |
| | 7 | 1,5 | 0,333333 |
| | 8 | 1,5 | 0 |
| | 9 | 1 | 1 |
| | 10 | 1 | 0,2 |
| Método Deprecated | 1 | 3 | 2,5 |
| | 2 | 3 | 3 |
| | 3 | 3 | 3 |
| O que é um método Deprecated? | 4 | 3 | 0 |
| | 5 | 1 | 0 |
| | 6 | 3 | 0 |
| | 7 | 0 | 0 |
| | 8 | 0 | 2,75 |
| | 9 | 3 | 1,75 |
| | 10 | 0 | 1,75 |

| | | | |
|---|---|---|---|
| Método final | 1 | 3 | 1 |
| | 2 | 0,75 | 0 |
| | 3 | 2,75 | 1,75 |
| O que é um método final? | 4 | 0 | 0 |
| | 5 | 1,5 | 0,2 |
| | 6 | 1,75 | 0 |
| | 7 | 0 | 0 |
| | 8 | 0 | 1,833333 |
| | 9 | 0 | 0 |
| | 10 | 0 | 0 |
| Métodos private e protected | 1 | 2,333333 | 1,875 |
| | 2 | 2,333333 | 1,666667 |
| | 3 | 1,666667 | 0,714286 |
| Qual a diferença entre um método private e um protected? | 4 | 0 | 0 |
| | 5 | 0,5 | 1,75 |
| | 6 | 0 | 1,166667 |
| | 7 | 0,666667 | 1 |
| | 8 | 2,6 | 0 |
| | 9 | 1,166667 | 0 |
| | 10 | 1 | 0 |
| O que é uma interface? | 1 | 0,166667 | 0,5 |
| | 2 | 1,666667 | 0 |
| | 3 | 0,5 | 1,666667 |
| O que é uma interface? | 4 | 1,5 | 0 |
| | 5 | 1,166667 | 0 |
| | 6 | 0,5 | 0 |
| | 7 | 2 | 0 |
| | 8 | 0,166667 | 0,714286 |
| | 9 | 1,666667 | 0 |
| | 10 | 0,833333 | 0 |
| Padrões | 1 | 2,333333 | 1,666667 |
| | 2 | 0,833333 | 0,142857 |
| | 3 | 0,166667 | 0 |
| Como utilizar o padrão Singleton? | 4 | 0,166667 | 0 |
| | 5 | 0 | 0 |
| | 6 | 0,833333 | 0,714286 |
| | 7 | 0 | 2,5 |
| | 8 | 0 | 0 |
| | 9 | 0 | 0 |
| | 10 | 0,5 | 0 |
| Subtração de datas | 1 | 1,428571 | 0,166667 |
| | 2 | 1,142857 | 0 |
| | 3 | 0 | 0 |
| Que comando devo utilizar para subrair duas datas? | 4 | 0,714286 | 0,666667 |
| | 5 | 0,142857 | 0 |
| | 6 | 2,142857 | 0 |
| | 7 | 2 | 0,142857 |
| | 8 | 0 | 0 |
| | 9 | 0,571429 | 0 |
| | 10 | 0,714286 | 0 |
| Threads | 1 | 1,4 | 3 |
| | 2 | 0,6 | 0,166667 |
| | 3 | 0,8 | 3 |
| Que biblioteca devo utilizar para implementar uma thread? | 4 | 0,4 | 0 |
| | 5 | 2 | 0 |
| | 6 | 1,8 | 0,714286 |
| | 7 | 0,6 | 1 |
| | 8 | 0,6 | 1 |
| | 9 | 0 | 0 |
| | 10 | 0 | 0 |
| Tratamento de exceção | 1 | 0,666667 | 2,5 |
| | 2 | 3 | 0 |
| | 3 | 0 | 0 |
| Quando utilizar try, catch e throws? | 4 | 1,666667 | 0,666667 |
| | 5 | 1 | 1,666667 |
| | 6 | 0,666667 | 1 |
| | 7 | 0,5 | 0 |
| | 8 | 2,333333 | 0 |
| | 9 | 1 | 0,5 |
| | 10 | 1,666667 | 0 |
| Total geral | | 1,078428 | 0,634184 |

# REFERÊNCIAS BIBLIOGRÁFICAS


ACKERMAN, M. S., MCDONALD, D. W. ,1996, "Answer Garden 2: Merging organizational memory with collaborative help". In: *CSCW '96*, pp. 97-105, Boston, MA

AGLETS, 1997, "Aglets", disponível em: http://www.trl.ibm.com/aglets/, acessado em 10/01/2007

ALLEN, J., 1995, *Natural Language Understanding*. Redwood City, CA.

AYRES, L. 2003, *Estudo e Desenvolvimento de Sistemas Multiagentes usando JADE*. Monografia de conclusão de curso, CCT, Faculdade de Informática, UNIFOR

BAADER, F., CALVANESE, D., MCGUINESS, D. L., NARDI, D., PATEL-SCHNEIDER, P. F., 2003, *The Description Logic Handbook: Theory, Implementation, Applications*. Cambridge, UK, Cambridge University Press.

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

RIJSBERGEN, CJ. VAN, 1979, *Information Retrieval*. MA, USA, Butterworth-Heinemann Newton.

BBS, 2006 "Historical BBS list", disponível em: http://bbslist.textfiles.com, acessado em 13/11/2006.

BELKIN, J.N., CROFT, B.W., 1992, "Information Retrieval and Information Filtering: Two sides of the same Coin?". In: *Comunications of the ACM*, v.35, n.12 (Dec), pp. 29-38.

BERNERS-LEE, T., HENDLER, J., LASSILA, O., 2001, *The Semantic Web,* Scientific American.

C, 2003, "The Development of the C Language", disponível em: http://cm.bell-labs.com/cm/cs/who/dmr/chist.html, acessado em 13/04/2005.

CHANDRASEKARAN, B., JOSEPHSON, J. BENJAMINS, V.R., 1999, "Ontologies: What are they? Why do we need them". *IEEE Intelligent Systems and Their Applications*.

CMEMORY, 2006, "Community Memory", disponível em: http://www.well.com/~szpak/cm/index.html, acessado em 13/11/2006.

DARPA, 2002, "Defense Advanced Research Projects Agency (DARPA): DAML+OIL", disponível em: http://www.daml.org/2001/03/daml+oil-index.html, acessada em 13/04/2005.

DARPA, 2005, "Defense Advanced Research Projects Agency (DARPA): DARPA Agent Markup Language (DAML)", disponível em: http://www.daml.org/index.html, acessada em 13/04/2005.

DAVIES , J., DUKE, A., SURE, Y., 2002, "OntoShare: Using Ontologies for Knowledge Sharing". *Proceedings of the WWW2002 Semantic Web workshop, 11th International WWW Conference WWW2002*, Hawaii, USA.

DONALD, M., 1991, *Origins of the Modern Mind: Three Stages in the Evolution of Culture and Cognition*, Cambridge, MA, Harvard University Press.

ETZIONI, O., WELD, D.S., 1995, "Intelligent Agents on the Internet: Fact, Fiction and Forecast", *IEEE Expert*, v.10, n.4 (Aug), pp. 44-49.

FAQ,2005, "Faq", disponível em: http://www.faqs.org, acessado em 11/04/2005.

FIPA, 2007, "Foundation for Intelligent Physical Agents", disponível em: http://www.fipa.org/, acessado em 11/01/2007

FORUM, 2005, "Internet Forum", disponível em: http://en.wikipedia.org/wiki/Internet_forum, acessado em 12/04/2005.

FRANKLIN, S., GRAESSER, A., 1996, "Is it an Agent or just a program? A Taxonomy for autonomous agents". *Proceedings of the Third international workshop on agent theories, architectures and languages*, New York, Springer Verlag.

GENESERETH, M. R., KETCHPEL, S. P., 1994, "Software Agents". In: *Communications of the ACM*, v.37, n.7, pp. 48-53.

GILBERT, D., APARICIO, M., ATKINSON, B., BRADY, S., CICCARINO, J., GROSOF, B., O'CONNOR, P., OSISEK, D., PRITKO, S., SPAGNA, R., WILSON, L., 1995, "IBM intelligent agent strategy", IBM Corporation.

GONZALES, M., 2000, *O Léxico Gerativo de Pustejovsky*. Trabalho Individual, PPGCC, Faculdade de Informática, PUCRS.

GONZALEZ, M., STRUBE DE LIMA, V. L., 2001, "Recuperação de Informação e Expansão Automática de Consulta com Thesaurus", In *XXVII Conferencia Latinoamericana de informatica*, v.1, pp. 68-75

GOOGLEAPI, 2006, "Google Api" disponível em: http://www.google.com/apis, acessado em 20/11/2006.

GOOGLEGROUPS,2005 "Google Groups". disponível em: http://groups-beta.google.com, acessado em 12/04/2005.

GRUBER, T. R., 1993, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing". *International Workshop on Formal Ontology.*

GUJ, 2006, "Grupo de Usuários Java", disponível em: http://www.guj.com.br, acessado em 14/05/2006.

HAWKING, D., CRASWELL, N., THISTLEWAITE, P., HARMAN, D., 1999 "Results and challenges in web search evaluation", *WWW8*, pp. 243-252.

HILDRETH, P., KIMBLE, C., WRIGHT, P., 2000, "Communities of Practice in the Distributed International Environment", *Journal of Knowledge Management,* v.4, n.1, pp.27–37.

JADE, 2007, "Java Agent DEvelopment Framework", disponível em: http://jade.tilab.com/, acessado em 11/01/2007

JDBC, 2007, "JDBC API", disponível em: http://java.sun.com/javase/technologies/database/, acessada em 11/01/2007

JFORUM, 2007, "JForum", disponível em: http://www.jforum.net/, acessado em: 11/01/2007

JAVA, 1995 "Java Technology", disponível em: http://java.sun.com/, acessado em 12/04/2005.

JUG, 2005, "Java User Groups", disponível em: http://java.sun.com/jugs/, acessado em 13/04/2005.

KLINK, S., 2001, "Query Reformulation with Collaborative Concept-Based Expansion".*Proceedings of the first international workshop on web document analysis*, Seattle, WA.

LANDAUER, T., DUMAIS, S., 1997, "A solution to plato's problem: The latent semantic analysis theory of acquisition", *Psychological Review.*

LINUX, 2005, "Linux" disponível em: http://www.linux.org/, acessado em 13/04/2005.

LUEG, C., 2000, "Where is the Action in Virtual Communities of Practice?" *In Proceedings of the Workshop Communication and Cooperation in Knowledge Communities at the D-CSCW,* Munique, Alemanha

MARON, M., KUHNS J., 1960, "On Relevance, Probabilistic, Indexing and Information Retrieval", *Association for Computing Machinery*, v.7, n.3, pp.216-244.

MATHES, A., 2004, "Folksonomies – Cooperative Classification and Communication Through Shared Metadata", disponível em: http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html, acessado em 11/11/2006.

MIZOGUCHI, R., IKEDA, M., 1996, *Towards ontology engineering*, In: Technical Report AI-TR-96-1, ISIR, Osaka University.

MOULIN, B., CHAIB-DRAA, B., 1996, "An Overview of Distributed Artificial Intelligence". In: G.M.P. O'Hare e N.R. Jennings, *Foundations of Distributed Artificial Intelligence*, chapter 1, New York, Wiley.

MYSQL, 2006, "MySQL", disponível em: http://www.mysql.com/, acessado em 22/06/2006

MYSQLFULLTEXT, 2006, disponível em: http://dev.mysql.com/doc/internals/en/full-text-search.html, acessado em 22/06/2006

NAVIGLI, R., VELARDI, P., 2003, "An Analysis of Ontology-based Query Expansion Strategies". *Workshop on Adaptive Text Extraction and Mining at the 14þ European Conference on Machine Learning.*

NIST, 2006, "National Institute of Standards and Technology. TREC home page", disponível em http://trec.nist.gov/, acessado em: 11/06/2006

NONAKA, I., TAKEUCHI, H., 1995, *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*, New York, Oxford Univ. Press.

NORMAN, D.A., 1993, *Things That Make Us Smart*, Reading, MA, Addison-Wesley Publishing Company.

NOY, N., MCGUINNESS, D. L., 2001, *Ontology development 101: A guide to creating your first ontology*. In: Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, Stanford Medical Informatics.

NWANA, H., 1996, "Software Agents: An Overview". *Knowledge Engineering Review*, v.11, n.3 (Oct-Nov), pp. 205-244.

ONTOKNOWLEDGE, 2000, "OntoKnowledge: Ontology Inference Layer (OIL)", disponível em: http://www.ontoknowledge.org/oil/oilhome.shtml, acessado em 03/11/2004 .

W3C, 2004a, "World Wide Web Consortium (W3C): Web Ontology Language (OWL)", disponível em: http://www.w3.org/TR/owl-ref, acessado em 03/11/2004.

W3C, 2004b, "World Wide Web Consortium (W3C): OWL Web Ontology Language Overview", disponível em: http://www.w3.org/TR/owl-features/, acessado em 03/11/2004.

W3C, 2004c, disponível em: http://www.w3.org/TR/owl-guide/, acessado em 03/11/2004.

PEREIRA, V., REZENDE, J., XEXÉO, G., SOUZA, J., 2006, "Building a Personal Knowledge Recommendation System using Agents, Learning Ontologies and

Web Mining", *The 10th International Conference on CSCW in Design*, Southeast University, Nanjing, China.

PINHEIRO, W. A., 2004, "An Ontology Based-Approach for Semantic Search in Portals" In: *Anais do 15º International Workshop on Database and Expert Systems Applications (DEXA)*, pp.127-131, Zaragoza.

POLANYI, M., 1983, *The Tacit Dimension.* Peter Smith Pub, Gloucester, MA, London Press.

POSTGRES, 2007, "PostGres", disponível em: http://www.postgresql.org/, acessado em 11/01/2007

PUSTEJOVSKY, J., 1995, *The Generative Lexicon,* Cambridge, The MIT Press.

W3C, 2005a, "World Wide Web Consortium (W3C): Resource Description Framework", disponível em: http://www.w3.org/RDF/, acessado em 03/11/2004.

W3C, 2004d, "World Wide Web Consortium (W3C): RDF Schema (RDFS)" disponível em: http://www.w3.org/TR/rdf-schema/, acessado em 03/11/2004.

SCHÖN, D.A., 1983, *The Reflective Practitioner: How Professionals Think in Action*, New York, NY, Basic Books.

RHEINGOLD,H., 2003, *The Virtual Community: Homesteading on the Electronic Frontier.* New York, Addison-Wesley.

ROCCHIO, J.,J., 1971, "Relevance Feedback on Information Retrieval", In G. Salton, *The SMART Retrieval System – Experiments in Automatic Document Processing*, chapter 14, Englewood Cliffs, NJ, Prentice Hall Inc.

RUTHVEN, I., 2003, "Re-Examining the Potential Effectiveness of Interactive Query Expansion", In *Proceedings of the 26th acm sigir conference on research and development in information retrieval,* pp. 213–220.

SALTON, G., BUCCKLEY, C., 1988, "Term-weighting approaches in Automatic Retrieval". *Information Processing & Management*, v.24, n.5, pp. 513-523.

SCHACHTER, J., 2004, "Del.icio.us About Page", disponível em: http://del.icio.us/doc/about, acessado em 11/11/2006.

W3C, 2005b, disponível em: http://www.w3.org/XML/Schema, acessado em 13/07/2005.

SEELY BROWN, J., DUGUID, S., 2000, *The Social Life of Information.* Boston, Massachusetts, Harvard Business School Press.

SHOHAM, Y., 1990, *Agent-oriented programming.* In: Technical Report STAN–CS–1335–90,Computer Science Department, Stanford University, Stanford, CA 94305.

SHOHAM, Y., 1997, *An overview of agent-oriented programming*, In: Bradshaw, JM, Software Agents, Cambridge, AAAI Press/ MIT Press, 1997, pp.271-290.

SINGHAL, A., BUCKLEY, C., MITRA, M., 1996, "Pivoted document length normalization" In: *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 21–29. Association for Computing Machinery, New York.

STAHL, G., 1993, *Interpretation in Design: The Problem of Tacit and Explicit Understanding in Computer Support of Cooperative Design*, Ph.D. dissertation, Department of Computer Science, University of Colorado, Boulder, CO. disponível em: http://www.cs.colorado.edu/~gerry/publications/dissertations/dis_intro.html.

STAHL, G., 2000, "Collaborative Information Enviroments to Support Knowledge Construction by Communities", *AI &Society,* v.14, n.1 (Mar), pp. 71-97.

SUN, 2005, "Sun Microsystems" disponível em: http://www.sun.com, acessado em 13/04/2005.

USENET,2005, "USENET". disponível em: http://www.ibiblio.org/usenet-i/usenet-help.html, acessado em 11/04/2005.

VAN RIJSBERGEN, C.J., 1979, *Information Retrieval*, 2 ed. MA, USA, Butterworths.

WENGER, E., 2005, "Communities of practice, a brief introducion". disponível em: http://www.ewenger.com/theory/communities_of_practice_intro.htm, acessado em 11/04/2005.

WENGER,E., 1998, "Communities of Practice. Learning as a Social System". *Systems Thinker.* disponível em: http://www.co-i-l.com/coil/knowledge-garden/cop/lss.shtml, acessado em 11/04/2005

WILKS, Y. A., SLATOR, B. M., GUTHRIE, 1996., *Louise M. Electric Words: Dictionaries, Computers, and Meanings*. Cambridge, The MIT Press

WINDOWS, 2005, "Microsoft Windows", disponível em: http://www.microsoft.com/, acessado em 13/04/2005

WOOLDRIDGE, M., JENNINGS N.R., 1995, *Intelligent Agents: Theory and Practice,* The Knowledge Engineering Review

W3C, 2005c, "XML Especification" disponível em: http://www.w3.org/XML/, acessado em 13/04/2005

YAHOOGROUPS, 2005, "Yahoo! Groups". disponível em: http://groups.yahoo.com/, acessado em 12/04/2005