UNIVERSIDADE DE CAXIAS DO SUL
Centro de Ciências Exatas e Tecnologia
Curso de Bacharelado em Ciência da Computação

Daniela Maria Uez

UM *WEB SEARCHER AGENT* BASEADO EM REGRAS CONTEXTUAIS E NA COLABORAÇÃO ENTRE AGENTES

Caxias do Sul

2009

Daniela Maria Uez

UM *WEB SEARCHER AGENT* BASEADO EM REGRAS CONTEXTUAIS E NA COLABORAÇÃO ENTRE AGENTES

Trabalho de Conclusão de Curso para obtenção do Grau de Bacharel em Ciência da Computação da Universidade de Caxias do Sul.

João Luis Tavares da Silva
Orientador

Caxias do Sul

2009

À minha mãe, Maria.

À minha avó, Ida.

Ao meu irmão, Pablo.

Ao meu namorado, Cleverson.

AGRADECIMENTOS

Agradeço ao meu orientador, o professor João Luis Tavares da Silva, que com muita paciência me orientou e permitiu a realização deste trabalho e também ao professor Maurício Escobar pelos esclarecimentos que permitiram que eu utilizasse o *framework* SemantiCore para desenvolvimento dos agentes.

Pela dedicação e carinho que tem me dado todos esses anos e que me permitiram chegar até aqui, minha mãe, Maria, merece um agradecimento especial. Agradeço também ao meu irmão Pablo pelo apoio e ao meu namorado Cleverson pela compreensão e paciência.

Essa jornada não teria se completado sem o auxílio de grandes amigos. Diego, Caio e Zeze, valeu pelo apoio! Marcio, obrigada pela amizade. Chan, obrigada pela paciência e por me ouvir nas crises existenciais!

Muito obrigada a todos os amigos e familiares que me incentivaram e me ajudaram para conclusão deste trabalho. Por fim, agradeço a Deus que tem olhado por mim em todos os momentos e colocou tantas pessoas especiais em minha vida.

RESUMO

Este documento apresenta um estudo para desenvolvimento de um agente que realizará a busca de informações na *Web* utilizando filtragem semântica. O agente descrito utiliza regras para filtragem dos resultados que são criadas com base em uma ontologia do contexto no qual deve ser realizada a busca. Este estudo foi desenvolvido através de uma pesquisa bibliográfica e da implementação do agente para teste dos resultados. A pesquisa bibliográfica forneceu as bases para o desenvolvimento do agente, que foi implementado utilizando o *framework* SemantiCore e a API Jena. O SemantiCore foi utilizado para o desenvolvimento dos agentes e das interações entre os agentes. A API Jena permitiu que fosse realizada a leitura e a manipulação da ontologia utilizada para filtragem semântica. Através deste estudo chegou-se à conclusão de que a busca baseada em regras semânticas definidas por ontologias é uma alternativa viável para melhoria dos resultados apresentados pelas ferramentas de busca atuais.

Palavras-chaves: Sistema multiagentes. Busca na *Web*. Busca Semântica. Ontologia.

ABSTRACT

This paper presents a study to develop an Web searcher agent for semantic information retrieval on the Web. This agent uses semantic rules defined by an ontology to filter the search context. For this study was made a bibliographic search and the agent was developed to test the results. The bibliographic search provid the pilars to develop the agent. The agent was developed using the SemantiCore framework and Jena API. SemantiCore framework was used to develop the agents and its interactions. The Jena API enabled the reading and manipulation of the ontology used for semantic filtering. This study concludes that searching using semantic rules defined by ontologies is a viable alternative to improve web search results.

Keywords: Multiagent systems. Web search. Semantic search. Ontology.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

Sigla	Significado
ACL	Agent Communication Language
API	Application Programming Interface
BDI	Belief-Desire-Intention
FTP	File Transfer Protocol
HTML	Hipertext Markup Language
OWL	Ontology Web Language
PDF	Portable Document Format
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SMA	Sistema Multiagente
SOAP	Simple Object Access Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WWW	World Wide Web

SUMÁRIO

1 INTRODUÇÃO

As ferramentas de busca são *sites* especiais desenvolvidos para ajudar na recuperação de informações armazenadas em outros *sites* (GUIMARÃES, 2002). Quando um usuário faz uma solicitação de busca, a ferramenta compara as palavras-chave digitadas pelo usuário com as entradas do índice de *sites* em sua base de dados e depois classifica o resultado de acordo com algoritmos próprios de classificação. As ferramentas de busca não conseguem filtrar com eficiência as informações porque utilizam como método básico de busca a comparação sintática entre os termos. Os métodos de classificação levam em conta aspectos como a quantidade de vezes que a palavra-chave é repetida na página e a quantidade de *links* que apontam para o *site*. Essas ferramentas não tratam casos em que as palavras-chave têm mais de um significado e não levam em conta as relações semânticas entre as palavras. Por isso o resultado apresentado normalmente não é satisfatório. Uma busca simples retorna uma quantidade muito grande de *sites*, sendo que muitos não têm relevância para o usuário ao mesmo tempo em que *sites* importantes podem ser ignorados devido aos processos de classificação que a ferramenta utiliza.

Uma forma de melhorar o resultado apresentado pelas ferramentas de busca é alterar a estrutura das páginas da *Web* acrescentando informações que possam ser utilizadas por essas ferramentas para filtrar e classificar eficientemente o conteúdo de cada *site*. Essa é a proposta da *Web* Semântica. Segundo (OLIVEIRA, 2002):

> A *Web* Semântica será uma extensão da *Web* atual porém apresentará estrutura que possibilitará a compreensão e o gerenciamento dos conteúdos armazenados na *Web* independente da forma em que estes se apresentem, seja texto, som, imagem e gráficos à partir da valoração semântica desses conteúdos, e através de agentes que serão programas coletores de conteúdo advindos de fontes diversas capazes de processar as informações e permutar resultados com outros programas.

A *Web* semântica, então, acrescentará informações estruturadas sobre a semântica do conteúdo às páginas da *Web*. Para tornar a *Web* semântica real, é necessária a definição da estrutura semântica das páginas, os conceitos e as relações entre os conceitos. É importante também que essas definições sejam comuns a todas as páginas, garantindo assim que o conteúdo seja compreendido de forma igual por todas as ferramentas. A especificação formal e explícita dos conceitos é chamada de ontologia. Uma ontologia é um conjunto de regras de conhecimento que inclui o vocabulário, as conexões semânticas e algumas regras simples de inferência para tópicos particulares (HENDLER,2001). As ontologias definem um vocabulário comum que é compreendido da mesma forma por diversas aplicações, tornando possível a troca de informações e a colaboração

entre elas. Assim, agentes que não foram desenvolvidos especificamente para trabalharem juntos poderão cooperar entre si.

Outra alternativa para melhorar a performance das ferramentas de recuperação de informações na *Web* é a utilização de sistemas multiagentes. Um sistema multiagentes (SMA) é um sistema compostos por dois ou mais agentes autônomos e organizados que cooperam entre si na resolução de problemas que não podem ser resolvidos por cada um individualmente (HUBNER, 2003). Em um sistema multiagentes cada agente existe e funciona de forma independente dos demais, mas pode interagir com os outros na busca de soluções para um problema comum ou de auxílio para resolução de um problema individual. No contexto da *Web* semântica agentes inteligentes são usados para recuperação de informações a partir de diversas fontes heterogêneas ou para tratamento de informações de busca em paralelo. Abordagens do tipo *webcrawlers* ou *metacrawlers* realizam simultaneamente a busca de informações em várias fontes, comparando URL's a partir de critérios de relevância pré-estabelecidos (repetição dos termos, frequência no documento, número de visitas, etc.). Para realização dessa tarefa vários agentes podem ser utilizados de forma distribuída e paralela. Essa abordagem de busca, porém, limita a relevância a uma análise quantitativa das URL's.

Outra abordagem para recuperação de informações na *Web* é a utilização de agentes com ontologias que permitirão a categorização da informação buscada. Nessa abordagem, os vários agentes enviados na realização da busca possuem regras ligadas ao contexto do assunto a ser buscado. Os resultados encontrados individualmente pelos agentes serão filtrados através da combinação das regras que cada agente possui. Dessa forma, a busca levará em conta também a semântica dos termos envolvidos e o contexto para o qual a busca está sendo realizada. O resultado apresentado, portanto, tende a ser mais refinado e próximo das necessidades do usuário. Para o desenvolvimento eficiente dessa abordagem de busca é importante a definição correta dos agentes. Os agentes serão responsáveis pela realização da busca na *Web* e pela filtragem dos resultados. É importante também a especificação correta das ontologias. São elas que definirão as regras contextuais utilizadas para filtragem dos resultados da busca e permitirão a troca das informações entre os agentes.

O principal objetivo deste trabalho é realizar um estudo para definição de um sistema multiagentes que permita a recuperação de informações na *Internet* utilizando regras contextuais para filtragem semântica dos resultados. Para permitir que o objetivo seja

plenamente alcançado, será realizado um estudo sobre o funcionamento dos mecanismos atuais de busca na *Web*, sobre o comportamento dos agentes inteligentes em sistemas multiagentes e sobre o uso de ontologias.

Este trabalho está estruturado da seguinte forma: no capítulo 2 é apresentado um estudo sobre o funcionamento das ferramentas tradicionais de busca na *Web,* no capítulo 3 são apresentados os agentes e os sistemas multiagentes e, no capítulo 4, é feito um estudo sobre as ontologias. Por fim, no capítulo 5 é apresentado o agente de busca contextual e os resultados alcançados com uso do agente.

2 RECUPERAÇÃO DE INFORMAÇÕES NA *INTERNET*

A *Internet* é como uma enorme biblioteca onde podem ser encontradas informações sobre qualquer assunto. Porém, encontrar essas informações pode não ser uma tarefa muito simples, já que as páginas da *Internet* não possuem nenhum tipo de organização que permita encontrar rapidamente uma informação. Assim, para evitar que o usuário tenha que percorrer as páginas uma a uma quando precisar de alguma informação, foram criadas ferramentas para auxiliar na busca por páginas específicas na *Internet.* Essas ferramentas, chamadas *search engines* ou mecanismos de busca, percorrem as páginas procurando por aquelas que possuam as palavras ou expressões informadas pelo usuário e em pouco tempo retornam uma lista contendo o endereço e uma pequena descrição dos *sites* que encontram.

Segundo NORVIG (2004), para poder atender com rapidez às requisições dos usuários, as ferramentas de busca executam quatro passos básicos:

1. Programas chamados *spiders* percorrem a *Internet,* seguindo os *links*, em busca das páginas que serão armazenadas na ferramenta;
2. As páginas encontradas pelos *spiders* são indexadas para permitir que sejam recuperadas quando o usuário fizer uma requisição de busca;
3. Quando um usuário faz uma requisição o mecanismo varre o índice em busca de páginas que contenham as palavras informadas pelo usuário. As páginas encontradas são ordenadas de acordo com critérios de classificação próprios de cada ferramenta com o intuito de determinar o grau de importância de cada página e a relevância que o conteúdo terá para o usuário. As páginas mais importantes são apresentadas primeiro;
4. Os resultados são mostrados de forma simples e ordenada para os usuários.

A Figura 1 apresenta os passos de funcionamento dos mecanismos de busca.

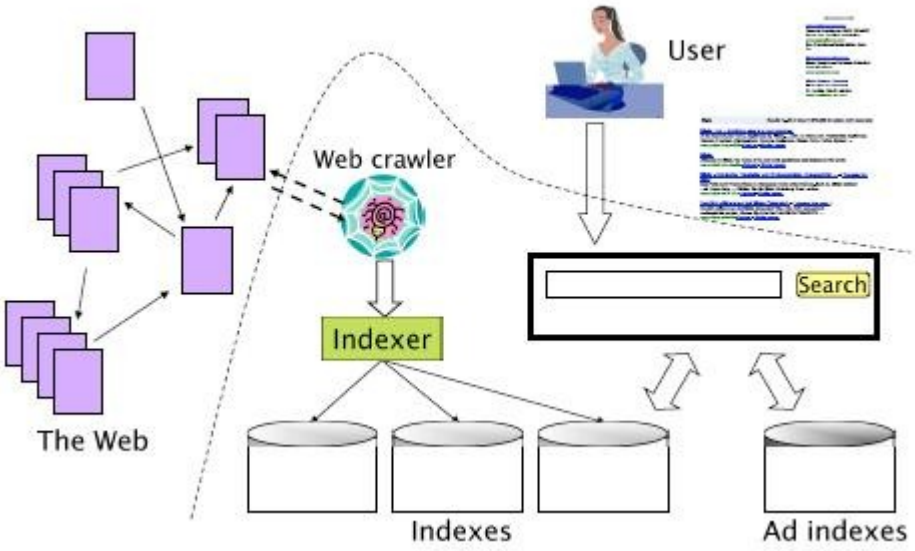

Figura 1: Funcionamento dos mecanismos de busca (MANNING, 2008)

2.1 SPIDERS

Também chamados de *web spiders , robots, bots, crawlers* ou *web crawlers,* os *spiders* são programas que percorrem a *Internet* automaticamente, seguindo os *links* existentes nas páginas. A partir de uma lista de *sites* conhecidos, chamada de *seed* (semente), os *spiders* visitam as páginas da *Internet* procurando por *links* que serão acrescentados à lista de *sites* a percorrer. Cada página visitada pelo *spider* é copiada para que a ferramenta de busca possa indexá-la posteriormente. (FREITAS, 1997) apresenta o algoritmo que ilustra o funcionamento básico de um *spider*:

1. Inicia a busca a partir de um conjunto conhecido de documentos;
2. Descobre novos documentos;
3. Marca os documentos que são retidos (adicionados para a base de dados de índices);
4. Descobre seus índices;
5. Indexa o conteúdo do documento.

Na prática, porém, a tarefa dos *spiders* é muito mais complexa, já que eles têm que

lidar com uma série de problemas que podem ocorrer durante a navegação pela *Internet*. Como o volume de informações existentes na *Web* é muito grande e todo o conteúdo está em constante atualização, o *spider* pode, por exemplo, não conseguir acessar algumas das páginas que tenta visitar. Algumas vezes a indisponibilidade ocorre porque a página deixou de existir, mas em outros casos a página pode simplesmente estar indisponível devido ao grande número de acessos ou por algum outro problema técnico no servidor. Essas páginas voltarão a funcionar posteriormente e o *spider* terá que visitá-la novamente para poder indexá-la. Caso a página tenha sido excluída, o *spider* terá que remover esta página do índice para que ela não conste mais na lista de resultados. Também existem os casos em que uma página está hospedada ao mesmo tempo em *sites* diferentes ou que a mesma página pode ser acessada através de URLs diferentes no mesmo *site*. O *spider*, nestes casos, deve cuidar para que as páginas repetidas não sejam indexadas mais de uma vez, o que comprometeria o resultado apresentado para o usuário.

Outro problema que o *spider* enfrenta são as páginas que não seguem os padrões da linguagem HTML, normalmente desenvolvidas por pessoas com pouco conhecimento na linguagem. Essas páginas não são processadas corretamente pela ferramenta e podem levar o mecanismo a erros de indexação. Páginas dinâmicas, que dependem do preenchimento de formulários para serem geradas, também não podem ser indexadas, da mesma forma que as páginas que utilizam animações ou imagens ao invés de textos. Existem também as páginas que se utilizam de protocolos que impedem a indexação (protocolos de exclusão de robôs). Além disso, o *spider* deve levar em conta que, como o conteúdo na *Web* está em constante atualização, no momento em que terminar de analisar um *site* ou um conjunto de *sites* é muito provável que as páginas visitadas já tenham sido alteradas ou que outras páginas tenham sido incluídas no mesmo *site*

Para poder solucionar estes problemas o *spider* utiliza alguns mecanismos. Por exemplo, para evitar que uma mesma página seja copiada mais de uma vez, os *spiders* costumam utilizar funções de normalização das URLs. A normalização inclui a conversão da URL para letras minúsculas, a remoção de segmentos (“.” ou “..”) e a inclusão de barras invertidas nos caminhos não-vazios. Os *spiders* também seguem políticas de *crawling,* que são políticas que determinam o comportamento do *spider* durante a navegação pelos *sites.* As políticas de *crawling* usadas normalmente são a *política de seleção*, a *política de revisitação*, a *política de cortesia* e a *política de paralelização*.

2.1.1 POLÍTICA DE SELEÇÃO

Todos os dias, milhares de páginas são adicionadas às milhões já existentes na *Internet*. Dessa forma, é impossível para qualquer mecanismo de busca armazenar todo o conteúdo disponível. Além disso, o *spider* só consegue visitar uma pequena quantidade de páginas em um determinado tempo. Isso significa que quando o *spider* terminar a visitação de um *site*, muitas alterações já ocorreram nos *sites* visitados anteriormente. Diante dessas limitações, o mecanismo deve garantir que as páginas visitadas tenham conteúdos relevantes para os usuários e não sejam somente um conjunto aleatório de páginas. Para isto, o mecanismo tenta classificar as páginas que serão visitadas com base em métricas de importância. Segundo (CASTILLO, 2004):

> A importância de uma página é uma função com uma qualidade intrínseca, sua popularidade em termos de *links* ou visitas e até mesmo de sua URL (este último é o caso de *search engines* verticais restritos a um único domínio *top-level*, ou *search engines* restritos a um *Web site* fixo).

Criar políticas de seleção eficientes é uma tarefa com alto nível de dificuldade pois, como o número total de páginas da *Web* não é conhecido durante a busca, o algoritmo de seleção deve trabalhar somente com uma parte da informação.

Como forma de restringir a busca, os *spiders* procuram seguir somente *links* que levam para páginas HTML, eliminando outros tipos de protocolo que circulam pela *Internet* (correio eletrônico, FTP, etc). Para determinar o tipo de página, o *spider* compara a extensão do *link* com uma lista de extensões de páginas HTML conhecidas, como .html, .htm, .php, .asp, entre outras. Normalmente os mecanismos de busca possuem algoritmos específicos para analisar outros tipos de documentos, como .pdf, .doc ou .txt. Nestes casos o *spider* envia a URL do *site* para o mecanismo responsável pela sua análise.

Alguns *spiders* também evitam seguir *links* que possuam uma interrogação ("?") na URL. Esse caractere indica que a página é criada dinamicamente, de acordo com o preenchimento de um formulário ou com alguma informação dada pelo usuário. *Links* desse tipo podem fazer com que o *spider* entre em um *loop* infinito de captura de URLs dentro de um mesmo *site*, por isso são evitados.

2.1.2 POLÍTICA DE REVISITAÇÃO

O *spider* precisa de muito tempo para percorrer uma parte da *Internet*. Assim, devido à natureza extremamente dinâmica da rede, no momento em que o *spider* terminar sua tarefa em uma parte da *Web*, muitas páginas podem ter sido atualizadas, excluídas ou incluídas nos *sites* por onde o *spider* passou. Para o mecanismo de busca não é interessante ter conteúdo desatualizado em sua base de dados. Por isso o *spider* precisa se preocupar com a idade - quanto tempo faz que a cópia foi atualizada – e com a atualidade – quantas cópias estão desatualizadas – das páginas que a ferramenta possui em seu repositório. Segundo (CASTILLO, 2004), "O objetivo do *crawler* é manter a média de atualização das páginas em sua coleção o mais alto possível ou manter a média de idade das páginas o mais baixo possível" .

Os dois tipos principais de políticas de revisitação são a *política uniforme* e a *política proporcional*. Pela política uniforme o *spider* revisita todas as páginas da coleção com a mesma frequência. Pela política proporcional o *spider* revisita com mais frequência as páginas que são modificadas mais frequentemente. Estudos mostram que a política uniforme mantém a média de atualização melhor do que a política proporcional. Isso se deve ao fato de que quando uma página é frequentemente atualizada o *spider* gasta muito tempo tentando refazer a cópia da mesma página a cada atualização e não vai conseguir manter as outras páginas atualizadas.

2.1.3 POLÍTICA DE CORTESIA

Os *spiders* facilitam a realização de muitas tarefas. Porém, essas ferramentas usam quantidades muito grandes de largura de banda e paralelismo para funcionarem corretamente. Além disso, quando muitas requisições de páginas são feitas para o mesmo servidor este pode ficar sobrecarregado. Para evitar esses problemas criaram-se protocolos para organizar o acesso às páginas de um servidor. O mais conhecido desses protocolos é o protocolo de exclusão de robôs. O protocolo de exclusão de robôs informa o *spider* sobre quais páginas de um *site* podem ou não ser indexadas e se os *links* existentes nessas páginas devem ou não ser seguidos. O protocolo de exclusão de robôs, contudo, não especifica intervalos de tempo entre as requisições para um mesmo servidor. O intervalo de tempo é a forma mais efetiva de evitar

a sobrecarga dos servidores. Porém, o uso de intervalos muito grandes entre uma requisição e outra faz com que a indexação dos *sites* leve muito tempo para ser realizada.

2.1.4 POLÍTICA DE PARALELIZAÇÃO

Para conseguir visitar um número maior de páginas da *Web* os *spiders* normalmente rodam múltiplos processos em paralelo. O problema da paralelização é que uma mesma página pode ser obtida múltiplas vezes por processos diferentes. Como forma de minimizar a duplicidade de páginas, os *spiders* utilizam-se de políticas de atribuição de endereços a serem visitados. Na *atribuição dinâmica* um servidor central envia ao *spider* os endereços a serem acessados dinamicamente. Esse sistema permite o controle da carga que é distribuída a cada *spider* e a adição ou remoção dinâmica de endereços. Na *atribuição estática* os endereços são atribuídos aos *spiders* no início da varredura. Em alguns momentos pode haver troca de URLs entre os *spiders*, como por exemplo em casos em que uma URL de um primeiro processo acessa um *site* atribuído a um segundo processo.

2.2 INDEXAÇÃO

Todas as páginas visitadas pelo *spider* são enviadas para o indexador onde serão analisadas para criação dos índices. São os índices que permitem a recuperação rápida das informações quando é feita uma requisição de busca para o mecanismo.

Para criar o índice, o indexador associa cada palavra às páginas onde elas acontecem. Alguns mecanismos de busca criam índices específicos para expressões, citações ou frases, dependendo das suas necessidades de busca. Além das palavras e das páginas, os mecanismos de busca também armazenam informações relevantes sobre cada entrada do índice, como a quantidade de vezes que a palavra aparece no texto, a posição na qual aparece, o tipo e o tamanho da fonte com a qual foi escrita, entre outras. Essas informações, juntamente com algumas informações sobre a página (como a quantidade de *links* que levam à essa página) são utilizadas para calcular a importância do *site.* Os mecanismos de busca utilizam algoritmos especiais para realizar esse cálculo, como o algoritmo *Page Rank* utilizado pela ferramenta de busca da *Google* (GOOGLE, 2009).

Com tantas informações, o maior problema dos índices é o tamanho. Mesmo utilizando-se técnicas de compactação, os índices chegam a utilizar alguns *terabytes* de espaço para serem armazenados. Portanto, os mecanismos precisam utilizar estruturas de dados que permitam armazenar uma quantidade grande de informações e recuperá-las rapidamente quando necessário. Normalmente as estruturas de dados utilizadas para armazenar os índices são as árvores ou matrizes. Porém, cada mecanismo de busca estrutura o índice da forma mais conveniente, de acordo com suas necessidades de busca.

2.3 RETORNANDO OS RESULTADOS PARA O USUÁRIO

Os mecanismos de busca executam a varredura da *Web* e a indexação das páginas mesmo que não haja nenhuma requisição de usuário para ser respondida. Essas etapas são executadas automaticamente de forma paralela às solicitações dos usuários. Por isso, quando um usuário faz uma solicitação de busca, o mecanismo não precisa buscar pelos *sites* diretamente na *Internet*. A busca é feita nos *sites* que a ferramenta já indexou, o que permite responder ao usuário com rapidez.

Supondo que um usuário digite em um mecanismo de busca as palavras *"inteligência artificial" sistema especialista.* Nesta solicitação, as palavras entre aspas são entendidas pelo mecanismo como uma expressão, ou seja, devem constar no *site* na mesma ordem na qual foram escritas. Segundo (NORVIG, 2004) para responder à solicitação deste usuário o mecanismo primeiro procura em seus índices as listas de *sites* que contêm cada uma das palavras: “inteligência”, “artificial”, “sistema” e “especialista”. Depois o mecanismo confere as listas e seleciona somente os *sites* onde todas as palavras estão presentes. Como as palavras “inteligência” e “artificial” foram escritas em forma de expressão, o mecanismo ignora os *sites* nos quais essas palavras não foram escritas em sequência.

O passo seguinte é ordenar os *sites* encontrados de forma que aqueles considerados mais relevantes sejam apresentados primeiro. Normalmente a ordenação é feita pela quantidade de ocorrências de cada palavra no *site* porém alguns mecanismos podem levar em conta outras informações como a proximidade entre as palavras, em que lugar da página as palavras aparecem (no título, no início do texto, no *link* que leva à página), se estão em destaque (em negrito, em fontes maiores, etc) e a quantidade de *links* que apontam para a página.

O passo final é apresentar os resultados para os usuários. É comum que os resultados sejam apresentados juntamente com uma pequena descrição do *site*. Nessa descrição costumam ser incluídos o título, a URL e algumas linhas representando o conteúdo do *site* onde as palavras-chave utilizadas na pesquisa são destacadas. Na Figura 2 pode-se ver alguns resultados apresentados por um mecanismo de busca em resposta à solicitação de usuário, com o título das páginas selecionadas, uma pequena descrição de cada página e a URL que leva à pagina.

Figura 2: Resultado apresentadas pelo mecanismo de busca Google (GOOGLE, 2009)

Para o mecanismo de busca é muito importante que os resultados apresentados nas páginas iniciais sejam relevantes. "A eficiência de um buscador será avaliada pela sua capacidade em apresentar, logo nas primeiras linhas, informações que atendam às necessidades do usuário" (BRANSKI, 2004). Porém, devido às características da *Internet* atual é muito comum o retorno de *site*s que não possuem nenhuma relação com a busca do usuário. Por exemplo, se um usuário que busque informações sobre agentes inteligentes no contexto da Inteligência Artificial digitar em um mecanismo de buscas somente a palavra *agentes, e*m menos de 1 segundo o mecanismo retornará alguns milhões de páginas que

contêm a palavra digitada. Dentre os resultados o usuário encontrará *sites* sobre "agentes federais", "agentes de viagens", "agentes fiscais" e alguns outros tipos de agentes que não têm relação com o tipo de informações que o usuário procura.

Os mecanismos de busca utilizam-se somente da comparação sintática entre as palavras-chave digitadas pelo usuário e as entradas nos índices para determinar se uma página deve ou não ser apresentada. A análise sintática não leva em conta os diversos significados que uma mesma palavra pode ter em linguagem natural, ou seja, não consegue diferenciar os *sites* que trazem informações sobre agentes de viagens daqueles que contêm informações sobre agentes inteligentes. Da mesma forma a análise sintática não permite que o mecanismo identifique palavras diferentes que possuem o mesmo significado (sinônimos). Por isso os mecanismos de busca podem incluir em suas listas de resultados muitos *sites* que não correspondem aos critérios solicitados pelos usuários e ignorar alguns *site*s relevantes.

Para apresentar resultados mais precisos, os mecanismos têm tentado compreender melhor a linguagem do usuário. Assim, alguns mecanismos armazenam em seu índice informações linguísticas ou léxicas sobre as palavras (RICOTTA, 2007), indexam termos e expressões utilizados em buscas anteriores e buscam automaticamente palavras ou expressões relacionadas com as digitadas pelo usuário. Por exemplo, se o usuário digitar somente a palavra *agentes,* alguns mecanismos apresentarão dentre os resultados opções para o refinamento da busca usando expressões que são frequentemente buscadas, como "*agentes inteligentes*". Da mesma forma, o mecanismo irá procurar por *site*s que possuam no conteúdo a palavra *agentes* e algumas de suas variações, como *agente* (no singular). Os mecanismos também oferecem meios de refinar a busca através da inclusão de operadores lógicos (E, OU) e do uso de comandos especiais.

Outro fator que faz com que os resultados das buscas não seja preciso é o uso incorreto das *meta-tags*. As *meta-tags* são *tags* especiais da linguagem HTML criadas para permitir a inclusão de informações relevantes sobre o conteúdo nas páginas. Essas *tags* ficam no topo da página e não podem ser visualizadas pelos usuários, por isso são utilizadas pelos mecanismos para classificar a página. Alguns desenvolvedores passaram a incluir nas *meta-tags* listas de palavras repetidas, mesmo que muitas dessas palavras não tenham relação com o conteúdo do *site*. Dessa forma os algoritmos de busca dariam uma relevância maior para o *site* e poderiam incluir esse *site* em mais resultados de busca. Os mecanismos não tem recursos para avaliar se as informações apresentadas nas *meta-tags* realmente são relevantes. Por isso os algoritmos de

classificação passaram a dar um valor menor às informações presentes nessas *tags*. Mesmo assim muitos *sites* ainda se utilizam desse recurso para serem apresentados nos resultados das buscas.

2.4 *WEB* SEMÂNTICA

Um grande problema enfrentado pelos mecanismos de busca é a falta de informações sobre o conteúdo e sobre a estrutura das páginas da *Web*. A *Internet* hoje não inclui nenhum tipo de informação que permita aos mecanismos de busca interpretar corretamente o texto que está contido nas páginas. Para os seres humanos é fácil diferenciar um *agente de viagens* de um *agente inteligente*, porém os mecanismos de busca não conseguem visualizar essa diferença.

Para solucionar esse tipo de problema o W3C (*World Wide Web Consortium)* idealizou a *Web* Semântica, que incluirá informações sobre o conteúdo e sobre a semântica nas páginas da *Internet.* Assim os agentes de *software* poderão compreender e trabalhar com as informações disponíveis nos *sites*.

> "A *Web* Semântica é uma extensão da *Web* dotada de mais significado onde qualquer usuário da *Internet* poderá encontrar respostas às suas perguntas de forma rápida e simples graças a uma informação melhor definida. Ao dotar a *Web* de mais significado e, portanto, de mais semântica pode-se obter soluções para os problemas habituais na busca de informações graças a utilização de um infraestrutura comum mediante a qual é possível compartilhar, processar e trocar informações de forma simples." (W3C, 2008)

O objetivo da *Web* semântica é estruturar as páginas inserindo informações sobre os conceitos, sobre as relações que existem entre os conceitos e sobre a semântica dos conteúdos apresentados na página, independente da forma como estes se apresentam (sons, imagens, texto, etc). Essa estruturação permitirá aos sistemas computacionais ler, compreender e analisar os conceitos para, a partir destes, inferirem novos conceitos.

O W3C está criando mecanismos que permitam o desenvolvimento de ferramentas que viabilizem a *Web* Semântica. Nesse novo ambiente a busca por conteúdo será facilitada pois as informações necessárias para que os mecanismos encontrem exatamente o que o usuário

procura estarão nas próprias páginas da *Internet*. Porém, para que as ferramentas desenvolvidas para a *Web* Semântica tenham o resultado esperado será necessário que todas as páginas existentes sejam reescritas nos padrões da nova *Web,* o que levará algum tempo para acontecer.

2.5 CONSIDERAÇÕES FINAIS

Atualmente a *Internet* é um conjunto desorganizado de informações formada por milhares de páginas. Dessa forma é complicado para os mecanismos de busca encontrar com precisão as informações solicitadas pelos usuários. A *Web* Semântica é uma alternativa para organizar as informações apresentadas na rede, porém sua implementação obrigará a alteração de todas as páginas existentes. Qualquer ferramenta desenvolvida para a *Web* Semântica somente funcionará corretamente nas páginas que seguirem o padrão desenvolvido. Por enquanto, portanto, um mecanismo de busca baseado nas regras da *Web* Semântica pode não ser mais eficiente que os mecanismos tradicionais.

Diante da realidade da *Web* atual, algumas pesquisas têm sido feitas para desenvolver mecanismos mais eficientes de busca de informações. A alternativa apresentada pelos membros do projeto *Clever* (CLEVER, 1999), mantido pela IBM, classifica os *sites* de acordo com a quantidade de *links* que levam a ele e com o grau de confiabilidade do *site* que possui este *link*. Nesta abordagem os *sites* são divididos em dois tipos: *hubs* e autoridades. Os *sites* do tipo *hub* são *sites* que possuem *links* que levam a outros *sites*. Os *sites* do tipo autoridade são os *sites* referenciados pelos hubs. Quanto mais referenciada por *hubs* confiáveis uma autoridade for, melhor será a sua classificação. A dificuldade encontrada nessa solução está em determinar quais são os *hubs* e qual é o grau de confiança de cada um deles.

Uma outra alternativa é incluir regras semânticas nos mecanismos de busca. Essas regras permitirão ao mecanismo conhecer o contexto onde a busca deverá ser realizada e seriam utilizadas para filtrar os *sites* encontrados, eliminando respostas que não são contextualmente relevantes para o usuário.

3 AGENTES INTELIGENTES

Um agente é uma entidade humana ou artificial que está situada em um ambiente, pode percebê-lo e agir sobre ele. O agente tem um objetivo específico e, para atingi-lo, raciocina com base no conhecimento disponível e nas suas percepções sobre o ambiente para tomar a melhor decisão (BARROS, 2002). Segundo (WOOLDRIDGE, 2002), Os agentes têm como características básicas a reatividade, proatividade e sociabilidade. **Reatividade** é a capacidade de interagir com o ambiente, respondendo às suas mudanças que ocorrem no ambiente. A **proatividade** é a capacidade que os agentes possuem de executar ações para atingir um objetivo. Assim, os agentes não agem somente como resposta à eventos ocorridos. E a **sociabilidade** é a capacidade de agir interativamente com outros agentes, de forma cooperada, através de uma linguagem de comunicação.

Algumas outras propriedades atribuídas aos agentes, segundo o autor:

1. Mobilidade: os agentes podem se mover através dos ambientes para executar tarefas específicas;
2. Veracidade: os agentes não transmitem falsas informações intencionalmente.
3. Benevolência: o agente ajuda outros agentes desde que não atrapalhe os seus objetivos;
4. Adaptação e capacidade de aprendizagem: os agentes mudam seu comportamento devido às mudanças do ambiente e com base em sua experiência passada.
5. Racionalidade: os agentes agem da melhor forma para atingir seus objetivos.

3.1 AGENTES REATIVOS E COGNITIVOS

De acordo com o seu comportamento, os agentes podem ser classificados como reativos ou cognitivos. Os agentes reativos têm um comportamento baseado em organizações biológicas, como o das colônias de insetos. Ou seja, simplesmente respondem às mudanças do ambiente ou às mensagens recebidas de outros agentes sem raciocinar sobre suas intenções ou sobre o estado do ambiente antes de agir. Os agentes reativos não podem redefinir seus objetivos, não planejam suas ações e não podem utilizar experiências ou ações anteriores para tomar as decisões futuras. A inteligência provém da interação entre todos os agentes do sistema. Por sua vez, os agentes cognitivos, também chamados deliberativos (RIBEIRO, 2007), têm um comportamento baseado em organizações humanas. Esse tipo de agente mantêm uma representação explícita do ambiente e dos outros agentes da sociedade, podem raciocinar sobre planos e ações que serão executadas para

atingir os objetivos propostos e, se for necessário, podem repensar seus objetivos. Os agentes cognitivos mantêm um histórico das ações passadas, o que lhes permite tomar decisões com base em conhecimento anterior. Assim os agentes cognitivos têm capacidade de aprender com seus erros e acertos. Alguns agentes são desenvolvidos de forma a combinar o comportamento cognitivo e reativo. Esse tipo de agente, chamado agente híbrido, apresenta uma camada reativa que permite uma resposta rápida aos eventos mais importantes e uma camada cognitiva que interpreta e decide quais são as ações que devem ser executadas para que o agente atinja seus objetivos.

Os agentes podem ser classificados também por outros critérios, como as funções que exercem (agentes de pesquisa de informação, agentes de interface, etc), se há a possibilidade do agente movimentar-se através da rede, residindo temporariamente em máquinas diferentes da sua máquina de origem (agentes móveis) ou se o agente tem capacidade para agir de forma cooperada com outros agentes (agentes cooperativos) (REIS,2003).

3.2 ARQUITETURA BDI

Segundo (REIS, 2003), a arquitetura BDI *(Belief-Desire-Intention)* é uma arquitetura de agentes onde o estado interno do agente é descrito a partir de conjunto de crenças *(belief)*, desejos (*desire*) e intenções (*intention*). As **crenças** representam a noção do agente sobre um determinado fato, ou seja, a informação que o agente possui sobre o ambiente a cada instante. As crenças são dinâmicas e podem ser alteradas com o tempo. Os **desejos** do agente referem-se ao que o agente deseja obter. O autor afirma que os objetivos do agente são um subconjunto de desejos atingíveis e consistentes. As **intenções** do agente são um conjunto de tarefas selecionadas para serem realizadas. As intenções devem ser consistentes internamente e representam o resultado do processo de deliberação.

Os agentes BDI estão sempre comprometidos com suas intenções e não obrigatoriamente com seus desejos. O agente mantém um compromisso até que acredite que a intenção foi atingida (compromisso cego), enquanto acreditar que a intenção pode ser atingida ainda (compromisso honesto) ou enquanto a intenção for um objetivo (compromisso aberto). Em sistemas multiagentes o processo de deliberação exige que o agente raciocine não somente sobre suas intenções, mas também sobre as intenções dos outros agentes e sobre as intenções do conjunto de agentes da sociedade. (REIS, 2003) cita a noção de *acordo* entre os

agentes, apresentada por (NORMAN *et al*, 1998). Os acordos são baseados em *direitos* e *ações* a serem realizadas. As ações que o agente pode realizar sem incorrer em penalidades são determinadas pelas ações que é capaz de executar. Os agentes podem ainda delegar ações, ou seja, permitir que outros agentes executem ações que somente eles tem o direito de executar. Os agentes envolvidos num acordo devem estar comprometidos com esse acordo e suas ações devem ser executadas em conformidade com ele.

3.3 AMBIENTE

Os agentes vivem em um determinado ambiente, podendo percebê-lo e agir sobre ele. Os ambientes podem ser físicos, como aqueles nos quais estão inseridos os robôs, ou ambientes de *software* (também chamados ambientes de realidade virtual), onde se faz simulação do ambiente físico. As características do ambiente determinam a arquitetura do agente e na sua forma de operação (REIS, 2003).

(WOOLDRIDGE, 2002) apresenta a seguinte classificação dos ambientes, baseada no trabalho de (RUSSEL *et al*, 1995):

Acessível versus Inacessível: ambientes acessíveis permitem que o agente obtenha, através dos seus sensores, informações atualizadas, precisas e completas sobre o ambiente. Os ambientes físicos reais e a *Internet* são exemplos de ambientes inacessíveis.

Determinístico versus Não-Determinístico: em ambientes determinísticos cada ação do agente tem um efeito único e garantido. O agente, portanto, tem certeza sobre o estado resultando de qualquer ação.

Estático versus Dinâmico: um ambiente estático permanece inalterado enquanto o agente decide qual será a próxima ação a executar. Em ambientes dinâmicos outros agentes agindo no mesmo ambiente podem alterá-lo sem que o agente tenha controle sobre as alterações. O mundo real é um ambiente extremamente dinâmico, bem como a *Internet.*

Discreto versus Contínuo: em um ambiente discreto somente é permitido um número finito de percepções e ações possíveis para o agente. Caso contrário o ambiente é chamado contínuo. Ambientes podem ser contínuos quanto às percepções do agente e discretos no que diz respeito às suas ações ou vice-versa. (REIS, 2003).

Desenvolver agentes para operarem em ambientes acessíveis é mais simples do que agentes que operam em ambiente inacessíveis. Como as decisões tomadas pelos agentes estão

ligadas à qualidade e à quantidade de informações que recebem do ambiente, é mais fácil para o agente tomar decisões corretas em ambientes acessíveis.

Ambientes não-determinísticos impõem restrições à capacidade do agente de prever o estado futuro do mundo e de tomar decisões. Para viver em ambientes não-determinísticos, o agente deve incorporar capacidades de recuperação e tolerância a falhas, já que as ações do agente podem não ter um efeito único e previsível ou podem falhar. Da mesma forma ambientes dinâmicos também acrescentam complexidade ao agente. Em ambientes dinâmicos uma ação poderá ter um efeito totalmente diferente dependendo do instante no qual é executada já que o ambiente pode ser alterado sem a intervenção do agente. Nesse tipo de ambiente muitos agentes coexistem e acabam por interferir, de forma positiva ou não, nas ações efetuadas por outros agentes. Isso faz com que, em ambientes dinâmicos, os agentes tenham que coordenar suas ações, comunicarem-se, cooperarem mutuamente, negociarem e competirem entre si para executar suas tarefas. Por fim, como os sistemas computacionais são discretos por natureza, ambientes discretos são mais simples para os agentes. Pode-se concluir, então, que os ambientes que apresentam maior complexidade para o desenvolvimento de agentes são os ambientes inacessíveis, não-determinísticos, dinâmicos e contínuos, como os ambientes do mundo real. Estes ambientes são chamados de ambientes abertos (WOOLDRIDGE, 2002).

3.4 SISTEMAS MULTIAGENTES

Sistemas multiagentes (SMA) são sistemas compostos por múltiplos agentes voltados à resolução de um determinado problema. Os agentes em um sistema multiagentes são autônomos, ou seja, sua existência não depende de intervenção externa, e podem ter graus diferentes de capacidade para resolver problemas. Os agentes interagem uns com os outros através de protocolos de interação social. Num sistema multiagentes os agentes trabalham juntos com o intuito de resolver problemas que não podem ser resolvidos por um agente sozinho. Esses agentes compartilham suas crenças e possuem intenções conjuntas quando atuam juntos para atingir o mesmo objetivo. O relacionamento entre os agentes é feito através de mecanismos de interação.

Para resolução dos problemas, os agentes planejam suas atividades, dinâmica ou estaticamente. O planejamento das atividades pode ser feito de forma centralizada, quando um

agente elabora o plano e aloca as tarefas aos agentes executores, ou distribuída, quando cada agente elabora seus plano individual e troca informações com os outros agentes visando a identificação e resolução dos conflitos. Em planejamentos distribuídos os agentes enfrentam problemas para formular seus planos individuais pois precisam descobrir e levar em contas as ações que os outros agentes pretendem executar.

3.4.1 COMUNICAÇÃO ENTRE AGENTES

A comunicação é a troca de informações que permite que haja coordenação das ações entre os agentes. Para que a comunicação seja estabelecida é preciso que todos os agentes participantes conheçam as regras de envio e recebimento de mensagens, a linguagem utilizada e a forma como as mensagens são estruturadas. Ao ler uma mensagem recebida de outro agente, com base em seu próprio conhecimento e nas informações nela contidas, o agente determina as ações que irá executar. O envio das mensagens pode ser feito diretamente para um agente específico, através de seu endereço, para um conjunto de agentes ou para todos os agentes presentes no ambiente.

3.4.2 NEGOCIAÇÃO

Negociação é a troca de mensagens com objetivo de estabelecer acordos sobre as formas de cooperação entre os agentes. O objetivo da negociação é estabelecer uma concordância sobre quais são as tarefas específicas que devem ser desenvolvidas por cada agente. O processo de negociação é utilizado para resolução de diversos problemas, como distribuição de tarefas, a comunicação e a cooperação entre os agentes e na manutenção da coerência do sistema.

Do processo de negociação podem surgir três situações**:**

Conflito: quando nenhum acordo é possível;

Compromisso: quando os agentes preferem trabalhar de maneira isolada, mas negociam um acordo quando isto não for possível;

Cooperativo: quando os agentes participantes preferem os acordos negociados. agentes.

Um dos processos de negociação mais utilizadas são as *redes contratuais* (SMITH, 1980). Neste processo os agentes coordenam suas ações através de contratos estabelecidos entre um agente *gerente* (que distribui as tarefas) e agentes *empreiteiros* (que executam as tarefas).

3.4.3 COORDENAÇÃO

Em um sistema multiagentes, os agentes devem trabalhar de forma coordenada para garantir que a comunidade funcione como um unidade (LOURENCO *et al,* 2006). A coordenação entre os agentes é importante porque nenhum dos agentes possui todos os recursos, as competências ou as informações que serão necessárias para resolver os problemas de forma independente. Assim, um agente pode necessitar que outro agente termine a execução de uma tarefa para poder realizar uma ação e algumas tarefas deverão ser executadas por mais de um agente ao mesmo tempo. Coordenar as ações dos agentes é uma forma de melhorar a eficiência do sistema e de evitar a anarquia e o caos. Podemos definir coordenação como sendo a capacidade que os agentes possuem de executar algumas atividades de forma compartilhada, gerenciando os recursos utilizados por outros agentes e os conflitos que podem existir.

Normalmente, para haver coordenação entre os agentes é necessário que haja comunicação. Porém a coordenação também pode ser conseguida sem a comunicação, como nos casos em que os agentes conhecem o modelo de comportamento do restante dos agentes. Outra forma de coordenação consiste em definir previamente a estrutura organizacional e a hierarquia entre os agentes.

3.4.4 COOPERAÇÃO

Ao receber um problema, os agentes o decompõem em subproblemas com os quais possa lidar, seja a partir dos seus recursos e informações individuais, seja buscando a cooperação de outros agentes. A cooperação é a habilidade que o agente possui de interagir com outros agentes para atingir um objetivo. Os agentes agem de forma cooperada quando necessitam de auxílio para solução de um problema. Para que aja cooperação entre os agentes,

é necessário que aja também a coordenação das atividades e dos objetivos comuns.

3.5 AGENTES DE BUSCA NA *INTERNET*

Como a *Internet* é composta por milhares de páginas e também é altamente dinâmica, os sistemas multiagentes estão sendo largamente empregados neste ambiente. Muitas aplicações se utilizam das vantagens dos agentes para lidar com as complexidades da rede. (IVASSAKI, 2006), por exemplo, propõe o uso de agentes para e divulgação de conteúdo personalizado em jornais *on-line*. Na aplicação proposta, os agentes coletam informações sobre o perfil do leitor e, sempre que este acessar o jornal, os agentes criarão uma interface personalizada de acordo com as características coletadas pelo agente. Outra aplicação propõe a organização de informações específicas sobre o agronegócio do café (CISCON, 2004). O sistema funciona em três etapas: coleta de informações relevantes para o agronegócio do café, armazenamento das informações coletadas e disponibilização das informações. Os agentes monitoram os *site*s pré-selecionados especializados no agronegócio para extrair deles as informações necessárias e armazená-las no sistema de banco de dados. Por fim um outro agente monta uma página a partir das informações armazenadas nos banco de dados para que essas informações sejam disponibilizadas para consulta.

As aplicações de busca e recuperação de informações também se utilizam de agentes. Muitos mecanismos usam sistemas multiagentes para realizar a varredura da *Internet* de forma distribuída. O objetivo é indexar um número maior de páginas em um tempo menor. Outros mecanismos, chamados metabuscadores, utilizam os agentes para permitir que uma busca seja realizada simultaneamente em diversos mecanismos de busca. Ou seja, ao receber uma solicitação de busca de um usuário, o metabuscador irá, através de agentes, realizar essa mesma requisição em diversos outros mecanismos de busca. Dessa forma um número maior de páginas é incluído na consulta, já que o metabuscador não se restringe à estrutura de índices de um único mecanismo.

A ferramenta *Copernic Agent* (*COPERNIC*, 2009) é um exemplo de metabuscador. O *Copernic* é uma ferramenta comercial desenvolvida pela empresa *Copernic* Inc que funciona da seguinte forma: quando um usuário faz uma requisição de busca para o *Copernic*, a ferramenta dispara agentes que farão a mesma requisição para vários mecanismos de busca diferentes. O *Copernic* reúne os resultados apresentados por todos os mecanismos visitados,

remove as entradas e apresenta os resultados restantes para o usuário. A facilidade de uso e as muitas versões que a ferramenta apresenta, como uma versão específica de buscador para documentos armazenados em um computador ou rede local, fazem do *Copernic* uma ferramenta largamente utilizada atualmente.

4 ONTOLOGIAS

Para que sistemas de computador possam trocar informações e trabalhar cooperativamente na resolução de problemas é necessário que haja um entendimento comum e não ambíguo dos termos e conceitos utilizados. As ontologias foram criadas para permitir o entendimento entre os sistemas de computador a partir da descrição formal dos termos, dos conceitos e das relações entre os conceitos utilizados em um domínio. As ontologias incluem também um vocabulário dos termos empregados nesse domínio e um conjunto de regras para a interpretação desse vocabulário.

Na literatura encontramos muitas definições de Ontologia. Porém, a mais citada foi fornecida por Thomas Gruber (GRUBER, 1993): "Uma ontologia é uma especificação explícita de uma conceitualização compartilhada". Para (GUIMARAES, 2002), essa definição mostra algumas características de uma ontologia: ela deve ser explícita (os conceitos, termos e relações devem ser definidos de forma clara e explícita), formal (deve ser escrita utilizando linguagem formal, permitindo que seja lida e compreendida pelos sistemas de computadores) e compartilhada (deve descrever conceitos aceitos por um grupo de pessoas).

O uso de ontologias é importante pois fornece um vocabulário que permite a descrição e a representação exata de um conhecimento. Esse vocabulário, por ser expresso em linguagem formal, não permite termos ambíguos. Assim, o uso de ontologias permite o compartilhamento do conhecimento sem que hajam interpretações diferentes para um mesmo termo. As ontologias podem ser estendidas, o que permite que uma ontologia genérica seja readequada de forma a expressar um domínio específico de conhecimento, e podem ser mapeadas, permitindo que uma mesma ontologia seja expressa em várias línguas sem alterações de conceitualização.

Basicamente as ontologias são compostas por:

a) Um conjunto de conceitos hierárquicos (taxonomias);
b) Um conjunto de relacionamentos e de funções que podem existir entre os conceitos. Uma função é um caso especial de relacionamento no qual há uma relação única entre os elementos;
c) Um conjunto de regras que sempre serão válidas no contexto da ontologia (axiomas);
d) Um conjunto conhecimentos prévios inerentes ao domínio do conhecimento (instâncias).

4.1 RDF

Para permitir a representação das ontologias é necessário dispor de uma linguagem que

permita a descrição do conhecimento de maneira clara e sem ambiguidades. O primeiro padrão proposto com esse fim foi o RDF (*Resource Description Framework*) (W3C, 2009)

O RDF é um padrão muito simples, baseado em XML que permite que sejam descritas sentenças sobre objetos. Os conceitos fundamentais do RDF são os conceitos de **recurso**, **propriedade** e **sentença** (RIBEIRO *et al*, 2007). Um recurso é qualquer coisa sobre a qual se possa falar, como objetos, lugares ou pessoas. Propriedades são tipos especiais de recursos que descrevem as relações entre os recursos. Tanto os recursos quanto as propriedades são identificados através de uma URI (*Uniform Resource Identifier*). As sentenças descrevem as propriedades dos recursos.

Uma sentença RDF é expressa através de triplas do tipo *(sujeito, predicado, objeto)*, onde o sujeito é um recurso, o predicado é uma propriedade e o objeto pode ser outro recurso ou um valor literal. As sentenças em RDF mostram que um sujeito está relacionado por um predicado a um objeto (EUBANKS, 2005). A Figura 3 mostra a representação gráfica em notação RDF da sentença "*Júpiter é um planeta*"

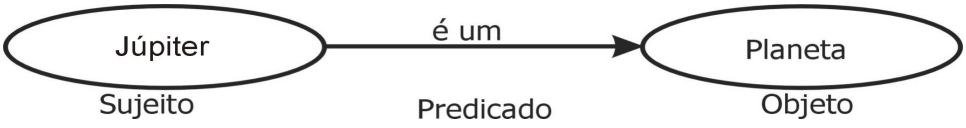

Figura 3: Representação gráfica de uma sentença RDF

Na verdade como todos os recursos e propriedades em RDF são identificado através de uma URI, as sentenças representam como a URI do sujeito se relaciona através da URI do predicado com a URI do objeto. Assim, na sentença acima o objeto *planeta* pode ser representado pela URI *http://www.w3.org/TCC#planeta*, o predicado *é um* pode ser representado pela URI *http://www.w3.org/TCC#e_um* e o sujeito pela URI *http://www.w3.org/TCC#Jupiter.* Em RDF a sentença será escrita da seguinte forma:

```
<http://www.w3.org/TCC#Jupiter>
      <http://www.w3.org/TCC#e_um>
            <http://www.w3.org/TCC#planeta>
```

O RDF não estabelece um vocabulário padrão com termos específicos para descrever um domínio mas oferece mecanismos que podem ser usados para descrever o domínio. Para descrever a semântica do domínio utiliza-se a extensão RDF *Schema* (RDFS). Segundo

(LEMKE, 2007), o RDFS representa as classes e as propriedades utilizadas na definição de um novo vocabulário. Assim como na orientação a objetos, uma classe em RDFS representa um conjunto de objetos com características comuns. Dessa forma, o RDFS define as classes e o RDF as instâncias dessas classes (RIBEIRO, 2007).

4.2 OWL

O OWL (*Ontology Web Language*) é uma linguagem de marcação baseada em XML e RDF que permite a criação e o compartilhamento de ontologias. A OWL adiciona vocabulário aos documentos RDF que permite descrever propriedades, classes e as relações entre classes com maior precisão. A linguagem OWL se subdivide em 3 sub-linguagens OWL *Lite*, OWL DL e OWL *Full*.

A OWL *Lite* é uma simplificação da linguagem OWL que permite somente a implementação e definição de classes e propriedades. A OWL DL (OWL *Description Logic*) estende a OWL Lite e inclui algumas funcionalidades, como permitir restrições de cardinalidade que não são limitadas em 0 ou 1. A OWL *Full* é a versão completa da OWL. Cada sub-linguagem é uma extensão da linguagem precedente. Ou seja, expressões escritas em OWL *Lite* são expressões válidas em OWL DL e também válidas em OWL *Full*.

Os documentos OWL são compostos por **indivíduos**, **classes** e **propriedades**. Um indivíduo é um objeto específico de um domínio. Uma classe representa um conjunto de objetos com características comuns e as propriedades representam as relações binárias entre os indivíduos. Em OWL existem dois tipos de propriedades: propriedades de objetos, que relacionam um indivíduo com outro indivíduo e propriedades de tipo de dados, que relacionam um indivíduo a um literal ou a um valor XML *Schema* (como uma *string*, um *integer*, etc) (LEMKE, 2007).

Abaixo está um fragmento de ontologia escrita em linguagem OWL. Este exemplo foi criado a partir da ontologia desenvolvida pelo projeto *Sweet* (SWEET, 2009).

```
<owl:Class rdf:about="#Planet">
      <rdfs:subClassOf rdf:resource="#SolarSystemPart"/>
</owl:Class>
<owl:Class rdf:about="#Earth">
      <rdfs:subClassOf rdf:resource="#Planet"/>
```

```
</owl:Class>
<owl:Class rdf:about="#Jupiter">
      <rdfs:subClassOf rdf:resource="#Planet"/>
</owl:Class>
<owl:Class rdf:about="#Satellite">
      <rdfs:subClassOf rdf:resource="#SolarSystemPart"/>
</owl:Class>
<owl:Class rdf:about="#Moon">
      <rdfs:subClassOf rdf:resource="#Satellite"/>
</owl:Class>
```

Nesta ontologia estão representados alguns relacionamentos do tipo *subClassOf* entre objetos do sistema solar. Por exemplo, a classe *Moon* é representada como uma subclasse da classe *Satellite.* Ou seja, *Moon* é um tipo de (*is a*) *Satellite.* A Figura 4 apresenta uma representação gráfica dos relacionamentos descritos nesta ontologia.

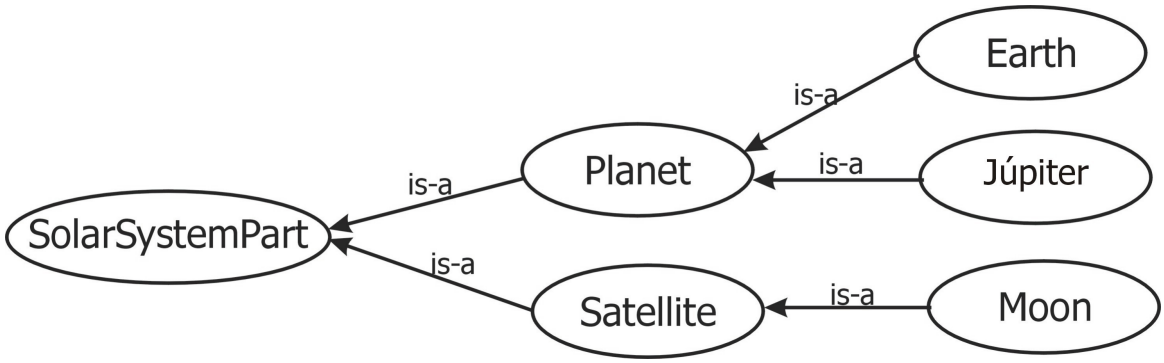

Figura 4: Representação gráfica da ontologia OWL

4.3 ONTOLOGIAS NA *WEB* SEMÂNTICA

O objetivo principal da *Web* semântica é explicitar o relacionamento entre os conceitos de diversos domínios de conhecimento e permitir que os agentes de *software* possam trabalhar sobre esses relacionamentos, compreendê-los e a partir deles inferir novos conceitos. A *Web* semântica utiliza um conjunto de padrões, tecnologias e ferramentas que constroem uma infraestrutura básica para permitir a inclusão de significado nas páginas da *Web*. A arquitetura da *Web* semântica é normalmente representada como um conjunto de blocos que representam

os inter-relacionamentos entre os padrões, conforme mostrado na Figura 3.

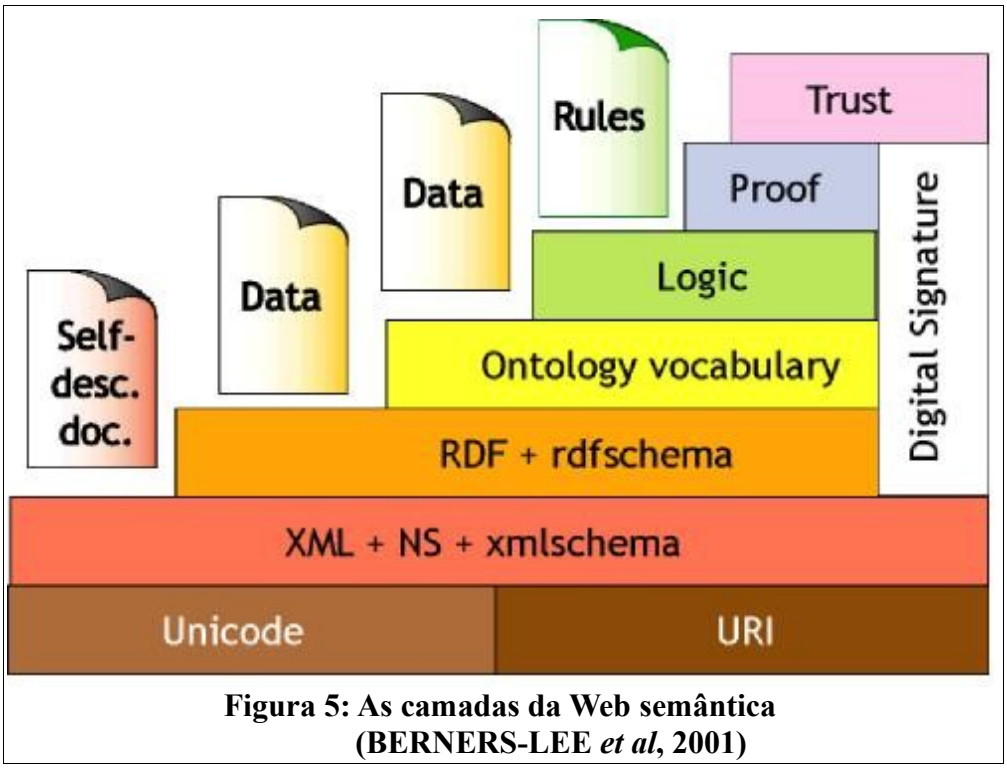

Figura 5: As camadas da Web semântica (BERNERS-LEE *et al*, 2001)

A camada de ontologia é um dos pilares da *Web* semântica (W3C, 2009) pois fornece um mecanismo eficiente para representação e compartilhamento dos conhecimentos. Por isso, muitos grupos têm desenvolvido ontologias sobre diversas áreas do conhecimento. Na página do projeto SWEET (NASA,2009), por exemplo, encontram-se ontologias que descrevem o contexto das ciências da Terra e do meio ambiente. Muitos projetos utilizam e estendem essas ontologias, como o projeto DOLCE (DOLCE, 2006) e o projeto SPIRE (SPIRE, 2009). Para facilitar o compartilhamento de ontologias e outros recursos *Web* semânticos, foi criado um mecanismo para busca de documentos semânticos na *Web* chamado SWOOGLE (SWOOGLE, 2007). O SWOOGLE analisa e indexa os documentos permitindo que sejam facilmente encontrados.

Uma importante ferramenta para manipulação de ontologias é o Protégé (PROTÉGÉ, 2009). O Protégé é um ambiente para criação, visualização, manipulação e edição de ontologias em diversos formatos de representação que também permite a criação de modelos de conhecimento. Desenvolvido em Java, o Protégé foi criado pela Faculdade de Medicina de Stanford.

Outra ferramenta para manipulação de ontologias, o *framework* JENA (JENA, 2009) foi desenvolvido pelo núcleo de pesquisa em *Web* semântica da *Hawlett-Packard.* O *framework* permite a construção de aplicações para a *Web* semântica, fornecendo um ambiente de programação em RDF, RDFS e OWL juntamente com um motor de inferência baseado em regras. Utilizando o JENA é possível criar ferramentas para ler, manipular e persistir ontologias e também criar ou estender mecanismos de inferência existentes.

4.4 CONSIDERAÇÕES

Uma ontologia relaciona os conceitos que fazem parte de um contexto. Por isso as ontologias podem ser utilizadas como forma de filtrar a busca realizada pelos mecanismos de busca. Por exemplo, se uma busca pelo termo *Jupiter* no contexto de astronomia for solicitada, através da ontologia associada ao contexto o mecanismo saberá que o termo *planet* está relacionado ao termo *Jupiter.* Assim, o mecanismo poderá filtrar as páginas onde o termo pesquisado *Jupiter* apareça juntamente com o termo *planet.* Dessa forma os resultados apresentados serão mais precisos, ou seja, mostrarão páginas com conteúdos mais relevantes dentro do contexto do que se a busca fosse realizada somente pelo termo *Jupiter.*

5 AGENTE DE BUSCA CONTEXTUAL

O *Web Searcher Agent* é um sistema multiagentes que tem por objetivo realizar uma busca na *Web* utilizando regras semânticas para filtragem dos resultados. Para o desenvolvimento do *Web Searcher* foram utilizados o *framework* SemantiCore e a ferramenta Jena. O SemantiCore facilita o desenvolvimento de sistemas multiagentes para a *Web* semântica, por isso permite que os agentes manipulem ontologias com facilidade. O Jena permite leitura e a criação de ontologias e é utilizado pelo SemantiCore para prover as facilidades de manipulação de ontologias pelos agentes.

5.1 O *FRAMEWORK* SEMANTICORE

O SemantiCore é um *framework* que permite o desenvolvimento de sistemas multiagentes. Segundo (RIBEIRO *et al*, 2006), o SemantiCore é dividido em dois modelos: o modelo de domínio semântico, responsável pela definição da composição do domínio e suas entidades e o modelo de agente, que determina a estrutura interna do agente.

No SemantiCore os ambientes são denominados **Domínios Semânticos**. Cada domínio semântico possui um *Controlador de Domínio* e um *Gerente de Ambiente*. O Controlador de Domínio é o responsável pelo registro dos agentes no ambiente, pela recepção aos agentes móveis e pelas características de segurança. O Gerente de Ambiente faz a ponte entre o domínio semântico do SemantiCore e os domínios *Web* tradicionais.

O modelo de agente do SemantiCore possui uma estrutura orientada a componentes na qual cada componente representa uma parte do funcionamento do agente. Os quatro componentes básicos do agente são o componente **sensorial**, o componente **decisório**, o componente **executor** e o componente **efetuador.**

O componente sensorial é responsável pela captura dos recursos que trafegam pelo ambiente. Cada sensor presente neste componente captura um tipo diferente de objeto. Os sensores são verificados sempre que um objeto semântico é percebido pelo ambiente. O componente sensorial, ao capturar um objeto através de um sensor, encaminha o objeto para que seja analisado pelos outros componentes.

O componente decisório é responsável pela tomada de decisões do agente. O SemantiCore é voltado para o desenvolvimento de aplicações para a *Web* semântica, por isso o componente decisório opera sobre ontologias. A saída gerada pelo componente decisório é uma **ação**. As ações mapeiam os comandos necessários para que o agente trabalhe de forma apropriada e podem ser

aplicadas tanto aos elementos do agente quanto aos elementos do domínio semântico.

O mecanismo executor contém os planos de ação que serão executados pelo agente e o mecanismo efetuador é responsável pelo encapsulamento das mensagens transmitas no ambiente. Semelhantemente ao que acontece com o sensor, o mecanismo efetuador possui uma série de efetuadores. Cada um deles é responsável por publicar um tipo de objeto no ambiente.

5.1.1 AGENTES NO SEMANTICORE

Conforme (RIBEIRO *et al*, 2007) os agentes do SemantiCore estendem a classe *SemanticAgent*. A execução do agente inicia com a execução do método *setup*, onde são criados os sensores, os efetuadores e os planos de ação e também são definidas as regras, os fatos e os objetivos do agente. Esse método é chamado uma única vez durante a criação do agente. Depois de criado, o agente executa basicamente quatro métodos em *loop:* o agente sente o ambiente, decide que ação será executada, executa essa ação e retorna os resultados obtidos para o ambiente, conforme ilustrado pela Figura 6.

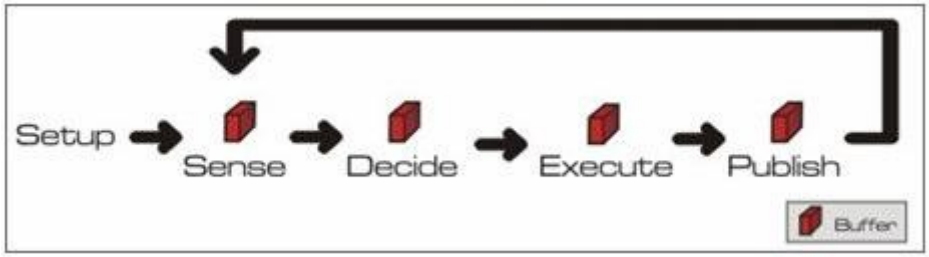

Figura 6: O ciclo de vida do agente SemantiCore (RIBEIRO *et al*, 2007)

As quatro operações básicas do ciclo de vida do agente estão associadas a um dos quatro componentes básicos do agente: sensorial (*sensorial*), decisor (*decision*), executor (*executor*) e efetuador (*effector*).

O agente pode sentir o ambiente através dos seus sensores. Os sensores devem ser criados estendendo a classe *Sensor* do SemantiCore. Um agente pode ter diversos sensores trabalhando simultaneamente, um para cada tipo de mensagem que deve receber. Quando uma mensagem for recebida pelo sensor, ela é repassada para o componente decisor. O componente decisor manipula as regras e os fatos e administra o mecanismo de tomada de

decisões do agente. Este componente, que deve estender a classe *GenericDecisionEngine*, pode se utilizar de métodos diferentes para tomar suas decisões, como o uso de mecanismos de inferência, redes neurais ou árvores de decisão. Com base nas decisões tomadas pelo mecanismo decisor, uma ou outra ação deve ser executada. O mecanismo executor é responsável pela execução das ações do agente. As ações são desenvolvidas estendendo a classe *Action*. Durante a execução de uma ação pode ser que o agente necessite enviar mensagens para outros agentes no ambiente. O encapsulamento e envio dessas mensagens é responsabilidade do componente efetuador. Quando um agente é criado, podem ser criados diversos efetuadores que serão responsáveis pelo envio de tipos diferentes de mensagens. Assim um agente pode conversar simultaneamente com diferentes outros agentes através de tipos diferentes de mensagens (SOAP, ACL ou OWL).

A Figura 7 apresenta o código de criação do agente Gerente.

```
1. Public class Gerente extends SemanticAgent
2. {
3.     Public Gerente(Environment env,
4.       String agentName, String arg)
5.     {
6.       super(env, agentName, arg);
7.     }
8.
9.     protected void setup()
10.    {
11.      this.addSensor(new SensorGerente("SensorGerente"));
12.
13.      this.setDecisionEngine(new DecisorGerente());
14.
15.      ActionPlan novaAcaoGer = addActionPlan("acaoGerente");
16.      novaAcaoGer.addAction(new AcaoGerente(environment));
17.
18.      ActionPlan acaoRespGer = addActionPlan("acaoRespGerente");
19.      acaoRespGer.addAction(new AcaoRespGerente(environment));
20.    }
21. }
```

Figura 7: Agente Gerente

O método *setup*() é o responsável pela inicialização dos componentes do agente. Através do método *add.Sensor()* (linha 11), os sensores são habilitados para serem utilizados pelo agente. O método *setDecisionEngine*() (linha 13) determina qual será o mecanismo decisório utilizado pelo agente e o método *addActionPlan()* (linhas 15 e 18) cria os planos de

ação que poderão ser executadas pelo agente.

O SemantiCore possui uma integração nativa com o *framework* Jena. Assim, é possível que os agentes interajam através da manipulação de ontologias. No componente decisório do agente, por exemplo, o *framework* Jena pode ser utilizado como um mecanismo de inferência para auxiliar no processo de tomada de decisões. As ontologias podem também ser utilizadas para representar fatos e regras que deverão ser levados em conta pelo agente. Os próprios agentes do SemantiCore podem ser representados através de uma ontologia em formato OWL. O uso de ontologias torna mais simples a tarefa de recriar o agente, bastando para isso que a ontologia associada a ele seja instanciada. No caso de agentes móveis, por exemplo, para permitir que esses agentes trafeguem por diversos ambientes basta enviar a ontologia pela rede. No destino, basta instanciar a ontologia para que o agente seja recriado com todos os elementos necessários para continuar sua execução corretamente.

5.2 ARQUITETURA DO *WEB SEARCHER AGENT*

O *Web Searcher* utiliza dois tipos diferentes de agentes: o agente **Gerente** e o agente **Buscador.** O agente gerente é o responsável pelo gerenciamento da busca e pela criação dos agentes buscadores. Os agentes buscadores executam a busca das informações solicitadas pelo gerente. A Figura 8 mostra o diagrama das classes utilizadas para implementação do agente.

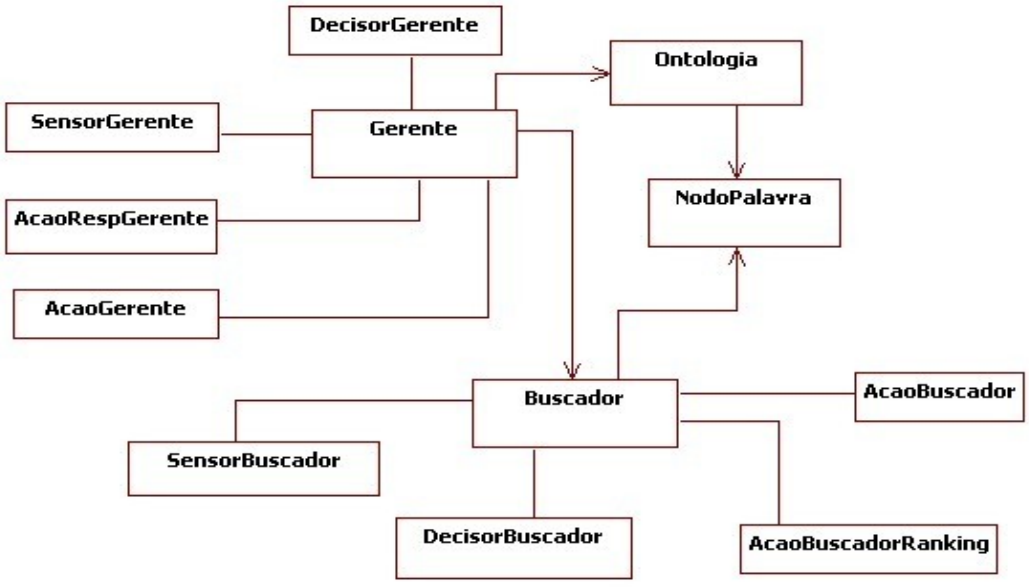

Figura 8: Diagrama de Classes

5.2.1 O AGENTE GERENTE

O Gerente é o agente responsável pelo gerenciamento dos agentes buscadores. É o Gerente quem recebe a requisição de busca, juntamente com o contexto em que a busca deverá ser realizada. Para cada contexto de busca está associada uma ontologia, que apresenta termos relevantes dentro do contexto onde a busca será realizada. O Gerente lê essa ontologia, busca os termos associados à ela e determina o peso que cada termo terá na busca. Para cada termo encontrado, o Gerente cria um agente Buscador. O Gerente também é responsável pelo controle da pontuação das páginas, somando as pontuações encontradas por cada Buscador. Por fim, quando os Buscadores finalizarem seu trabalho, o Gerente ordena as páginas encontradas e apresenta os resultados para o usuário.

O funcionamento do Gerente pode ser descrito pelo algoritmo abaixo.

1. Recebe requisição de busca;
2. Lê a ontologia associada ao contexto da busca a ser realizada;
3. Para cada conceito existente na ontologia, cria um agente de busca;
4. Enquanto houverem páginas a ser avaliadas;
   4.1. Recebe a página encontrada pelo agente de busca com sua respectiva pontuação;
   4.2. Repassa a página para ser avaliada pelos demais agentes no ambiente;
   4.3. Soma as pontuações que os agentes atribuíram para a página;
5. Ordena a lista de páginas pela pontuação recebida;
6. Apresenta os resultados.

5.2.2 O AGENTE BUSCADOR

Os Buscadores são responsáveis pela varredura das páginas da *Web* em busca dos termos apresentados pela ontologia. Quando é criado, o Buscador recebe do Gerente uma palavra da ontologia, juntamente com o peso atribuído a essa palavra, e uma lista de páginas a serem visitadas, chamada de semente. O peso da palavra representa a importância dessa palavra no contexto de busca. O Buscador realiza basicamente duas funções: a busca pelas

páginas e a valoração da página. A ação de busca das páginas é iniciada no momento em que o Buscador é criado. O algoritmo abaixo ilustra o funcionamento da ação de busca.

1. Recebe do Gerente a semente, a palavra que deve ser buscada e o peso atribuído a ela.
2. Para cada página:
    2.1. Dispara o *spider;*
    2.2. Determina a pontuação da página ;
    2.3. Envia a página e a pontuação encontrada para o Gerente;
3. Finaliza a execução das tarefas.

O Buscador envia para o Gerente as páginas que encontra com sua respectiva pontuação. A pontuação da página é calculada pela fórmula *k*p*, onde *k* representa a quantidade de vezes que a palavra aparece na página e *p* representa o peso da palavra. Cada Buscador procura na página somente a palavra que recebeu no momento de sua criação. Para que as páginas sejam corretamente filtradas, o Gerente envia cada página recebida para que os demais Buscadores pontuem a página de acordo com suas palavras. Assim, o Buscador é responsável também pela análise das página lidas pelos outros Buscadores e pelo cálculo da pontuação dessas páginas com base em sua palavra específica. A ação de valoração da página é conceitualmente simples, conforme o algoritmo abaixo:

1. Recebe a página a ser analisada;
2. Determina a pontuação da página;
3. Envia a página encontrada para o Gerente.

5.2.3 O *SPIDER*

O *spider* é responsável pela busca e pela análise das páginas da *Web*. O *spider* lê a página e procura pelos *links* que ela possui. Esses *links* são adicionados à lista de páginas a serem pesquisadas pelo Buscador. O *spider* também é responsável pela análise que vai determinar a pontuação da página. O *spider* funciona de forma semelhante aos *spiders* utilizados pelos mecanismos de busca tradicionais, conforme pode ser visto no algoritmo abaixo.

1. Recebe uma página a ser acessada;
2. Copia a página;
3. Procura pelos *links* existentes na página;

4. Calcula a pontuação da página;
5. Retorna os dados encontrados.

Para calcular a pontuação da página, o *spider* procura dentro dela por todas as ocorrências da palavra recebida pelo Buscador. Para cada ocorrência da palavra, o peso correspondente é somado à pontuação já calculada para aquela página, conforme o algoritmo abaixo:

1. A página inicia com pontuação zero;
2. Enquanto encontrar a palavra na página;
   2.1. Soma o peso da palavra à pontuação da página;
3. Retorna pontuação calculada pela página.

5.2.4 O FILTRO SEMÂNTICO

O filtro semântico representa a parte principal do processo de busca. O filtro é realizado em dois passos

1. Leitura da ontologia associada ao contexto;
2. Cálculo da pontuação de cada página.

A leitura da ontologia é feita pelo agente Gerente. O Gerente utiliza a API do Jena para percorrer a ontologia e procurar pelas palavras solicitadas pelo usuário. O Jena fornece classes específicas para leitura das ontologias em formato OWL, como pode ser visto no código de exemplo apresentado na Figura 9 abaixo.

```
Public class Ontologia
{
   public TreeMap leOntologia(String URI, String local)
   {
      OntModel  onto;
      // cria uma ontologia no modelo OWL
      onto = ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM);
      onto.getDocumentManager().addAltEntry(URI, local);
      onto.read(this.local);

      // Pega a lista de classes existentes na ontologia
      ExtendedIterator classe =(ExtendedIterator) onto.listClasses();

      // Cria uma TreeMap onde serão especificadas
      //as classes da ontologia.
      TreeMap<OntClass, String> tm = new TreeMap<OntClass, String>()

      // enquanto houver outra classe
      while(classe.hasNext())
      {
         OntClass aux = (OntClass) classe.next();
         // se houver uma superclasse
         if(aux.hasSuperClass())
            // adiciona a superclasse
            tm.put(aux.getSuperClass(), aux.getLocalName());
         else
            tm.put(null, aux.getLocalName());
      }

      return tm;
}
```

Figura 9: Leitura de uma ontologia usando a API Jena

Cada palavra encontrada na ontologia receberá um peso que determina a importância dessa palavra no contexto da busca. Por exemplo, seja a ontologia que está representada na Figura abaixo, representando o contexto da astronomia:

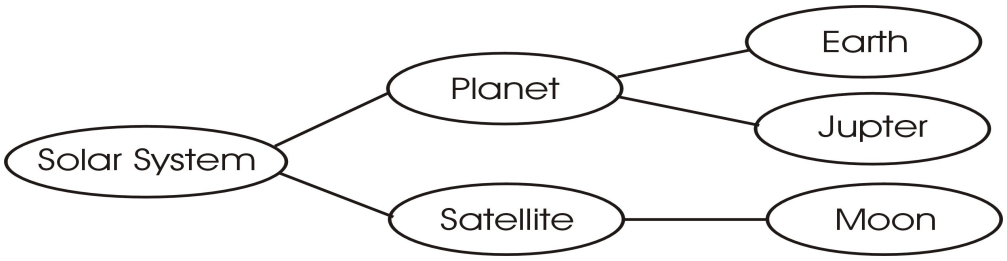

Figura 10: Exemplo de ontologia

Supondo que seja solicitada a busca pelas palavras "*Earth*" e "*Solar System*". O

Gerente fará a leitura da ontologia e definirá um peso para cada uma das palavras que constam na ontologia. As palavras diretamente relacionadas com aquelas solicitadas pela busca ("*Solar System", "Planet"* e *"Earth*") terão peso 10. As demais palavras constantes na Ontologia ("*Satellite", "Jupiter"* e "*Moon*") terão peso 5. O Gerente criará um Buscador para cada uma das palavras retornadas pela ontologia e cada buscador será responsável por visitar uma lista de *sites* inicial.

Cada página visitada pelo Buscador receberá uma pontuação de acordo com a quantidade de vezes que a palavra aparece nessa página. Por exemplo, se o Buscador responsável pela busca da palavra "*Earth",* cujo peso é 10, encontrá-la 10 vezes em uma página, a pontuação referente à página será igual a 100. Essa página passará pela avaliação dos outros Buscadores. O Gerente somará todas as pontuações atribuídas por cada buscador e, no final, irá organizar as páginas de acordo com a pontuação conseguida por cada página. O diagrama de sequências apresentado na Figura 11, mostra o fluxo de mensagens entre os agentes.

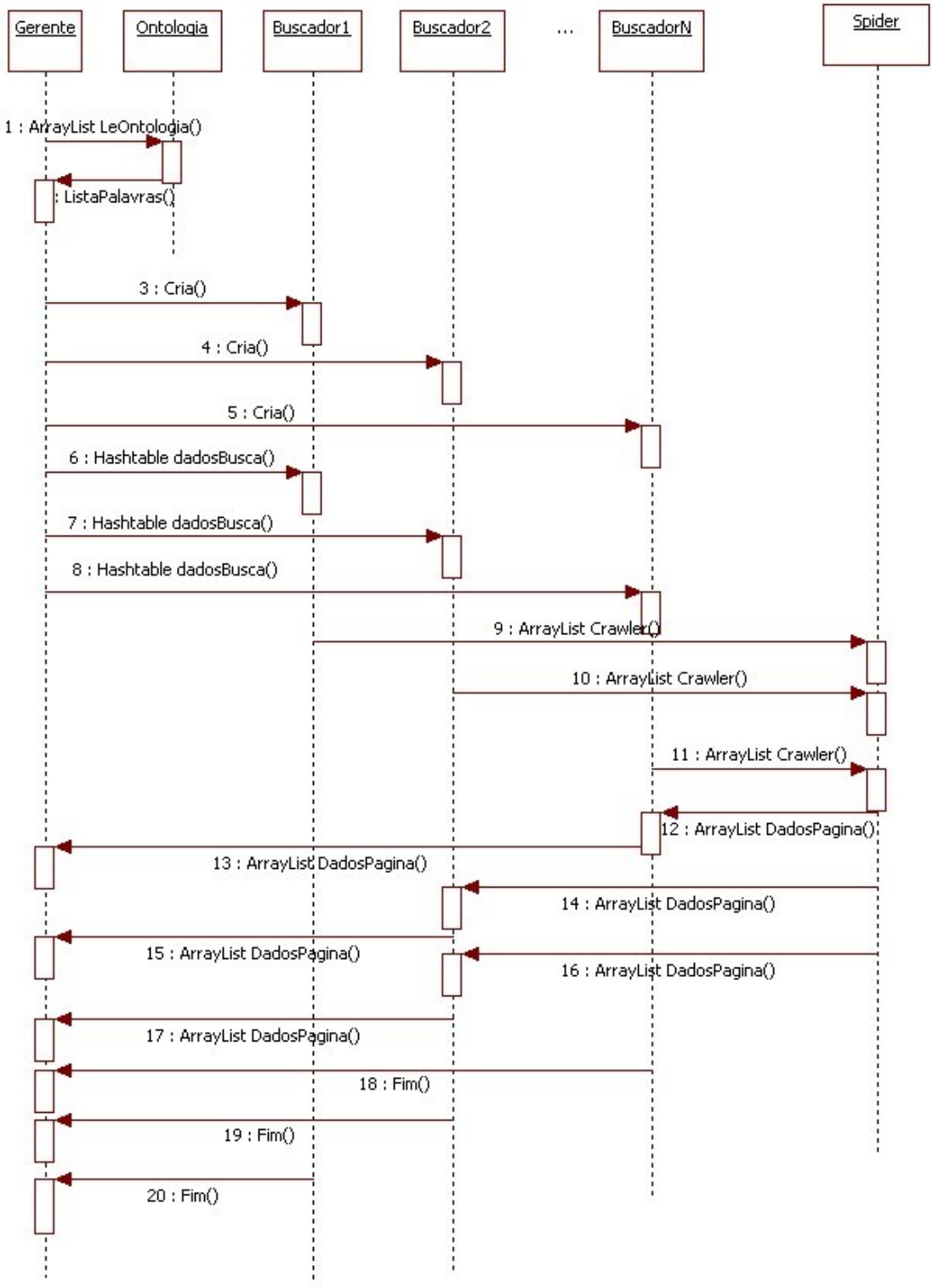

Figura 11: Diagrama de Sequência de Eventos

5.3 ESTUDO DE CASO

Para testar o mecanismo de busca foram feitos testes utilizando-se uma ontologia do contexto de astronomia. Essa ontologia foi desenvolvida com base na ontologia desenvolvida pelo Departamento de Astronomia da Universidade de Maryland (UMD, 2009). A ontologia precisou ser alterada pois as ontologias existentes mostram-se muito genéricas para serem utilizadas com sucesso no filtro semântico. Essa ontologia relaciona termos do contexto de astronomia conforme apresentado na Figura 12.

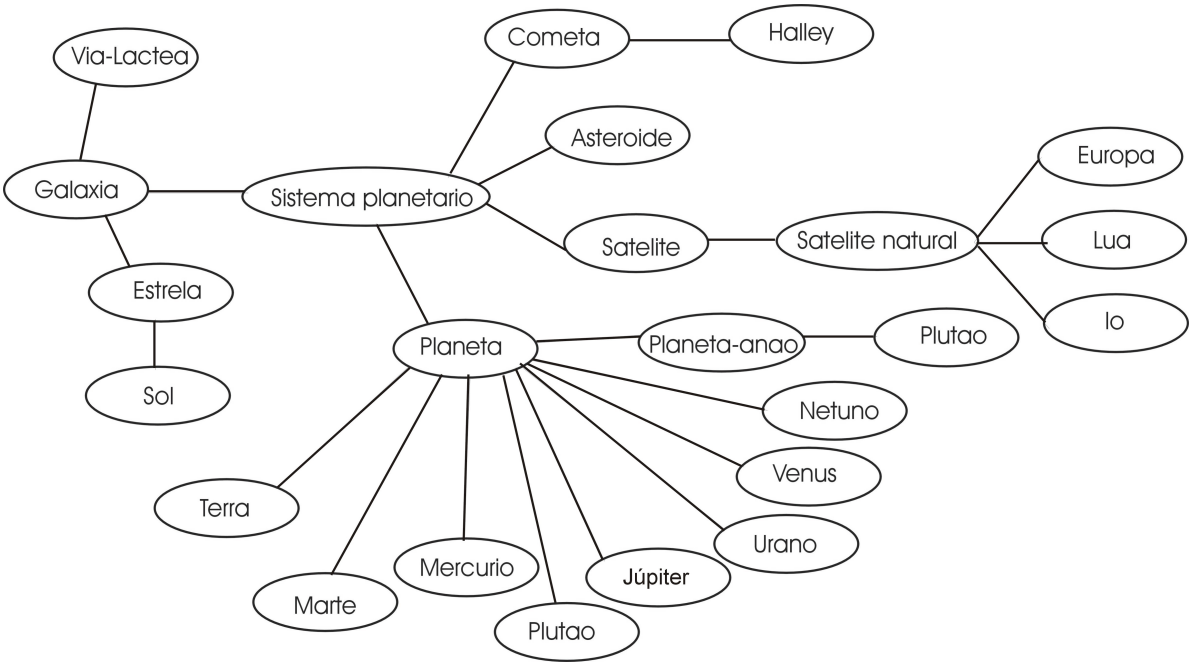

Figura 12: Representação gráfica da ontologia usada como exemplo

Para criação da semente foram utilizados endereços de *sites* diversos. A semente conta com uma lista de 125 *sites* apresentados como resposta à pesquisas de termos constantes na ontologia pelos mecanismos de busca tradicionais. *Sites* que continham arquivos PDF, F*lash*, pps, doc ou outros tipos de arquivo diferentes de páginas da *Internet* foram ignorados na busca.

Uma requisição de busca pelas palavras *terra* e *lua* foi feita ao *Web Searcher*. O agente Gerente realizou a leitura da ontologia e retornou as seguintes palavras: *galaxia, via-lactea, estrela, sistema planetario, cometa, planeta, satelite, asteroides, halley, planeta-anao, natural, plutao, lua, io, europa, mercurio, venus, terra, marte, juptier, saturno, urano, netuno,*

sistema solar, sol. Analisando a ontologia as palavras que possuem maior relação com os termos requisitados para a busca, e que portanto terão peso 10 durante a busca, são *sistema planetario, planeta, terra, satelite, satelite natural, lua*. As demais palavras terão peso 5.

Para cada uma dessas palavras o mecanismo de busca criou um agente Buscador. Cada Buscador é responsável pela busca de uma dessas palavras em um subconjunto dos *sites* constantes no arquivo semente. O primeiro agente de busca, por exemplo, analisará os 5 dos *sites* existentes na semente em busca da palavra *galaxia.* Cada *site* analisado pelo primeiro Buscador é enviado para que os demais buscadores procurem pelas demais palavras. Ao final, a pontuação do *site* é calculada com base na quantidade de vezes que todas as palavras da ontologia foram encontradas naquele *site*. Por exemplo, o *site http://astro.if.ufrgs.br* teve um total de 80 pontos encontrados nas palavras *asteroide* (5 pontos), *estrela* (5 pontos), *lua* (40 pontos), *marte* (5 pontos) e *sol* (5 pontos).

Após essa pesquisa os resultados foram ordenados e mostrados para o usuário. Os *sites* de melhor pontuação foram:

- Fases da Lua
  http://astro.if.ufrgs.br/lua/lua.htm;
- A Lua é uma estrela?
  http://br.answers.yahoo.com/question/index?qid= 20090127075344AA2KnCg;
- Eclipse:
  http://pt.wikipedia.org/wiki/Eclipse;
- Sistema Terra-Lua:
  http://zenite.nu/05/3-ter.php;
- Influência da Lua nas marés da Terra
  http://curiofisica.com.br/influencia-da-lua-nas-mares-da-terra;
- A Terra, a Lua, o Sol: os nossos relógios:
  http://calendario.incubadora.fapesp.br/portal/textos/professor/ptexto05 ;
- Terra e Lua – Luli e Luciana (letra) :
  http//letras.terra.com.br/luli-lucina/376840;
- Alguns dados sobre o planeta Terra:
  http://cfh.ufsc.br/~planetar/textos/terrabege.htm;
- Lua de Mel – Terra – Lua de Mel:
  http://mulher.terra.com.br/noivas/interna/0,,OI524523-EI4883,00.html;

- Observação do Céu:
  http://nautilus.fis.uc.pt/astro/hu/obser/corpo.html.

Como pode ser visto, os resultados apresentados pelo *Web Searcher Agent,* em sua maioria, são resultados relevantes para a busca. Neste exemplo, das10 páginas apresentadas pelo agente, 8 contém informações relevantes no contexto da astronomia.

Para efeito de comparação de resultados, a mesma pesquisa foi feita no mecanismo de busca *Google*. Dos 8 principais resultados apresentados pelo mecanismo, cinco são resultados relevantes para o contexto de astronomia. A Figura 13 mostra o resultado apresentado pelo mecanismo.

Figura 13: Resultado apresentado pelo mecanismo Google (GOOGLE, 2009)

6 CONCLUSÃO

O estudo realizado para desenvolvimento do agente de busca semântica apresentado neste trabalho mostrou que a filtragem semântica é uma alternativa viável para melhorar os resultados apresentados pelos mecanismos de busca na *Web*. O trabalho foi realizado em duas fases, sendo a primeira uma fase de pesquisa para definição dos termos e conceitos utilizados. A pesquisa forneceu a base para que o agente fosse desenvolvido corretamente. A segunda fase foi a implementação do agente e a realização de testes para determinar a relevância dos resultados apresentados.

O *Web Searcher Agent* descrito apresentou resultados satisfatórios na busca de informações, apesar de sua simplicidade de implementação. A filtragem utilizando ontologias facilita a recuperação de informações relevantes pois permite que as páginas sejam filtradas por uma série de termos ligados ao contexto da busca e não somente pelos termos informados pelo usuário. Porém, essa abordagem requer que a ontologia utilizada seja específica do contexto a ser pesquisado e especifique as entidades pertencentes a este conceito. Para permitir a busca contextual, o mecanismo deve especificar o domínio de conhecimento onde a busca será executada.

O agente realiza a busca direto na *Web*. Dessa forma para verificar se uma página é relevante ou não o agente precisa fazer a leitura da página e analisar os termos encontrados. Essa abordagem, apesar de ser de simples implementação, torna a busca lenta e não permite que um número grande de páginas seja visitado. Uma alternativa seria a criação de índices como aqueles utilizados pelos mecanismos de busca atuais. O agente de busca contextual pode ser utilizado para realizar a busca e a filtragem neste índice, o que tornaria a busca mais rápida e abrangente. Outras melhorias que podem ser feitas no agente incluem:

- Definição de ontologias específicas para serem utilizadas juntamente com o agente de busca. Essas ontologias apresentariam mais detalhes sobre os termos do contexto da busca, especificando entidades, não somente as classes. Por exemplo, uma ontologia para o contexto de astronomia deve incluir as entidades *Sol*, *Lua*, *Terra*, não somente as classes *Estrela*, *Satélite*, *Planeta*;
- A abordagem atual de busca é centralizada, utilizando um agente Gerente para organizar a troca de informações entre os agentes Buscadores. O uso de uma abordagem distribuída para implementação dos agentes de busca permitiria que a busca fosse feita com mais rapidez. Atuando de forma distribuída, os agentes Buscadores precisariam trocar informações entre si para que os *sites* fossem corretamente filtrados;
- Cada agente de busca pode ser disparado tendo conhecimento completo sobre a ontologia e não somente de uma palavra. Assim cada agente teria a capacidade de pontuar sozinho as

páginas visitadas, não precisando solicitar que outros agentes analisem a página. Nessa abordagem torna-se necessária a definição de regras específicas que determinarão como os agentes irão coordenar as páginas encontradas, como será feita a remoção de resultados duplicados e a ordenação dos resultados que serão apresentados.

Um dos principais objetivos de Berners-Lee (BERNERS-LEE et al., 2001) com a *Web* Semântica é tornar o conteúdo da *Web* compreensível pela máquina, permitindo que agentes e aplicações acessem uma enorme gama de recursos heterogêneos buscando automatizar suas atividades com o mínimo de interação ou interpretação humana. Por enquanto, os esforços tem se concentrado em arquiteturas que buscam uma homogeneização dos recursos através de XML, RDF, RDFS, OWL, entre outras representações. Possivelmente a evolução destes padrões realmente direcione *designs* arquiteturais sofisticados da *Web*, mas é possível também que soluções surpreendentemente simples sejam alternativas mais elegantes para a "significação" na *Web*, preconizada por Berners-Lee. Embora o presente trabalho não pretenda em momento algum solucionar estes problemas, procuramos estabelecer um estudo exploratório sobre o uso de uma abordagem multiagentes em uma espécie de trabalho colaborativo inferencial baseado em contextos controlados.

7 REFERÊNCIAS

ALMEIDA, Rubens Queiroz de. Busca de Informações na Web**.** In: **EAD Minicursos Virtuais.** UNICAMP, São Paulo – SP, 2002

BARROS, Flávia. **A Inteligência Artificial sob o olhar dos Agentes Inteligentes.** Universidade Federal de Pernambuco, 2002

BERNERS-LEE, Tim; HENDLER, James; LASSILA, Ora. The Semantic Web. **Scientific American Magazine,** v. 184, n. 5, 2001, pp. 34-43. Disponível em <http://www.sciam.com/article.cfm?id=the-semantic-web>. Acesso em 30 jan 2009.

BLUM, Thom, KEISLAR, Doug, WHEATON, Jim, WOLD, Erling. **Writing a Web Crawler in the Java Programming Language.** 1998 Disponível em < http:// java.sun.com/developer/technicalArticles/ThirdParty/WebCrawler/>. Acesso em 23 mar 2009

BRANSKI, Regina Meyer. Recuperação de Informações na Web. In: **Perspectivas em Ciência da Informação.** Belo Horizonte, Vol. 9 N° 1, 2004. Disponível em <http://www.eci.ufmg.br/pcionline/index.php/pci/article/viewFile/351/160>

BRIN, Sergey, PAGE, Lawrence. The Anatomy of a Large-Scale Hypertextual Web Search Engine**.** In: **Computer Networks and ISDN Systems. Stanford**, Vol. 30 Pg 107-117, 1998. Disponível em <http://www-db.stanford.edu/pub/papers/google.pdf>

CASTILLO, Carlos. **Effective Web Crawling.** Santiago do Chile, 2004. 179 f. Tese (Doutorado) – Universidade do Chile, 2004.

CISCON, Leonardo Aparecido, ALVES, Rêmulo Maia. **Desenvolvimento de um Sistema Web de Busca Inteligente para Suporte à Tomada de Decisões no Agronegócio do Café.** In: Convibra – Congresso Virtual Brasileiro de Administração, Lavras, 2004

CLEVER Project, The. **Hypersearching the Web.** Scientific American Magazine, June 1999. p.54-60. Disponível em <http://www.scientificamerican.com/article.cfm?id= hypersearching-the-web>

COPERNIC Inc. **Copernic Agent Family**, 2009. Disponível em <http://www.copernic .com/>

DOLCE. **Laboratory for Applied Ontology**, 2006. Disponível em <http://www.loa-cnr.it/DOLCE.html>

EUBANKS, Brian D. **Wicked Cool Java: Crawling the Semantic. Get started with RDF** Web. In: JavaWorld.com, 2005. Disponível em <http://www.javaworld.com/javaworld/jw-12-2005/jw-1205-wicked.html>

FREITAS, Fred. **Agentes de Busca na Internet.** Material utilizado na cadeira de Tópicos Avançados em Computação Inteligente. UFPE, Pernambuco, 1997.

GENESIS. **Global Environmental & Earth Science Information System**, 2009. Disponível em < http://genesis.jpl.nasa.gov/zope/GENESIS>

GOOGLE, **Visão Geral da Tecnologia**, 2009. Disponível em <http://www.google.com/

corporate/tech.html>.

GRUBER, Thomas R. **Towards Principles for the Design of Ontologies Used for Knowledge Sharing**. In: International Journal of Human and Computer Studies, Vol 43. Agosto 1993 p. 907-928. Disponível em <http://tomgruber.org/writing/onto-design.htm>

GUIMARÃES, Francisco J. Z. **Utilização de Ontologias no Domínio B2C**. Rio de Janeiro, 2002. 195 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Informática, Pontifícia Universidade Católica do Rio de Janeiro, 2002.

HENDLER, James. Agents and the Semantic Web. **IEEE Intelligent Systems,** v. 16, n. 2, 2001, pp. 30-37. IEEE Computer Society, Washington – DC.

HÜBNER, Jomi F.; SICHMAN, Jaime S. **Organização de Sistemas Multiagentes,** In: **III Jornada de Mini-Cursos de Inteligênica Artificial**, Vol. 8, 2003, pp. 247-296. SBC, Campinas – SP. Disponível em <*moise.sourceforge.net/doc/orgSMA-jaia-2003.pdf* >. Acesso em 29 dez 2008

IVASSAKI, Ivone Matiko, **Jornal on-line: Personalização do Conteúdo Através da Tecnologia de Agentes Inteligentes.** Dissertação (Mestrado) – Faculdade de Comunicação, Educação e Turismo. Universidade de Marília, Unimar. 94 f. Marília, 2006

IZU, André Kuraoka. **Mecanismos de Busca na Web.** Trabalho de Formatura Supervisionado – Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo – SP, 2006. Disponível em <http://www.linux.ime.usp.br/~izu/mac499/>

JENA. Jena – **A semantic Web Framework for Java.** 2009 Disponível em <http://jena. sourceforge.net/index.html>

LEMOS, Daniel, BONUTTI, Rafael, INÁCIO, Rodrigo. **Semantic Web Agents / Web Services.** Universidade Federal de Minas Gerais. Belo Horizonte, 2008

LEMKE, Ana Paula. **Um Framework para a organização do conhecimento de agentes de software**. 131 p. Mestrado em Ciência da Computação (Dissertação). Pontifícia Universidade Católica do Rio Grande do Sul. Porto Alegre, 2007

LOURENÇO, Ana *et al*. **Introdução à Inteligência Artificial: Sistemas multiagentes (SMA).** Monografia - Universidade da Madeira, 2006

MANNING, Christopher D., RAGHAVAN, Prabhakar, SCHÜTZE, Hinrich. **Introduction to Information Retrieval.** Cambridge University Press, Cambridge, 2008. Disponível em <http://www-csli.stanford.edu/~hinrich/information-retrieval-book.html>.

NASA, **Semantic Web for Earth and Environmental Terminology (SWEET)**, 2009. Disponível em <http://sweet.jpl.nasa.gov/ontology/>

NORMAN, Timothy J., SIERRA, Carlos, JENNINGS, Nick R., **Rights and Commitment in Multi-Agent Agreements**, 3rd International Conference on Multi-Agent Systems, Paris, France, 1995

NORVIG, Peter. Internet Searching**.** In: **Computer Science: Reflections on the field, Reflections from the field,** The National Academies Press, Washington. D.C. 2004. Disponível em <http://www.norvig.com/InternetSearching.pdf >.

OLIVEIRA, Rosa M. V. B. **Web Semântica: novo desafio para os profissionais da Informação.** *In:* **Seminário Nacional de Bibliotecas Universitárias**, 12., 2002, Recife. Anais. 2002. Disponível em <*www.sibi.ufrj.br/snbu/snbu2002/oralpdf/124.a.pdf*> . Acesso em 29 dez 2008.

PATTERSON, Anna. Why Writing Your Own Search Engine is Hard. **Queue,** Stanford University, Stanford,Vol 2, n° 2, 2004. Disponível em <http://queue.acm.org/detail.cfm?id=988407>

PEREIRA, Vasco Nuno Souza Simões. **Arquitectura de um Motor de Busca: Exemplo do Google**. Departamento de Engenharia Informática Universidade de Coimbra, Coimbra, 2004

PROTÉGÉ. Stanford Center for Biomedical Resource, Stanfort University School of Medicine, 2009. Disponível em <http://protege.stanford.edu/>

REIS, Luís Paulo. **Coordenação em Sistemas Multiagente: Aplicação na Gestão Universitária e Futebol Robótico.** Tese (Doutorado), Faculdade de Engenharia da Universidade do Porto. Porto, 2003. Disponível em <http://paginas.fe.up.pt/~lpreis/Research.htm>

RIBEIRO, Marcelo Blois. **Desenvolvimento de Sistemas Inteligentes**. Material utilizado na cadeira de Desenvolvimento de Sistemas Inteligentes. Programa de Pós Graduação em Ciência da Computação da Pontifícia Universidade Católica do Rio Grande do Sul - PUCRS, Porto Alegre, 2007. Disponível em <http://www.inf.pucrs.br/~blois/materiais/dsi/>

RIBEIRO, Marcelo Blois, ESCOBAR, Maurício da Silva. **Minicurso: Agentes e Ambientes de Programação para a Web**. In: WESAAC 2007 – Workshop – Escola de Sistemas de Agentes para Ambientes Colaborativos. Pelotas, 16 a 18 de Abril, 2007. Disponível em <http://ppginf.ucpel.tche.br/wesaac/Anais/Mini-cursos/mini-curso-lois.pdf>. Acesso em 13 maio 2009

RIBEIRO, Marcelo Blois, ESCOBAR, Maurício da Silva, CHOREN, Ricardo. **Using Agents and Ontologies for Application Development on the Semantic Web.** Journal of the Brazilian Computer Society, v. 1, p. 1-15, 2007. Disponível em <http://www.sbc.org.br/bibliotecadigital>

RIBEIRO, Marcelo Blois, ESCOBAR, Maurício, LEMKE, Ana Paula. **SemantiCore 2006 – Permitindo o Desenvolvimento de Aplicações Baseadas em Agentes na Web Semântica.** In: Second Workshop on Software Engeneering for Agent-Oriented Systems, 2006, Florianópolis. SEAS 2006. SBC: Florianópolis, 2006. v. 1. p. 72-82

RICOTTA, Fábio Carvalho Motta. **Como os Search Engines Funcionam?** Projeto Final de Graduação. Departamento de Matemática e Computação. Universidade Federal de Itajubá, Itajubá, 2007

RUSSEL, Stuart, NORVIG, Peter, **Artificial Intelligence – A Modern Approach**, Prentice Hall Series in Artificial Intelligence, 1995

SANTOS, Nilson Moutinho dos et al.. **Agentes Autônomos Inteligentes Um Tutorial.** 2000. Disponível em <http://www.din.uem.br/ia/vida/agentes/index.htm>

SCHILDT, Herbert, HOLMES, James. **Crawling the Web with Java**. In: The Art of Java, McGraw-Hill, 2004.

SILVA, Alberto Sales e. **Web Semântica: O Estado da Arte**. Programa de Pós-Graduação em Ciência da Computação. Pontifícia Universidade Católica do Rio Grande do Sul. 2008.

SMITH, Reid G. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. In IEEE Transactions on Coomputers, v. C-29. N 12. Dec 2009. Disponível em <http://www.rgsmithassociates.com/The_Contract_Net_Protocol_Dec-1980.pdf>

SPIRE Research Group. **Spire**, 2009. Disponível em <http://spire.umbc.edu/ont/>

SWOOGLE **Semantic Web Search,** 2007. Disponível em <http://swoogle.umbc.edu/>

UMD, **Simple HTML Ontology Extenisions**, 2009. Disponível em <http://www.cs.umd.edu/projects/plus/SHOE>

W3C – Word Wide Web Consortium Oficina España. **Guía Breve de Web Semántica.** Gijón, 2008. Disponível em < http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica >. Acesso em 20 abr 2009

W3C – Word Wide Web Consortium. **Resource Description Framework (RDF)**. Disponível em < http://www.w3.org/RDF/>

WOOLDRIDGE, Michael. **An Introduction to Multiagent Systems**. John Wiley & Sons Ltd, 2002

YAMAOKA, Eloi Juniti. **Recuperação de Informações na WEB.** Brasília, 2003. 18 f. Disponível em <http://www.ct.ufrj.br/bib/bibliotecaonline/pesqapoio/Recuperacao_informacao_web.pdf >. Acesso em 26 dez 2008.