UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Eduardo Menna da Silva

ARQUITETURA PARA RECUPERAÇÃO DE INFORMAÇÃO EM DOCUMENTOS ANOTADOS USANDO SEMÂNTICA

Dissertação de Mestrado submetida à Universidade Federal de Santa Catarina como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Fernando Álvaro Ostuni Gauthier

Florianópolis, abril de 2008.

Dedico este trabalho àqueles que mais amo, meus pais.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

API	Aplication Programmer Interface
Compsem	Grupo de Computação Semântica da UFSC
CGI	Common Gateway Interface
DAML	DARPA Agent Markup Language
DTD	Document Type Definition
IDE	Integrated Development Environment
GML	General Markup Language
HTML	Hipertext Markup Language
OIL	Ontology Inference Layer
OWL	Ontology Web Language
PHP	Hypertext Preprocessor
NLP	Natural Language Processing
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WWW	World Wide Web
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XOL	XML-based Ontology Exchange Language
SDK	Java Development Kit
SGML	Standard Generalized Markup Language

Resumo

O presente trabalho apresenta uma arquitetura de recuperação de informação em nível semântico para documentos anotados. Tal arquitetura visa organizar os recursos de informação e permitir que usuários tenham maior precisão quanto aos resultados das consultas por documentos de seus interesses.

A abordagem se faz importante, pois os usuais meios de recuperação de informação não são capazes de restringir os resultados das consultas. Isso ocorre em virtude da grande quantidade de dados indexados e a pouca capacidade em analisá-los.

A arquitetura aqui apresentada baseia-se em ontologias para responder às consultas mais avançadas efetuadas pelos usuários. Ontologias possuem grande capacidade em representar conhecimento, provendo uma fonte muito rica para responder a requisições. Assim, aplicações desenvolvidas dentro dessa concepção têm capacidade de recuperar, além de informações explícitas, também informações implícitas aos usuários.

É apoiando-se sobre essa constatação que se desenvolve este trabalho. São apresentados os principais conceitos sobre a área, métodos de recuperação de informação e a arquitetura proposta para recuperação de informação usando ontologias.

Palavras-chave: Web Semântica, Recuperação de Informação, Nível semântico, Ontologia.

Abstract

The present work shows information recovery architecture on semantic level to annotated documents. This architecture aims to organize information sources and to allow that user has more precision about query results by documents of their interests.

The approach is important because the usual information recovery doesn't be able to restrict query results. This happen because there are amount indexes data and quiet capability in analyses them.

The architecture here presented is based on ontologies to response advanced query executed for user. Ontologies have large capability in knowledge represent providing a rich source to response requests. So, application developed into this conception has capability in recovery, besides explicit information, also implicit information to the end users.

Based in this finding the present work is developed. Are presents the mains concepts area, information recovery methods and the architecture here proposed to recovery information using ontologies.

Keywords: Semantic Web, Information Retrieves, Semantic Layer, Ontology.

1 Introdução

Este capítulo apresenta, sucintamente, os principais componentes que permeiam esse trabalho de dissertação. Primeiramente, faz-se uma breve introdução do problema que motiva esse projeto, a fim de dar ao leitor uma idéia acerca do tema.

Posteriormente, são apresentados os objetivos geral e específicos, os quais resultam na definição de uma arquitetura e desenvolvimento de uma ferramenta para validação da arquitetura proposta. Por fim, descreve-se o mecanismo de recuperação de informação dentro da Web Semântica, o qual é a proposta do trabalho .

1.1 Considerações iniciais

O presente trabalho é subordinado ao projeto denominado InWebS. Tal projeto foi concebido pelo grupo de Computação Semântica da UFSC – CompSem e visa ser uma plataforma de desenvolvimento de trabalhos voltados à área de Web Semântica.

Dentre os subprojetos envolvidos destacam-se: (*i*) OntoInWebS, editor e manipulador de Ontologias; (*ii*) RegInWebS, manipulador de regras semânticas; (*iii*) MarSInWebS, anotação e recuperação de informação semântica; e, (*iv*) RecInWebS, módulo de raciocínio sobre conhecimento representado em ontologias.

O trabalho descrito aqui pertence ao subprojeto MarSInWebS, e exerce a função de recuperador de informação semântica. Para a correta aplicação desse trabalho é necessária a existência do mecanismo de anotação semântica desenvolvido dentro do mesmo subprojeto.

1.2 Descrição da Proposta

A proposta visa recuperar informações interessantes aos usuários, a partir de documentos quaisquer armazenados em um repositório de dados. Para isso, o sistema trabalha de forma reativa. O usuário constrói as consultas que deseja executar e submete a uma ferramenta desenvolvida sobre a arquitetura aqui proposta. Tais consultas podem ser por palavras-chaves sobre ontologias, classes ou intâncias. É permitido o refinamento das consultas através de sucessivas buscas de acordo com os termos pesquisados pelo usuário. Além disso, há a opção de construir consultas avançadas, através da seleção de subclasses ou classes em mesmo nível para cada um dos termos informados na consulta.

Para organizar todo o desenvolvimento dessa arquitetura, opta-se por trabalhar em duas etapas: (*i*) indexação de informação, e; (*ii*) recuperação de documentos através dos dados indexados.

A indexação da informação é responsável por capturar documentos que contenham anotação semântica e extrair a parte que diz respeito a essa anotação. A extração dos registros semânticos popula uma base de dados relacional que serve como suporte para a segunda etapa do trabalho. Já a etapa de recuperação tem o intuito de executar buscas estimuladas por usuários que desejam ter acesso a determinados tipos de documentos existentes no repositório.

O conceito de indexação de informação é muito similar entre as ferramentas de busca existentes hoje em dia. Basicamente, em todas elas se utiliza um repositório central que concentra todos os documentos a serem pesquisados e recuperados. Além disso, ainda existe uma tabela que referencia todos os termos aos documentos que cada qual se relaciona. Na proposta desse trabalho, a abordagem de repositório central também é aplicada. Porém, existem duas fontes de dados; A dos documentos propriamente ditos e a fonte de ontologias.

Quanto à recuperação, a maioria das ferramentas utiliza unicamente a tabela que referencia termos a documentos. Mesmo as ferramentas de busca voltadas para a Web Semântica, as quais permitem o uso da semântica para melhor especificar as buscas, usam somente a tabela que relaciona instâncias de uma anotação a documentos onde existam tais instâncias. Essa característica de busca é denominada busca explícita e só

capta informações superficiais. Nem mesmo a ontologia a qual está relacionada um documento é utilizada para efetuar o processo de consulta.

A idéia do trabalho proposto aqui é, em um primeiro momento, melhor representar as informações de uma anotação em um banco de dados. As informações não são concentradas em apenas uma tabela, mas sim utilizando um conjunto de tabelas relacionadas entre si. Isso reflete em um melhor desempenho nas respostas às consultas dos usuários. Outro detalhe do trabalho é a utilização da ontologia para responder às consultas. Com isso, não somente as informações explícitas serão capturadas, mas também as implícitas que somente são alcançadas através do uso da ontologia.

1.3 Justificativa

A Web Semântica tem por objetivo incrementar o processo de representação e recuperação do conhecimento nas redes [DOAN et al., 2002] e [MCILRAITH et al., 2001]. Como forma de representação, a Ontologia [GRAU, 2004] surge como o principal componente capaz de exercer esse papel. São responsáveis por descrever conhecimentos sobre um determinado tema ou área. A associação de uma ontologia a um documento resulta em um metadado do documento denominado anotação semântica. A anotação é uma descrição que pode trazer informações, tais quais: do que se trata o documento, quem é o autor, entre outros descritivos.

Um metadado contém instâncias de classes descritas dentro de uma ontologia. A anotação é uma peça importante, pois além de exercer o papel de elo entre o conhecimento descrito na ontologia e documento, também é utilizada pela maioria das ferramentas de recuperação de dados semânticos como fonte de informação capaz de auxiliar nas buscas.

Porém, existem limitações em muitas dessas ferramentas de recuperação na Web Semântica. Por exemplo, costuma-se utilizar apenas as instâncias existentes dentro de anotações. A Figura 1 - Exemplo de anotação auxilia na descrição de uma possível limitação das ferramentas de busca.

```
<xml>
  <classe nome='cao'>
    <instancia nome='snoop' />
  </classe>
  <classe nome='gato'>
    <instancia nome='felix' />
  </classe>
</xml>
```

Figura 1 - Exemplo de anotação

Considera-se que a anotação contenha as classes Cão e Gato. Respectivamente, há as instâncias Snoop e Felix. A maioria das ferramentas consegue apenas recuperar consultas do tipo: "*Esse documento trata de Cão?*". Esse tipo de consulta considera somente as instâncias que constam dentro da anotacao. Praticamente, não ocorre a utilização da ontologia propriamente dita.

Todavia, o usuário poderia alterar a consulta para: "*Quais são os documentos que falam de mamíferos?*". Nesse caso, as ferramentas de busca, em sua maioria, não conseguiriam recuperar esse tipo de informação. Isso acontece porque tais ferramentas apenas procuram instâncias diretas de Mamífero na anotação do documento. Superclasess e subclasses são desconsideradas no momento da pesquisa. No exemplo apresentado na Figura 1 - Exemplo de anotação não há explicitamente instâncias da classe Mamífero.

Todavia, através da utilização de uma ontologia que descreva mamíferos e afins, seria possível identificar que a classe Mamífero possui subclasses, e essas por sua vez possuem instâncias que podem ser retornadas como instâncias de Mamífero. Esse tipo de relacionamento descoberto somente com o auxílio da ontologia são classificadas como informações implícitas aos usuários.

Outra limitação conhecida das ferramentas de recuperação semântica diz respeito às pesquisas por classes filhas de uma mesma superclasse. Esse tipo de relacionamento, também conhecido como classes vizinhas, necessita da ontologia para ser descoberto.

Frente à carência de ferramentas que contemplam maior capacidade em responder consultas de usuários, é que se propõe esse trabalho.

1.4 Objetivos

1.4.1 Objetivo geral

O objetivo do trabalho é projetar e desenvolver um mecanismo de recuperação de informação mais eficaz que as abordagens normalmente encontradas na literatura. A eficiência está relacionada à utilização da arquitetura de recuperação de informação em nível semântico. Tal arquitetura descrita aqui faz uso de ontologias, com o intuito de revelar informações implícitas às anotações. Entende-se por informações implícitas àquelas que somente são descobertas efetuando pesquisas na ontologia, sendo possível identificar relacionamentos de subclasses ou classes de mesmo nível, por exemplo.

1.4.2 Objetivos específicos

Os objetivos específicos desse projeto são: a organização de repositórios de dados capazes de armazenar documentos e ontologias; a utilização de um módulo indexador com o intuito de capturar as informações existentes nas anotações semânticas de documentos, e; um módulo responsável por recuperar, efetivamente, as informações contidas na base de conhecimento. Esse último faz uso dos dados indexados e retorna documentos armazenados no repositório.

1.5 Organização do trabalho

O presente trabalho está estrutura sob sete capítulos. Abaixo há um pequeno relato de cada um:

- O capítulo atual traz a introdução acerca do trabalho desenvolvido, comentando a sua justificativa e objetivo traçado dentro do projeto;
- O segundo capítulo faz um apanhado geral sobre os mecanismos de recuperação de informação;
- O terceiro capítulo aborda a área principal na qual está inserido o este trabalho. A estrutura da Web Semântica é apresentada e comentada através dos seus níveis estruturais;
- A arquitetura construída a partir do levantamento bibliográfico é apresentado no capítulo quatro;
- O capítulo cinco traz a demonstração da ferramenta desenvolvida baseada na arquitetura proposta. Tal ferramenta tem o intuito de validar a arquitetura;
- O capítulo seis apresenta as considerações finais acerca do trabalho descrito neste documento;
- Por fim, são apresentadas as referências bibliográficas utilizadas como fonte de informação para a construção do trabalho de dissertação.

1.6 Considerações do capítulo

Neste capítulo foi apresentada, de forma sucinta, a descrição do problema que motivou o desenvolvimento do presente trabalho, além de apresentar os objetivos geral e específicos e, por fim, uma descrição do que consiste a proposta de solução para o problema verificado.

No próximo capítulo, será apresentada a Web Semântica, conceito central desse projeto. Para essa apresentação, descreve-se o conceito em todas as suas camadas para um melhor entendimento de todo o escopo que a Web Semântica engloba.

2 Mecanismos de Busca - Conceitos, Características e Limitações

Este capítulo apresenta conceitos acerca dos mecanismos de busca, suas estruturações internas e classificações. O conceito de agentes de busca também é abordado, visto que, também são considerados importantes na busca por informações em um determinado ambiente. Por fim, comentam-se as características principais de uma anotação semântica e seus tipos de representação.

2.1 Mecanismos de Busca de Informação

Os mecanismos de busca de informação têm papel fundamental na estrutura global da Web. Da mesma forma que a rede se destaca por compartilhar inúmeros recursos, a mesma importância é dada aos mecanismos de captura dessas informações. Afinal, de que serviria a Web se não posse possível a consulta de conteúdos por parte dos usuários?

Esses mecanismos de busca, também conhecidos como Máquinas de Busca, têm o objetivo de, a partir de uma consulta efetuada por um usuário ou software, recuperar e processar informações contidas em documentos quaisquer. Os autores, em [MORAIS; SOARES, 2004] classificam as máquinas de busca (mecanismos de recuperação de informação) em três gerações:

- Primeira Geração: Utiliza somente informações intrínsecas aos documentos para prover a recuperação de informação. Essas informações compreendem as encontradas nos textos, que, por exemplo, podem ser determinadas através da ocorrência de palavras e sua localização no texto;

- Segunda Geração: utiliza, além de intrínsecas, informações extrínsecas aos documentos. Entendem-se aqui por informações extrínsecas os demais documentos representados dentro de outro documento, como *links*, por exemplo;
- Terceira Geração: Efetuam a recuperação de informação baseada na semântica das informações contidas nos documentos. Como exemplo, pode-se citar as anotações semânticas existentes em documentos.

Apesar do avanço quanto à geração dos mecanismos de recuperação de informação, muitas ferramentas amplamente utilizadas na atualidade, ainda se baseiam, principalmente, nas abordagens da primeira e segunda gerações. Alguns exemplos, bem conhecidos, são os mecanismos mais conhecidos de recuperação de informação, tais como: Google[1], Yahoo[2], Alta Vista [3]e All the Web[4] etc.

As citadas máquinas de busca se utilizam de diversas políticas de classificação do conteúdo da Web para agilizar o processo de resposta às consultas dos usuários. Porém, esses mecanismos, por estarem compreendidos às Gerações Primeira e Segunda, não apresentam suporte a recuperação de informação baseada em Semântica.

Além disso, outras falhas são verificadas segundo [RILOFF; LEHNERT, 1994], como por exemplo: problemas com palavras sinônimas, polissemia, frases (conjunção de duas ou mais palavras que resultam em uma expressão significativa), ou ainda, contexto local (considerando o problema da frase, dentro de uma expressão pode haver uma palavra mais importante e que direciona a busca).

Mais informações sobre os problemas enfrentados por esses mecanismos juntamente com as descrições das políticas de recuperação podem ser melhor exploradas em [BARROSO et al., 2003], [KRAFT et al., 2006] e [JANSEN et al., 2003].

---

[1] www.google.com

[2] www.yahoo.com

[3] www.altavista.com

[4] www.alltheweb.com

2.2 Recuperação e Extração de Informação

Segundo [SILVA, 2003], as atividades desenvolvidas pelos mecanismos de busca podem ser classificadas de acordo com o modo de recuperação de informação e o refinamento que é feito sobre as informações capturadas. Frente a esses dois aspectos, surgem as técnicas de Recuperação de Informação (RI) e Extração de Informação (EI).

A técnica de RI veio da necessidade de recuperar documentos baseada no fornecimento de palavras-chaves para consulta. A maneira como esses documentos são recuperados são definidos através de dados estatísticos que registram a ocorrência das palavras consultadas dentro do documento. Por exemplo, na busca por determinadas palavras-chaves os mecanismos de busca, tal qual descrito por [ARASU et al., 2001], trazem uma relação dos documentos que contenham, em alguma parte do material, uma ocorrência da palavra-chave. Essas informações são levadas em consideração para uma posterior elaboração de um *ranking* dos resultados, classificando os documentos conforme a ocorrência das palavras pesquisadas nesses documentos.

A técnica é, de certo modo, a mais simples de ser aplicada. Porém, os resultados trazidos pelos mecanismos que aplicam essa técnica são generalistas e pouco precisos. Os principais problemas que são encontrados dizem respeito às palavras sinônimas, polissemia, contexto composto, etc.

No caso das palavras sinônimas, os mecanismos de RI não conseguem capturar, como resultado, a ocorrência de palavras com o mesmo sentido e de escrita diferentes. A polissemia também é outro problema dessa técnica onde é retornada qualquer ocorrência de uma consulta independente do sentido no qual a palavra está inserida. Em consultas de contexto composto, por exemplo, a expressão *submissão de artigo*, os resultados são trazidos analisando cada palavra separadamente, o que distorce a consulta.

Em geral, as consultas utilizando a técnica de RI são pouco precisas. A principal questão são os resultados de busca ruidosos, os quais retornam uma série de informações não tendo nenhuma relação com o que foi consultado inicialmente.

Algumas alternativas foram criadas para diminuir esse problema. Os Dicionários de Dados ou *Therauri*, como apresentado em [CHEN et al., 2003], procuram resolver a questão sinonímia. Já o trabalho de [SANDERSON, 1994] apresenta algumas técnicas polissêmicas.

A técnica de Extração de Informação (EI) utiliza de métodos mais precisos para extrair dados a partir de documentos. A EI se baseia na premissa de que todos os documentos, sendo da Web ou não, e que tratam assuntos mais pontuais, apresentam alta similaridade quanto à formatação, estrutura e, principalmente, conteúdo. De acordo com [SILVA, 2003] esses documentos podem ser agrupados e formarem *classes de páginas*.

Um exemplo da formação dessas *classes de páginas* pode ser representado a partir de uma coleção de *sites* sobre restaurantes. Certamente, todos os sites apresentam o mesmo conteúdo e a estrutura será similar. Existirão seções de *menus*, tabela de preços, contatos com endereços e telefones, etc. Essa semelhança em relação ao domínio simplifica a atuação dos mecanismos de EI.

A EI é uma técnica que trabalha com partes específicas do documento onde há a possibilidade da existência de dados relevantes. Isso pode ser remetido a Web semântica. Se for considerado o fato de que a Web Semântica trabalha com anotações de documentos, pode-se entender que a parte do documento que traz as marcações e anotações é uma parte específica do documento que contém dados relevantes [SWOOGLE, 2005].

Se por um lado a EI é interessante, pois provê a extração de dados de uma forma mais precisa, por outro lado, esses mecanismos de extração ficam limitados ao domínio de conhecimento acerca do conteúdo. A fim de tornar os mecanismos de EI mais flexíveis, alguns trabalhos autores visam associar RI e EI.

Essa idéia de recuperação de informações exatas, que obedecem àquilo que realmente foi solicitado pelo usuário, teve sua expectativa correspondida até meados da década de 1990. Porém, com a popularização da Internet e o inevitável aumento na demanda e no consumo por informação, a tarefa de organizar todo o conteúdo disponível na Web ficou esquecida. Dessa forma, as abordagens de recuperação de informação compreendidas a primeira e segunda gerações passaram a não mais

corresponder da melhor maneira possível às necessidades dos usuários. Foi pensando em reorganizar as informações dispostas aos usuários que a terceira geração da Web surgiu.

Essa geração tem seu princípio de funcionamento baseado no comportamento humano, o qual tem a capacidade semântica de discernir vários conceitos. Por esse fato é que a Web Semântica está diretamente relacionada com a área de Inteligência Artificial da Computação, pois prima por simular o comportamento de um humano na tarefa de classificar, por grau de importância, uma grande quantidade de temas e conceitos que lhe é apresentado.

Assim como os mecanismos da primeira e da segunda geração, os motores de busca compreendidos à terceira geração também obedecem a uma arquitetura largamente aplicada quando o assunto é Recuperação ou Extração de Informação. Essa arquitetura é mais bem representada por [ARASU et al., 2001], o qual sintetiza, perfeitamente, os principais componentes existentes nos mecanismos que recuperam e disponibilizam informação. A Figura 2 apresenta esse conceito de [ARASU et al., 2001].

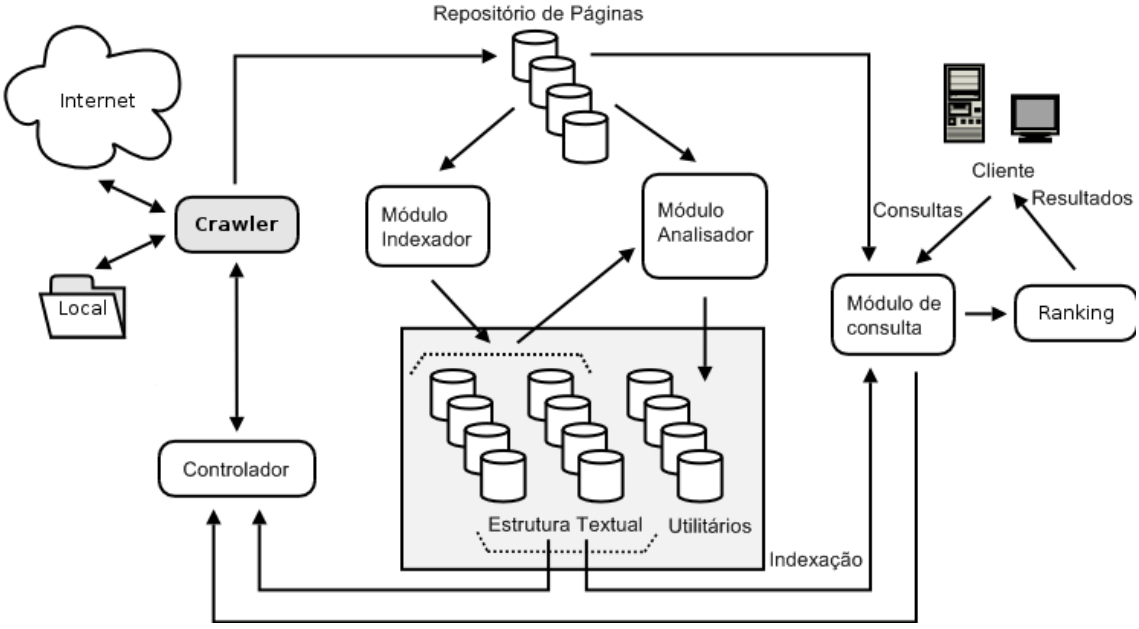

Figura 2 - Arquitetura de um Mecanismo de Busca

[ARASU et al., 2001] representa um mecanismo de busca baseado em três etapas bem definidas. Em ordem, chamam-se: Busca, Indexação e Recuperação. Traçando um paralelo à Figura 2 - Arquitetura de um Mecanismo de Busca, a etapa de Busca corresponde ao *Crawler* e ao *Crawl Control*. Esses dois componentes são comentados logo a seguir.

A Indexação corresponde à parte intermediária à Figura 2 - Arquitetura de um Mecanismo de Busca, a qual engloba o *Page repository*, *Indexer Module*, *Collection Analysis Module* e o *Indexes*. Já a Recuperação compreende, basicamente, ao *Query Module* e ao *Ranking*.

A etapa de Busca é a primeira em um mecanismo de recuperação de informação. Sua responsabilidade é varrer a Web, ou um repositório onde se encontram fontes de informação, e disponibilizar essa fonte de informação, normalmente compreendido como um documento, para a apreciação da etapa Indexadora. Quem cumpre a etapa de buscar por documentos é um agente ou, como mais conhecido nos mecanismos mais recentes, um *Crawler*. Tal *Crawler* é responsável por coletar todos os documentos e disponibilizá-los para outro módulo do Indexador, para que este por sua vez, possa fazer uma análise e catalogar as informações inerentes a esses documentos. O *Crawler* será comentado adiante.

Todavia, o *Crawler* não executa sua tarefa de forma desordenada. Ele somente entra em ação após ser estimulado pelo *Crawl Control*, módulo responsável por repassar ao *Crawler* os dados que estão sendo requisitados pelo usuário, ou também pelo próprio Indexador. Nesse segundo caso, o Indexador solicita uma consulta ao *Crawl Control* nas situações quando este encontra em um arquivo a existência de um *link*, por exemplo, onde é importante buscar as informações dos documentos relacionados com o que está sob análise. Isso exemplifica bem o funcionamento dos mecanismos pertencentes à segunda geração.

Após o *Crawler* capturar as páginas e armazená-las no *Page Repository* entra em ação o *Indexer Module*.

Esse módulo cria uma tabela com as palavras e as URLs onde ocorrem tais vocábulos. Com isso, as consultas podem ser respondidas mais rapidamente. As tabelas geradas a partir desse módulo são representadas pelo *Text Structure*. Além disso, esse

módulo também cria a estrutura de índice importante para o *Crawl Control* e já comentada anteriormente.

Outro módulo que atua sobre as informações contidas no *Page Repository* é o *Collection Analysis*. Esse módulo cria um índice chamado *Utility*, o qual traz algumas informações importantes sobre as páginas, como por exemplo, tamanho e número de figuras inseridas. Em outras palavras, a arquitetura de indexação apresentado por [ARASU et al., 2001] funciona obedecendo o princípio de dicionário de dados. Isso porque um determinado termo pode corresponder a uma página ou uma coleção de páginas, as quais contêm esse termo. Todavia, nem sempre as respostas que trazem as páginas relacionadas a um termo estão corretas. Esse é um problema típico dos mecanismos da segunda geração.

Por fim, o módulo *Query Engine* é responsável por processar as consultas efetuadas pelos usuários. Basicamente, esse módulo trabalha diretamente com os repositórios de índices (*Indexes*) e de páginas (*Page Repository*).

Dependendo da quantidade de informação retornada como resposta à consulta do usuário, é necessário um mecanismo de classificação dessa informação. Nesse ponto se faz importante o uso do módulo de *Ranking*, o qual cria uma estrutura de classificação por grau de importância referente à informação.

2.3 Anotação Semântica

O conceito de anotação teve sua criação baseada na evolução e nas necessidades da computação. Não se apresenta como algo que caminhou paralelamente com o desenvolvimento dos computadores, porém emergiu em virtude da troca do objetivo em relação ao uso desses computadores.

Logo no seu surgimento, o cenário da computação fora voltado para o processamento único e exclusivo de dados. A imagem que se tem dos computadores daquela época são de máquinas grandiosas com o intuito de executar cálculos que, até então, eram de extrema complexidade para os humanos (muitos deles continuam sendo

até hoje). Em outras palavras, tinham-se grandes processadores que se resumiam a ferramentas de cálculo.

Com o passar do tempo, a computação evoluiu e, em uma proporção distinta da maioria das demais áreas da sociedade. Aquelas máquinas antigas, cujo propósito era de apenas efetuar cálculos, passaram a adquirir outros aspectos, como por exemplo, estruturas de armazenamento, maior capacidade de processamento, organização, recuperação e troca de informações, baseadas principalmente pelo avanço das redes de computadores. Isso se justifica frente à evolução dos *hardwares* que proporcionaram aos computadores uma ruptura de paradigma, onde deixaram de ser ferramentas de cálculo e passaram a ser ferramentas de comunicação [BAX , 2001].

Essa transformação, em relação ao novo paradigma, tornou-se mais evidente, visto que, ocorreu juntamente com o descobrimento da Web. Esse fato corroborou, de forma efetiva, para o estabelecimento de canais de comunicação entre as partes que trocariam informações.

Todavia, para o perfeito funcionamento do tráfego de informações, estas necessitavam de padrões que permitiriam a todas as partes publicar e recuperar informações de uma forma compreensível para todos. Com isso, surgiram as linguagens de marcação que, desde aquela época, servem para permitir que qualquer tipo de informação seja organizada sob uma marcação e, posteriormente, seja entendida por qualquer outro mecanismo desde que esse mecanismo tenha acesso à marcação.

Aqui cabe uma ressalva sobre marcação e anotação semântica: marcações são *tags* usadas em documentos e têm o intuito de criar metadados semi-estruturados que permitem a interoperabilidade de informação entre diferentes aplicações. Já anotações semânticas se constituem através da utilização de marcações para representação de alguma informação usando, por exemplo, ontologias de domínio. Em [UREN et al., 2005] o autor aborda a diferença entre os dois conceitos onde, ao mostrar um documento contendo *tags*, o chama de marcação. Por outro lado, para se referir às informações que o documento representa, o mesmo autor as chama de anotações semânticas.

Ainda na mesma publicação o autor comenta que anotações que referenciam ontologias são anotações semânticas. Isso reforça o conceito de que anotações

semânticas compreendem a marcações atreladas a estruturas de representação do conhecimento, como OWL, por exemplo.

A marcação é uma entidade totalmente separada do conteúdo que se deseja representar. Diz-se separada, pois representam informações distintas. O conteúdo, de um texto, por exemplo, traz informações sobre saúde, ciência, esportes, etc. Já a informação embutida em uma marcação é específica para permitir que mecanismos, como processadores de texto, por exemplo, possam estruturar a apresentação de um documento.

Por exemplo, documentos, do pacote *Microsoft Office* [OFFICE, 1998], *Openoffice* [OPENOFFICE, 2002 a], ou até mesmo uma página da Web (um arquivo .hmtl), são organizados sob a estrutura de marcações. Estas servem para definir informações, como: (*i*) quem é o proprietário do documento; (*ii*) instantes de criação e alteração; (*iii*) no caso dos pacotes de escritório, informações que definem o tamanho da fonte, cor, tipo de letra, título, etc.

As marcações podem ser implícitas ou explícitas ao usuário. Considerando os mesmos pacotes de escritório mencionados anteriormente, os quais são do tipo WYSIWYG[1], não apresentam qualquer tipo de marca que referencie alguma configuração do usuário. Já os editores de *Tex*[2] e *Latex*[3], amplamente utilizados no meio acadêmico, exigem do usuário certo grau de conhecimento sobre as marcações que configuram o *layout* de um documento, por exemplo.

Ainda sobre marcações, segundo [BAX , 2001] é possível classificá-la em dois diferentes tipos: Procedimental e Descritiva. A marcação Procedimental é um modelo um pouco mais antigo que não organiza a forma de representação da informação e seu documento é apenas compreensível por máquinas. Um exemplo desse tipo de marcação pode ser verificado na Figura 3.

---

[1] São editores que configuram o documento exatamente da forma que o usuário o está construindo. A sigla significa *What You See Is What You Get*.

[2] Processador de texto utilizado amplamente para a produção de textos matemáticos e científicos por causa de sua alta qualidade tipográfica.

[3] Conjunto de macros para o processador de textos Tex.

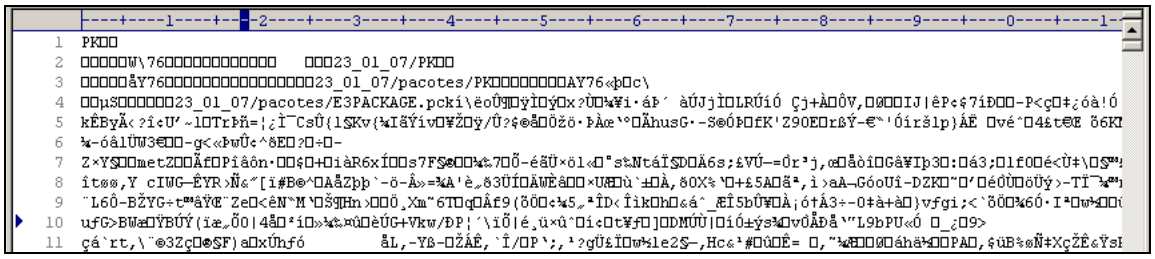

Figura 3 - Exemplo de um arquivo com extensão .zip e sua estrutura interna de marcação.

Percebe-se que na marcação Procedimental o conteúdo é totalmente ilegível para o usuário comum. Somente mecanismos apropriados, como processadores de texto, compiladores, entre outros. são capazes de decodificar tal conteúdo e apresentá-lo corretamente para o usuário final.

A marcação Descritiva é diferente da anterior, pois se baseia na utilização de *tags* como marcas. É importante ressaltar que nesse tipo de marcação, a informação de fato fica separada dos dados que estruturam o documento. Nela, as *tags* não fazem parte do documento e apenas definem informações que podem compreender a formatação do texto, ou ainda, que podem tornar legíveis para uma máquina de busca, como por exemplo, as informações dispostas em um documento. O pacote de escritório *OpenOffice*, ao contrário de suas versões mais antigas, atualmente disponibiliza seus arquivos construtores de um documento na forma de *tags* [OPENOFFICE, 2002 b], as quais se classificam como marcação Descritiva.

A Figura 4 apresenta um exemplo simplista de uma marcação Descritiva. Em um primeiro momento, tal exemplo pode ser associado a um documento XML, por exemplo. De fato, foi baseado nesse tipo de marcação que o XML foi definido e criado.

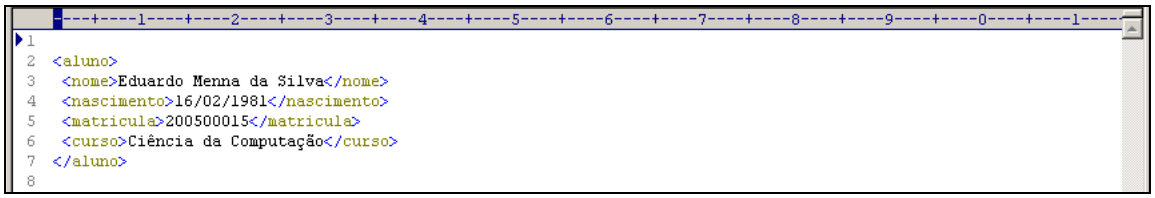

Figura 4 - Exemplo da estrutura interna de um arquivo baseado em marcação Descritiva.

O XML é uma das linguagens de marcação mais conhecidas e amplamente utilizada nas aplicações da Web atual. Todavia, o XML somente alcançou tal importância devido ao refinamento de outras linguagens de marcação que as sucederam. O primeiro exemplo nessa ordem cronológica é o SGML.

Segundo [EDWARDS, 1997] SGML foi criada com o intuito de definir uma linguagem de marcas para a representação de informações em texto. É tida como uma linguagem independente, pois não necessita de um conjunto pré-definido de marcas. Nela, é possível definir as próprias marcas, fato esse que a classifica como uma linguagem auto descritiva.

No entanto, por ter essa característica de não ter pré-definições, qualquer conjunto de marcas criado sobre SGML precisa carregar uma especificação formal de suas definições. Essa especificação é o que constitui o DTD [LI; LI, 2004], que valida os termos usados dentro da marcação SGML.

O DTD é uma estrutura que define como um documento SGML está formatado. Alguns autores colocam que o DTD trabalha como uma gramática formal que define como as marcas devem ser interpretadas.

Ainda segundo [EDWARDS, 1997], o SGML foi projetado e passou a ser adotado há cerca de 30 anos. Porém, em meados da década de 80, os pesquisadores passaram a encontrar dificuldades na manipulação do SGML. Isso porque cada um poderia criar suas marcas, o que inviabilizava a compatibilidade entre aplicações. Pensando em contornar tal problema, um grupo de pesquisados do CERN, dentre eles Tim Berners-Lee, proporam a criação da *World Wide Web* juntamente com uma linguagem de marcas com pré-definições estabelecidas. Nascia então o HTML.

O HTML é uma evolução do SGML, porém trabalha com um grupo de *tags* pré-definidas. Com isso, tornou-se possível que diferentes destinos passassem a compreender e compartilhar um único tipo de estruturação e representação de informação. Por tornar implícita a existência do DTD, o HTML passou a ser amplamente adotado e corroborou para a popularização da Web.

A divulgação da Web permitiu que muitas pessoas compartilhassem informações, sejam elas através dos *sites* espalhados por todo o mundo, ou então dentre as mensagens

de correio eletrônico. Dessa forma, as informações contidas na Web são de fácil acesso e estão ao alcance de qualquer usuário.

Porém, essa crescente abundância de dados trouxe um problema, pois há uma quantidade muito grande de informação disponibilizada, dificultando que o usuário encontre o que busca. Da maneira como a Web foi projetada e está sendo utilizada atualmente, só é compreensível pelos humanos. E estes não têm capacidade de processar uma quantidade enorme de informação em um curto período. Um exemplo disso são as consultas aos motores de busca. Quando estes retornam milhares de ocorrências de um determinado termo pesquisado, os usuários acabam por avaliar a relevância das primeiras páginas retornadas. É idéia então é: por que não deixar essa tarefa para uma máquina?

Do modo como a Web esta organizada atualmente, nem as máquinas conseguem suprir todas as necessidades. Nesse caso, os pesquisadores perceberam que era necessário criar uma nova estrutura de organização da informação a fim de permitir que as máquinas (motores de busca) pudessem executar as suas tarefas sobre essa nova estrutura. Logo, os pesquisadores optaram por desenvolver o XML, linguagem de marcação que evolucionou a forma de representação de informação.

O XML é um acrônimo para *Extensible Markup Language*. Seu primeiro objetivo foi de simplificar a utilização do SGML. Segundo [HARRUSI, 2006], é o formato padrão de representação e compartilhamento de informação na Web, com um alto poder explicativo de seus metadados na forma de elementos e atributos.

Já para [DEITEL et al., 2003], XML é uma tecnologia para criação de linguagens de marcação que descreve, de maneira estruturada, dados de qualquer tipo. De qualquer forma, todas as definições falam acerca de uma linguagem de marcação que pode ser usada para construir definições, formas de representação, de algo que está na Web, ou até mesmo fora dela.

O XML tem sua sintaxe muito parecida com o HTML. Ambas são linguagens de marcação e trabalham com *tags* como definições de estruturas internas. Porém, diferentemente do HTML, XML não define um número limite marcações. A utilização das *tags* fica a critério do usuário, podendo ele criar novas estruturas jamais utilizadas em outros documentos [BERNERS-LEE, 2000].

Nesse aspecto, XML apresenta uma liberdade muito grande. Não existe nenhuma restrição quanto à construção de seus documentos. A única premissa define que, para se entender uma marcação gerada aleatoriamente deve existir um componente receptor familiarizado com os termos utilizados na marcação gerada.

A partir disso é possível criar mecanismos capazes de interpretar uma informação qualquer representada por um documento XML. Isso permite que computadores possam exercer a tarefa, que até então cabia aos usuários, de entender e interpretar uma informação que está sendo representada através de um arquivo XML.

O XML foi responsável por um salto de qualidade na Web. Sua flexibilidade a projetou como sendo o ponto de partida para a idealização da Web 2.0. Isso é justificado pelo fato de que o XML permitiu que informações fossem trocadas a partir de componentes que, até então, não conseguiam estabelecer um canal comum de comunicação. O XML se tornou um formato padrão para troca de informações na Web.

A partir do instante em que a computação proveu mecanismos avançados de representação do conhecimento os pesquisadores se atentaram para o fato de que seria possível agregar, a esses sistemas de computação, formas bem mais elaboradas de descrições de conceitos sobre qualquer objeto. Sendo assim, foi trazido da filosofia o conceito de Ontologia a fim de inserir semântica às informações existentes dentro desses sistemas de computação.

2.4 Ontologia

O termo Ontologia vem do grego *ontos* + *logoi*, que significa "*conhecimento do ser*". Segundo [FERREIRA, 2006], Ontologia é "*é a parte da filosofia que trata da natureza do ser, da realidade, da existência dos entes e das questões metafísicas em geral. A ontologia trata do ser enquanto ser, isto é, do ser concebido como tendo uma natureza comum que é inerente a todos e a cada um dos seres*".

A princípio, as áreas gerais de filosofia e computação não apresentam muita relação. Por outro lado, a computação iniciou, e passa até os dias de hoje, por uma fase

de adaptação do conceito de diversas áreas para a sua representação. Um grande exemplo disso é a Inteligência artificial, subárea da computação. A IA, como é conhecida, traz conceitos da filosofia, biologia, química, etc. para ser representados dentro da computação. No caso da Ontologia, também ocorreu esse processo de adaptação.

A Ontologia é utilizada na computação com o intuito de criar uma estrutura que seja capaz de descrever um objeto, componente pertencente à computação. A Ontologia, nesse caso, faz o papel de um metadado que descreve um determinado elemento. O objetivo principal dessa ligação entre essas áreas distantes é prover uma formalização quanto à capacidade de descrição de um objeto. Sendo assim, uma Ontologia deve trazer a essência de algo, na sua descrição mais minuciosa possível.

A união dessa estrutura de Ontologia à computação, e mais especificamente à Web Semântica, colabora na tarefa de representação do conhecimento. Isso porque, a partir da existência de uma Ontologia, os objetos alocados pela Web podem ser associados aos conceitos que melhor os descreverem em sua essência. Todavia, a Ontologia puramente dita não pode ser aplicada à computação. É necessário criar outra estrutura que seja capaz de permitir a construção e formalização das Ontologias. É nesse instante que se faz importante a existência das linguagens de marcação voltadas às Ontologias. Ao definir uma linguagem de marcação voltada a Ontologias, defini-se também uma linguagem de representação do conhecimento, a qual faz uso de marcações.

Cabe ressaltar que, para o Capítulo 2, as observações feitas em relação à Ontologia e suas linguagens de representação do conhecimento dizem respeito ao escopo teórico e filosófico. A descrição e justificativa computacional para esses conceitos são feitas no Capítulo 3.

2.4.1 Tipos de anotação

Para a representação do conhecimento, seja utilizando ou não o conceito de Ontologia, tem-se a proposta de diversas linguagens de representação do conhecimento. Aqui, serão abordadas, rapidamente, as mais conhecidas e aplicadas.

Segundo a definição em [HERMAN et al., 2004], RDF (*Resource Description Framework*) é um padrão baseado em XML que serve para descrever recursos existentes na Web, *Intranets* ou *Extranets*. O RDF é formado a partir das técnicas de XML e URI, onde URI é usada para identificar todo e qualquer objeto. O conjunto de URIs permite a construção de sentenças sobre algum recurso. Essas sentenças descrevem o recurso propriamente dito, as propriedades desse recurso e os valores para cada uma das propriedades. Esse grupo formado por uma sentença é definido como tripla-RDF, a qual consiste em sujeito, predicado e objeto [DAVIES et al., 2003]. Sendo assim, um recurso representa um sujeito, as propriedades são predicados e os valores para cada propriedade são os objetos.

Contudo, o RDF apresenta algumas lacunas em aberto. Apesar de prover o modelo e a sintaxe para descrever recursos, ele por si próprio não permite a definição destes recursos. Nesse caso, fez-se importante a concepção de uma outra técnica denominada de RDF Schema (*Resource Description Framework Schema*).

Segundo [BRICKLEY et al., 2004], RDF Schema faz a definição do vocabulário de um domínio particular para um recurso RDF. Por exemplo, a partir dos vocabulários RDFS é possível criar a descrição de qualquer objetivo, desde um livro de auto-ajuda até padrões de serviços para a Internet. RDFS define propriedades que podem ser usadas por instâncias RDF em um dado domínio, além de especificar as classes pertencentes a cada recurso.

Para [HEFLIN; HENDLER, 2000], SHOE (*Simple HTML Ontology Extension*) é uma linguagem de representação do conhecimento baseada em Ontologia e projetada para a Web. Essa linguagem foi proposta como sendo uma extensão do HTML e foi projetada antes mesmo do XML e RDF. De acordo com [HEFLIN, 2001], SHOE tem sua sintaxe definida como uma aplicação de SGML que estende o DTD do HTML. Ainda para [HEFLIN, 2001], SHOE separa descrições de termos, denominados de parte ontológica, das assertivas, conhecidos como a parte de instância. A parte ontológica de

SHOE permite a definição de categorias, que podem ser comparadas às classes no paradigma de orientação a objetos, e aos conceitos na lógica.

Outra linguagem de representação do conhecimento denomina-se OIL, a qual é um acrônimo para "*Ontology Inference Layer*", ou ainda, "*Ontology Interchange Language*". Segundo [HARMELEN; HORROCKS, 2000], OIL objetiva ser uma linguagem de representação e inferência para Ontologias. O intuito dessa linguagem é sanar o problema de falta de expressividade e semântica formal do RDF. Para isso, utiliza a extensão RDF Schema, camada essa que provê semântica formal e suporte ao raciocínio [HORROCKS et al., 2000].

DAML (*DARPA Agent Markup Language*) é uma outra alternativa para linguagem de representação do conhecimento voltada para a Web. De acordo com [PAGELS, 2006], o objetivo da DAML é desenvolver uma linguagem e também ferramentas que facilitem a concepção da Web Semântica. Assim, propõe uma linguagem simples com poder de expressar definições de classes RDF mais sofisticadas do que as permitidas pelo RDFS. Essa característica é bem semelhante à linguagem OIL, apresentada anteriormente. Talvez isso justifique porque essas duas linguagens se uniram em um esforço único para criar uma nova linguagem de representação, chamada DAML+OIL.

A linguagem de representação do conhecimento para Ontologias com maior aceitação chama-se OWL (*Web Ontology Language*). Segundo [MCGUINNESS; HARMELEN, 2004], OWL é projetada para ser usada por aplicações que necessitem processar e compreender informações contidas nas páginas Web, as quais, do modo que são apresentadas hoje em dia, somente são entendidas por seres humanos.

OWL é capaz de interpretar certos conteúdos da Web, os quais não são suportados pelas demais linguagens XML, RDF e RDF Schema. Para isso, se utiliza de um vocabulário adicional junto com uma semântica formal. A OWL está dividida em três sub-linguagens, as quais serão comentadas na Subseção 3.3.2.

2.5 Considerações do capítulo

Esse capítulo faz um apanhado cronológico sobre os mecanismos de recuperação. O objetivo foi apresentar os primeiros modelos de representação do conhecimento juntamente como os mecanismos capazes de recuperar esse reconhecimento. Acerca desses mecanismos, foram apresentados seus modelos de recuperação, além de um diagrama completo de uma arquitetura de busca que é seguida em praticamente todos os mecanismos.

Em um segundo momento, retomou-se a discussão em torno dos modelos de representação do conhecimento e se traçou um histórico que mostrou as formas de representação através das tecnologias (linguagens) que foram surgindo em virtude das necessidades dessas representações.

Por fim, mostrou-se a necessidade da associação de conceitos da filosofia à computação, a fim de dar maior poder de descrição aos objetivos computacionais. Além disso, é feita uma pequena introdução conceitual sobre os tipos de linguagens de anotações semânticas mais populações na atualidade.

3 Web Semântica

Este capítulo visa fazer um apanhado geral sobre o tema onde está circunscrito esse trabalho. Serão apresentados os motivos que implicaram na concepção dessa nova Web, bem como os benefícios trazidos pela nova abordagem de compartilhamento de informação, usando a rede mundial de computadores. A estrutura na qual se baseia a Web Semântica também será comentada, juntamente com o aprofundamento de cada camada que a compõe.

Por fim, serão apresentados exemplos bem sucedidos de aplicação dos conceitos de Web Semântica na atualidade, além de apontar os principais obstáculos que ainda a impedem de se expandir e tornar-se uma realidade em aplicações para usuários leigos.

3.1 Idéia Geral

A Web Semântica foi um conceito gerado a partir de uma necessidade muito evidente dentro da Web atual. A expansão dos recursos de informação na rede mundial de computadores trouxe aspectos extremamente positivos no que diz respeito ao compartilhamento de dados. Por outro lado, a estrutura organizacional não acompanhou o avanço no que diz respeito ao compartilhamento desses dados.

Com isso, a Web passou a apresentar um grande paradoxo. De um lado, a capacidade de registrar todo o tipo de informação passível de ser usada por qualquer pessoa - assim como uma grande enciclopédia, e de outro, a dificuldade em como procurar essas informações e poder disponibilizar aos usuários.

Para amenizar o impacto da dificuldade gerada, várias abordagens foram propostas a fim de simplificar a tarefa das pessoas que utilizam a Web. Em meados da década de 90, mais especificamente em 1995, o *Yahoo* [MANBER et al., 2000] [MCCUE, 1999] criou o primeiro mecanismo de busca de dados para a Web. A

princípio, em virtude do tamanho da rede naquele tempo, os serviços providos pelo *Yahoo* atendiam perfeitamente às necessidades da época.

A partir desse marco, novos serviços de busca começaram a ser criados, dentre eles, o mais popular hoje em dia. O Google [Cusumano, 2005] entrou no mercado com novas idéias revolucionárias quanto aos aspectos de busca, indexação e recuperação das informações espalhadas na Web. Porém a principal questão que se levantou frente a esses mecanismos de busca foi o fato de que não são capazes de lapidar uma consulta e retornar aquilo que realmente importa a um usuário.

Muitos autores, dentre eles [DACONTA et al., 2003] e [ANTONIOU; VAN HARMELEN, 2004], sustentam a idéia de que a Web atual foi projetada apenas para o entendimento dos humanos. A tarefa das máquinas dentro da Web se restringe em procurar e apresentar uma coleção de informações aos usuários. Aos humanos, fica a tarefa mais árdua; selecionar o que realmente interessa.

Nesse sentido, [BERNERS-LEE et al., 2001] decidiu dar uma nova roupagem a web. Em suma, a idéia de [BERNERS-LEE et al., 2001] consiste em transferir a tarefa de seleção das informações, a qual é a mais pesada, para o computador. Assim, evita-se que usuários percam horas a fio selecionando qual documento ou informação lhes serão úteis ou não.

Para tornar esse conceito uma realidade não adianta somente criar novos mecanismos de recuperação de informação. Estes até podem melhorar seus algoritmos de busca ou criar métodos de indexação mais avançados e que ocupem menos espaço. Mesmo assim os mecanismos atuais de busca continuarão compreendendo somente os termos usados dentre as *tags* da linguagem de marcação para a Web (HTML).

Nesse sentido, [BERNERS-LEE et al., 2001] idealizou uma nova estrutura para a Web capaz de fazer com que máquinas passassem a compreender a Web assim como os humanos a entendem. Essa reestruturação é toda baseada em uma peça chave, denominada semântica. O intuito é agregar valores semânticos às informações já existentes na rede. Desse modo, não é necessário partir um processo do zero, o que resultaria numa nova Web. Mas sim, adaptar a coleção de dados atual a esse novo aspecto, o qual se espera que trazer a maior dinamicidade ao processo de busca e recuperação da informação.

Como forma de evitar a reconstrução trabalhosa da Web, um novo conceito emergiu com a missão de fazer essa adaptação. A esse novo conceito dá-se o nome de Metadado, o qual tem o objetivo de descrever qualquer componente ou objeto.

Além disso, [BERNERS-LEE, 1998a] idealizou uma nova arquitetura para a Web, a qual contempla as necessidades básicas para se transformar em Web Semântica. A seguir, algumas dessas necessidades:

- Definir a arquitetura e infra-estrutura para a Web Semântica;
- Estabelecer uma linguagem formal de representação do conhecimento ou conteúdo;
- Disponibilizar informações em linguagens baseadas em Ontologias;
- Criar serviços de inferência, dando maior poder aos mecanismos de busca;
- Desenvolver ferramentas de suporte à Web Semântica;
- Criar políticas de segurança e privacidade na Web Semântica.

Uma vez definida as principais necessidades, [BERNERS-LEE et al., 2001] apresenta em seu artigo mais famoso sobre o tema a nova estrutura da Web Semântica. A Figura 5 reproduz essa estrutura.

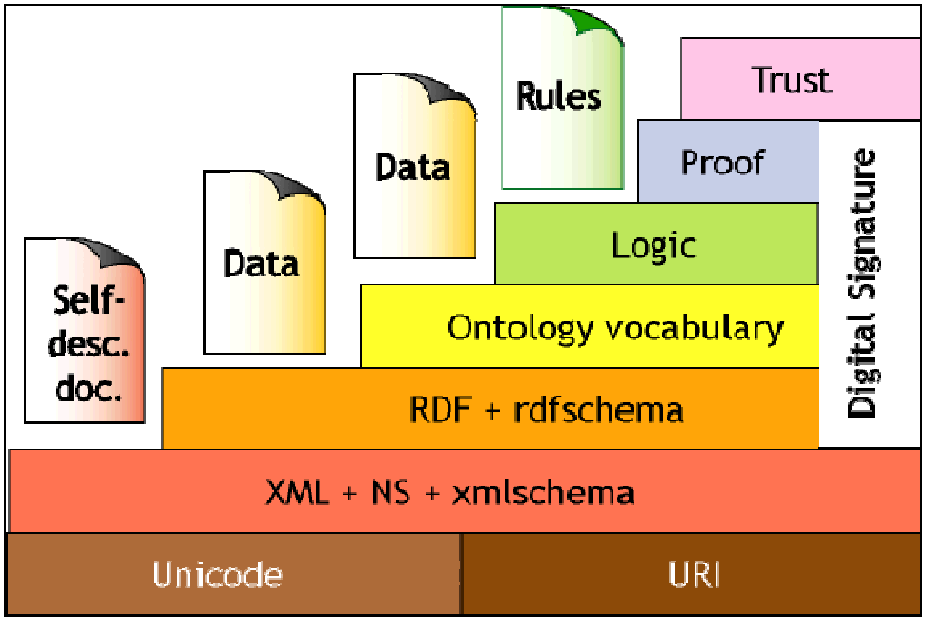

Figura 5 - Arquitetura proposta para recuperação do conhecimento [KOIVUNEN; MILLER, 2001].

A função da camada mais básica chamada de *Unicode* e URI é de definir representações únicas para os recursos que são descritos pela Web Semântica. Para isso, se utiliza do *Unicode*, código único que representa qualquer caracter, e uma URI que contempla a função de um ou mais *Unicodes* para identificar um objeto qualquer.

A segunda camada objetiva definir uma linguagem abstrata que possibilite à Web Semântica a independência em relação às plataformas. Sendo assim, o XML tornou-se fundamental na pilha da Web Semântica, pois possibilita a interoperabilidade de sistemas. Além disso, o XML permite a construção de metadados sobre objetos.

A camada de *RDF + RDF Schem* a acrescenta mais semântica a um documento, com vantagem de não, necessariamente, precisar se referir a sua estrutura [FENSEL, 2003]. O RDF serve para descrever recursos da Web, que devem apresentar um identificar na Web, sendo ele parte específica de um documento ou dados como lugares, pessoas, etc. [KOIVUNEN; MILLER, 2001].

A camada de Ontologia é responsável por trazer um conceito da filosofia à computação. A falta de soluções capazes de captar a semântica das páginas da Web, entre outros problemas computacionais, criou uma demanda de serviços que se ajusta à

aplicação de ontologias [FREITAS, 2003]. Sendo assim, essa camada tem o intuito de colaborar na associação de recursos a seus respectivos metadados descritivos.

A camada Lógica tem como base a Camada *Unicode* URI. Essa camada lógica é responsável pela definição de um mecanismo que como objetivo fazer inferências sobre os dados, ou representações ontológicas.

Já a camada de Prova envolve o processo dedutivo bem como a representação dessas provas em linguagens para a Web, além de suas validações [ANTONIOU; VAN HARMELEN, 2004].

A última camada é responsável por determinar se as provas geradas pela camada anterior são verdadeiras ou não. Enquanto na camada anterior a validação diz respeito à corretude da sintaxe, a camada de confiança é a que define a veracidade da prova.

3.2 Camadas Básicas

A Web Semântica é uma tecnologia ainda não totalmente em aplicação. Todavia, desde a sua concepção até os dias de hoje muitos dos conceitos adotados são fortemente fundamentados e usados como padrões. Um exemplo disso são os conceitos envolvidos nas primeiras três camadas da pilha da Web Semântica.

As três camadas passaram por um longo período de estudo e hoje apresentam definições formais e reconhecidas pelos mecanismos recuperadores de padrões. Por esse fato, as três camadas inferiores são condensadas em uma única seção, pois contêm conceitos já consolidados.

A camada mais básica da Web Semântica é composta pelos componentes URI e *Unicode*. Assim como comentado anteriormente, esses dois padrões foram adotados a fim de definir uma estrutura mínima independente. Dessa forma, a Web Semântica tornou-se uma aplicação sem restrições, pois provê sua própria camada básica que permite sua operação.

Nessa estrutura básica o URI e o *Unicode* apresentam funções fundamentais. O *Unicode* é responsável por mapear as informações da Web Semântica para os

mecanismos atuais da computação. Já o URI é a concatenação esse *Unicode* que faz com que cada componente da Web tenha uma identificação única.

No trabalho de [ZAGO, 2005} é reproduzida uma definição de [BERNERS-LEE et al., 1998] onde, em uma RFC, é feita uma caracterização de uma URI a partir dos componentes:

- Identificador: Um identificador é uma seqüência de caracteres com sintaxe restrita que faz referência a alguma coisa que tenha identidade;
- Uniforme: A uniformidade fornece vários benefícios, como permitir que diferentes tipos de identificadores de recursos sejam usados no mesmo contexto, permitir a introdução de novos identificadores de recursos sem interferir nos existentes e permitir a interpretação semântica uniforme de convenções sintáticas nos diferentes tipos de identificadores;
- Recurso: Um recurso pode ser observado como qualquer documento eletrônico, imagem, serviço ou coleção de outros recursos que possua identidade.

O URI é um identificador único para qualquer tipo de componente. Por exemplo, um arquivo de computador, uma pasta, o caminho para execução de um programa, etc, todos são definidos através de um localizador unido, o URI. No caso da Web, existe um tipo específico de URI chamado de URL (*Uniform Resource Locator*).

3.3 Ontologia

A camada de Ontologia da Web Semântica propõe agregar um conceito importante da Filosofia. O intuito de utilizar Ontologias como forma de organizar o conhecimento capacita os computadores a entenderem o significado dos conteúdos dispostos na Web.

Esse conceito foi adaptado à tecnologia da informação devido à área de Inteligência Artificial, que busca simular e representar o comportamento humano nas mais diversas tarefas [MOURA, 2001]. Com isso, projetou-se a idéia de que os computadores pudessem executar a tarefa de busca e seleção de informações pertinentes a um determinado tema baseada na representação de um conhecimento de senso comum (ontologia). O autor [GÓMEZ; BENJAMINS, 1999] ratifica a idéia de Ontologia como senso comum quando diz: "*Uma ontologia é uma especificação formal e explícita de uma conceitualização compartilhada*".

Ontologias, para a computação, são documentos descritivos sobre algum tema, assunto ou área que podem ser utilizados como referência para a criação de anotações em outros documentos a fim de descrevê-los.

De acordo com [GRUBER, 1993] "*Uma ontologia é uma especificação explícita de uma conceitualização. O termo é tomado da filosofia, onde uma ontologia é uma consideração sistemática da existência. Para os sistemas da inteligência artificial, o que existe é o que pode ser representado. [...] Dessa forma, no contexto da inteligência artificial, pode-se descrever a ontologia de um programa definindo um grupo de termos representacionais. Em tal ontologia, definições associam nomes de entidades no universo do discurso (por ex. classes, relações, funções ou outros objetos, com textos legíveis para pessoas, descrevendo o que os nomes significam, e, os axiomas formais que restringem a interpretação e o uso adequado desses termos) [...]*".

Em uma definição mais sucinta e atual [BIANCHINI et al., 2006] define que Ontologia é a "descrição explícita e precisa de conceitos e relações que existem em um domínio particular".

Toda essa conceitualização trazida da filosofia permite construir modelos descritos sobre algo. Todavia, a computação necessita de uma formalização onde seja possível traduzir o conhecimento e sua forma de representação, para uma linguagem legível aos computadores. Assim, criaram-se diversas linguagens de representação de informação, as quais foram comentadas na Subseção 2.4.1. Duas delas serão aprofundadas a seguir.

3.3.1 DAML+OIL

DAML+OIL é uma linguagem de representação do conhecimento que surgiu a partir da fusão entre DAML (*DARPA Agent Markup Language*) [DAML, 2000] e OIL (*Ontology Inference Layer* ou *Ontology Interchange Language*) [ONTOKNOWLEDGE.ORG/OIL, 2000]. Seu objetivo foi agregar as características mais interessantes de cada abordagem, como por exemplo, a integração de RDF com RDF Schema, característica inerente à DAML; e a lógica descritiva, formalismo de representação do conhecimento que era aplicada pela linguagem OIL.

Mais tarde, a junção dessas duas linguagens foi substituída pela linguagem OWL, a qual é comentada na subseção 3.3.2.

3.3.2 OWL

OWL é a atual linguagem para representação do conhecimento recomendada pela W3C, difundida e usada na construção de Ontologias dentro da Web Semântica. Essa linguagem de representação teve origem a partir do aperfeiçoamento de sua sucessora DAML+OIL. Seu principal objetivo é estender as funcionalidades do RSF Schema, permitindo construções além da semântica básica, a partir da utilização de classes e propriedades.

Segundo especificação em [W3C/OWL, 2004], OWL provê três sub-linguagens, as quais foram projetadas para que comunidades distintas possam adotar cada sub-linguagem de acordo com suas necessidades.

- OWL Lite: Sub-linguagem limitada direcionada a usuários que não precisam utilizar todas as funcionalidades que a linguagem suporta. Basicamente, fornece classificação hierárquica e restrições simples inerentes ao conhecimento descrito sobre a OWL Lite;

- OWL DL: Esta sub-linguagem contempla todas as funcionalidades da OWL, com a garantia de que todas as conclusões retiradas a partir da linguagem sejam computáveis e dentro de um período finito. Em outras palavras, obedecendo ao principio da decidibilidade. OWL DL recebe esse nome, pois se utiliza da lógica descritiva;
- OWL Full: OWL Full é uma sub-linguagem que fornece as mesmas funcionalidades que a OWL DL, porém sem se preocupar com questões computacionais. Segundo os autores [W3C/OWL, 2004], "*em OWL Full uma classe pode ser tratada simultaneamente como uma coleção de indivíduos e como um indivíduo por si mesma. OWL Full permite que uma ontologia aumente o vocabulário pré-definido de RDF ou OWL. É improvável que algum software de inferência venha a ser capaz de suportar completamente cada recurso da OWL Full*".

3.4 Camada Lógica

A Camada Lógica da Web Semântica situa-se no quinto nível da arquitetura descrita por Tim Berners-Lee [BERNERS-LEE et al., 2001]. É uma camada fundamental para a construção da Web Semântica no seu ideal maior. Somente essa camada é capaz de extrair conhecimento não explícito nos documentos em geral e também nas descrições (ontologias) anexadas aos documentos.

A estrutura da camada lógica, seu motor de inferência de regras, será o *kernel* dos *Crawlers* na tarefa de busca de determinadas informações. Sem essa capacidade de inferir regras mais elaboradas os *Crawlers* ficariam restritos as atividades exercidas hoje pelos motores de busca, tais como *Google*, *Yahool* e *All the Web*. Alguns autores questionam se caso a Web Semântica realmente conseguiria ser bem sucedida sem a presença das estruturas lógicas adaptadas aos *Crawlers*.

Outra característica interessante da utilização das regras de inferência para extração do conhecimento em ontologias seria o fato de que, essas regras permitem a

associação de outras ontologias, sendo possível a delimitação de padrões entre os conceitos representados nessas ontologias.

Um exemplo evidente dessa aplicação é mostrado em [HATALA, 2005], onde a máquina de inferência extrai informações dos usuários visitantes de um museu e determinam o perfil para esses usuários de acordo com os artefatos visitados. Através de uma seqüência de visitas a artefatos do tipo "múmia" se pode inferir que o usuário se interessa por cultura egípcia.

Todavia, para que a camada lógica possa exercer suas atividades normalmente é necessária uma série de pré-requisitos, dependências que devem estar bem definidas e resolvidas. A primeira dependência é a existência de uma ontologia (na camada inferior à Lógica) bem estruturada.

Além da existência de uma ontologia bem definida é necessária também a existência de um mecanismo ou *parser* que mapeie informação semântica (OWL ou DAML + OIL) para uma estrutura lógica.

Para a "compilação" da linguagem semântica para estrutura lógica, as Lógicas Descritivas e de Predicados estão sendo usadas como importante ferramenta de auxílio para essa representação. A importância da transformação do conhecimento representado semanticamente pelas ontologias em estruturas lógicas se dá pelo fato de que fica mais simples o trabalho de inferência de regras, também descritas sobre os mesmos aspectos da lógica, pois ambas estarão sob uma mesma estrutura lógica facilitando suas associações. Um pequeno exemplo sobre a atuação da camada lógica é apresentado abaixo. Considerando a Lógica de Predicados:

```
Prof ( x ) → facult ( X )

Facult ( X ) → staff ( X )

Prof ( Michel )
```

Tabela 1 – Exemplo de regras.

É possível deduzir:

```
Facult ( Michel )

Staff ( Michel )

Prof ( X ) → staff ( X )
```

Tabela 2 – Deduções alcançadas a partir das regras definidas.

Esse tipo de aplicação e dedução de um novo predicado é tipicamente encontrado em uma ontologia. Pode ser interpretado como a descoberta de algum conhecimento que estava representado de forma implícita. Porém, assim como o exemplo de Aristóteles citado em [Russel; Norvig, 2003] "Socratis e Imortal" deve-se tomar cuidado com os relacionamentos inesperados ou inconsistências geradas a partir de novos predicados.

Atualmente a W3C caminha para a definição de uma linguagem oficial para trabalhar com regras dentro da Web Semântica. Assim como a OWL, que está sendo adotada como linguagem para Ontologia, a RuleML [HORROCKS et al., 2005] desponta como linguagem para descrição de regras.

3.5 Camada de Prova e confiança

De acordo com a W3C [W3C/SW, 2001], as duas camadas mais superiores estão em processo de estudo e desenvolvimento de experimentos que demonstrem sua efetiva aplicação.

A camada de prova é responsável por executar as regras e normas criadas pela camada anterior a sua denominada camada Lógica. Após a execução dessas regras a camada de Prova repassa suas conclusões a próxima camada chamada Confiança, a qual tem o objetivo averiguar se tais regras e normas podem ser confiáveis ou não. Essa informação deve ser transferida novamente para a camada de Prova, a qual dará prosseguimento a processo, por exemplo, *Crawler* que busca pela veracidade das

informações de um pacote turístico, seus preços, locais de estadia, deslocamento, comparações, etc.

3.6 Linguagens para recuperação de informação

Nesse trabalho foram levantadas algumas abordagens responsáveis pela recuperação de informação em documentos anotados semanticamente. Ao todo foram estudadas as abordagens XQuery, Squish, RQL, SWRL e OWL-API.

A abordagem XQuery [BOAG et al, 2003] é uma linguagem de consulta para documentos XML. Foi desenvolvida frente à flexibilidade de representação do conhecimento sobre arquivos XML. Todavia, apresenta uma série de limitações. O conceito trabalhado em XQuery é de árvore e não de grafos, o que dificulta a tarefa de recuperação de informação em estruturas RDF, pois estas possuem seus próprios modelos de dados. Como exemplo de consulta, permite apenas os valores relacionados a atributos de uma determinada *tag*.

A Squish [MILLER, 2001] é uma abordagem destinada a consultas sobre documentos RDFS. É definida como uma linguagem de simples navegação em grafos RDF. Sua limitação diz respeito ao fato de só responder consultas ligadas a triplas explícitas (aquelas diretamente anotadas). Compreende ao Nível Estrutural da recuperação semântica.

A abordagem RQL [KARVOUNARAKIS, 2003] é uma evolução da Squish e permite consultas sobre todo o conhecimento representado em RDFS (informações explícitas e implícitas). Corresponde ao Nível Semântico para recuperação de informação.

A abordagem SWRL [Horrocks et al., 2005] combina OWL e RuleML. A principal característica da SWRL é estender a OWL através da inclusão de regras da lógica Horn. Tal abordagem visa resolver o problema denominado de Decibility. Segundo [Horrocks et al., 2005] "*Muitas da limitações da OWL derivam do fato de que,*

enquanto a linguagem inclui um rico conjunto de construtores de classes, a linguagem provida para falar sobre propriedades é muito fraca".

Por fim, a última abordagem estudada chama-se OWL-API [HORRIDGE et al., 2007] e é destinada a manipular arquivos OWL. A abordagem tem o mesmo poder de consulta que a abordagem RQL, porém ambas trabalham com linguagens de representação do conhecimento distintas. Enquanto RQL foca anotações em RDFS, OWL-API é voltada para anotações em OWL. A seção 4.8 traz mais detalhes sobre a abordagem OWL-API.

A seguir é apresentado um quadro comparativo entre as abordagens. São comparadas as linguagens de representação do conhecimento que cada uma trabalha e a qual níveis de recuperação estão compreendidas.

Abordagem	Ling. representação do conhecimento	Nível
XQuery	XML	Sintático
Squish	RDFS	Estrutural
RQL	RDFS	Semântico
SWRL	OWL	Semântico
OWL-API	OWL	Semântico

Tabela 3 – Quadro comparativo

3.7 Trabalhos correlatos

O presente trabalho traz a descrição de algumas ferramentas de recuperação de informação. O intuito é apresentar a aplicação da algumas linguagens de consulta comentadas anteriormente sobre algumas ferramentas.

A ferramenta Lucene é uma API para efetuar buscas textuais sobre documentos previamente armazenados. A tarefa é semelhante a maioria dos motores de busca onde é

feita a indexação de documentos na forma textual e posteriormente são efetuadas consultas que procuram por similaridade entre conteúdos.

A ferramenta Exploora é um projeto desenvolvido em meio acadêmico. Seus autores denominam o *Crawler* como um buscador semântico, pois trabalha com um dicionário de sinônimos associado aos termos que são indexados pelo mecanismo. Essa característica permite que usuários tenham uma lista de possíveis significados para cada termo consultado nessa ferramenta.

Sesame é uma arquitetura para indexação e recuperação de informações semânticas em documentos. A representação do conhecimento nesses documentos indexados é baseada em RDFSchema. A fase de recuperação da informação utiliza a linguagem RQL.

O projeto KAON se divide em duas partes. O KAON1 responsável pela recuperação de informação utilizando RDFSchema para descrição de documentos e RQL como linguagem de consulta. Já o KAON2 utiliza a OWL para descrição e OWL/SWRL para efetuar consultas. Um detalhe interessante do projeto KAON é que suas ferramentas não trabalham com a recuperação de informação baseada nas anotações dos documentos, mas sim diretamente no documento de representação do conhecimento (uma ontologia, por exemplo).

A ferramenta Cobse[1] é voltada para a recuperação de informações em documentos corporativos anotados semanticamente. é específico para anotações sobre autoria de documentos. Sua tarefa é analisar a anotação contida no documento e associar a uma ontologia que descreva autores.

Todas as ferramentas comentadas têm características interessantes e cabíveis a determinados cenários. Porém, ao considerar o foco desse trabalho percebe-se que as ferramentas deixam algumas questões em aberto. A ferramenta Lucene é voltada a busca textual em documento, característica essa que inviabiliza sua aplicação em buscas semânticas.

[1] Ferramenta de recuperação de informação semântica desenvolvida dentro do grupo CompSem.

A ferramenta Exploora também não se aplica ao problema, pois não considera documentos que representem o conhecimento (arquivos RDFSchema ou OWL). Apenas trabalha com um dicionário de sinônimos associados aos termos. Já o KAON é uma abordagem voltada a consultas sobre RDFSchema ou documentos em OWL e, a princípio, não associa documentos comuns contendo anotações que referenciem essas ontologias.

A ferramenta Cobse é a que mais se aproxima do problema foco desse trabalho. Porém, essa ferramenta está limitada a apenas considerar a anotação existente no documento e buscar de forma *ipsis litteris* essa anotação dentro da ontologia associada.

Ferramentas	Doc. Anotados	Abordagem	Tipo de Busca
Lucene	Sem anotação	SQL	Tradicional
Exploora	Sem anotação	SQL	Dicionário de sinônimos
Sesame	RDFs	RQL	Semântico
KAON1	RDFs	RQL	Semântico
KAON2	OWL	SWRL	Semântico
Cobse	SQL	OWL	Estrutural

Tabela 4 - Resumo das peculiaridades das ferramentas de recuperação

No intuito de propor uma alternativa para o problema focalizado nesse trabalho, o Capítulo 5 apresenta uma arquitetura que contempla alguns aspectos que foram abordados acima.

3.8 Considerações do capítulo

O presente capítulo abordou a área principal na qual está inserido o trabalho. A Web Semântica foi apresentada e comentada desde a sua concepção até a definição de

uma estrutura interna, a qual está dividida em camadas. Além disso, foram apresentadas as principais linguagens de representação do conhecimento e exemplos de aplicações da Web Semântica, em geral, a problemas do cotidiano.

4 Arquitetura Proposta

O presente capítulo faz a apresentação formal e detalhada da proposta que visa abordar o problema motivador desse trabalho. Os conceitos e idéias, a arquitetura do sistema proveniente do modelo e todos os aspectos tecnológicos que abrangem esse projeto são comentados nesse capítulo.

4.1 Descrição

O Capítulo 2 apresentou uma série de problemas existentes quanto à recuperação de informação na Web. As abordagens existentes e desenvolvidas como forma de contornar esses problemas também foram apresentadas e comentadas.

A intenção dessa arquitetura aqui proposto não é de reproduzir um mecanismo já existente. O objetivo principal é, a partir de uma revisão bibliográfica detalhada, apontar os principais aspectos acerca dos métodos de recuperação de informação. Com isso é possível avaliar a real eficácia dentre todas as abordagens desenvolvidas.

Com essa avaliação, a intenção é propor uma arquitetura que aproveite os métodos mais eficientes de recuperação e agregar novos conceitos provenientes dos estudos, a fim de solidificar uma nova proposta de um mecanismo de recuperação de informação para a Web. É com esse intuito que se desenvolve esse trabalho.

A estrutura básica dessa arquitetura aproveita o conceito apresentado por [ARASU et al., 2001]. No trabalho de [ARASU et al., 2001] está descrito que os mecanismos de busca podem ser divididos em três componentes distintos: A Busca, a Indexação e a Recuperação da informação propriamente dita. A Figura 6 apresenta a arquitetura genérica de [ARASU et al., 2001] adaptada para a presente arquitetura.

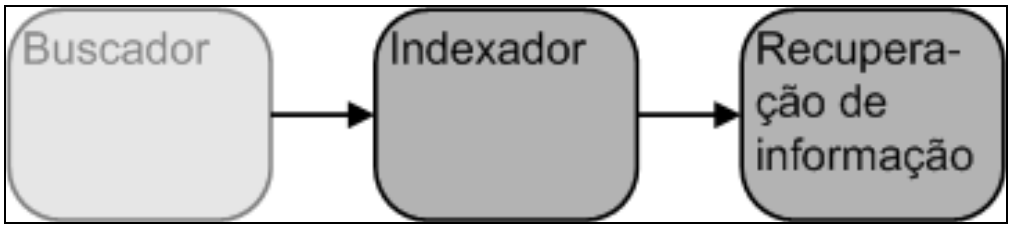

Figura 6 - Arquitetura genérica de uma ferramenta de busca.

A intenção é aproveitar os componentes em destaque na Figura 6 - Indexação de dados e Recuperação das informações. Com isso, parte-se do ponto de que o componente de Busca, normalmente resumido a um *Crawler*, será representado por uma base de dados local. Assim, não haverá um robô que coletará os arquivos a serem pesquisados. Os usuários deverão usar um repositório central para alocar seus arquivos.

O fato da arquitetura considerar que a etapa de busca é restrita a ocorrência de uma base de dados local não implica na incompletude da proposta. Isso porque, os documentos não precisam ser buscados, uma vez que se encontram em um repositório local conhecido. Desse modo, não caberia aqui aplicar um mecanismo buscador, como um *Crawler*, por exemplo, uma vez que já se conhece a localização desses documentos.

A decisão de trabalhar com um repositório local é aplicável não somente ao âmbito acadêmico, mas também à empresas, institutos, entre outros. Isso porque essas corporações trabalham sob o controle de suas *Intranets*, e estas primam pela concentração dos seus bens intelectuais. Assim, apresentam documentos dispostos em locais fixos, os quais são repositórios de materiais fonte de informação.

Os arquivos manipulados pela ferramenta implementada sobre a arquitetura aqui proposta foram confeccionados por um aplicativo específico, chamado Open Office. Isos se justifica, pois o CompSem já possui uma ferramenta de anotação para os arquivos desse aplicativo. Desse modo, todos os arquivos do repositório têm encapsuladas as anotações, ou metadados [LASSILA, 1998], em suas propriedades. Essas anotações trazem classes e instâncias que serão úteis na tarefa de indexação executada pelo Módulo Indexador. A estrutura das anotações será abordada mais adiante.

A partir dos arquivos e suas respectivas anotações, o componente de Indexação da arquitetura terá a tarefa de capturar esses arquivos do repositório e aplicar um *Parser* sobre eles. Esse *Parser* é responsável por entender a anotação e registrar em um banco de dados todas as informações contidas nessa anotação. A estrutura do banco de dados também será comentada adiante.

Após a inserção dos arquivos no repositório central de dados e dos registros dos desses dados de uma anotação por parte do componente Indexador em um banco relacional, a próxima tarefa fica a cargo do componente Recuperador de Informação. Esse componente é responsável por atender a consultas de usuários e retornar os arquivos que estiverem de acordo com as consultas desses usuários. Isso é feito através da utilização do banco de dados relacional e também de consultas efetuadas sobre as ontologias.

As seções a seguir comentam as características individuais dos componentes que fazem parte da ferramenta de recuperação de informação proposta nesse trabalho.

4.2 Busca (crawling)

Nessa proposta, considera-se que os documentos que compõem o repositório de dados não serão trazidos por um robô ou *Crawler*. Nesse caso, parte-se do princípio que os dados já foram carregados e estão armazenados em uma base.

O local onde se encontram os arquivos com suas anotações é chamado de repositório. Esses arquivos existentes nessa base são colocados pelos usuários que desejam concentrar, em um único lugar, as informações de uma empresa ou instituição.

Com isso, o módulo seguinte da estrutura, o de Indexação, considera apenas os arquivos que já constam em uma base de dados local. A forma de que como esses arquivos foram coletados não interessa à etapa de indexação.

4.3 Indexação

O módulo de indexação tem papel fundamental para a composição da ferramenta de recuperação de informação. Nessa arquitetura, a etapa de Indexação engloba o repositório que, a princípio, é parte do módulo de Busca. Isso porque, como já foi comentado anteriormente, o Buscador não faz parte do escopo do trabalho. Considera-se que os dados já estão disponíveis localmente.

A arquitetura proposta trabalha com dois tipos de fonte de informação: o repositório de arquivos e o repositório de ontologias. A base de arquivos é responsável por armazenar todos os documentos inseridos pelos usuários para que sirvam, futuramente, como fonte de consulta. Obrigatoriamente, todos os documentos inseridos nessa base deverão conter uma anotação. Os detalhes sobre a anotação de um documento serão comentados posteriormente.

O repositório de ontologia é responsável por armazenar as ontologias, as quais estarão relacionadas com os documentos. A existência desse repositório se faz importante, pois o módulo de recuperação, ao efetuar uma busca a partir da consulta de um usuário, precisará relacionar um registro da tabela da base de dados a algum termo, instância ou conceitos de uma determinada ontologia. Sendo assim, é necessário acessar a ontologia e efetuar buscas sobre as estruturas representadas internamente.

Uma vez tendo organizados os repositórios de arquivos e ontologias, o módulo indexador pode exercer normalmente sua tarefa. Nessa etapa, a Indexação é composta por alguns sub-módulos que auxiliam na tarefa de traduzir as informações existentes nas anotações para uma tabela relacional. A Figura 7 apresenta a arquitetura do módulo Indexador.

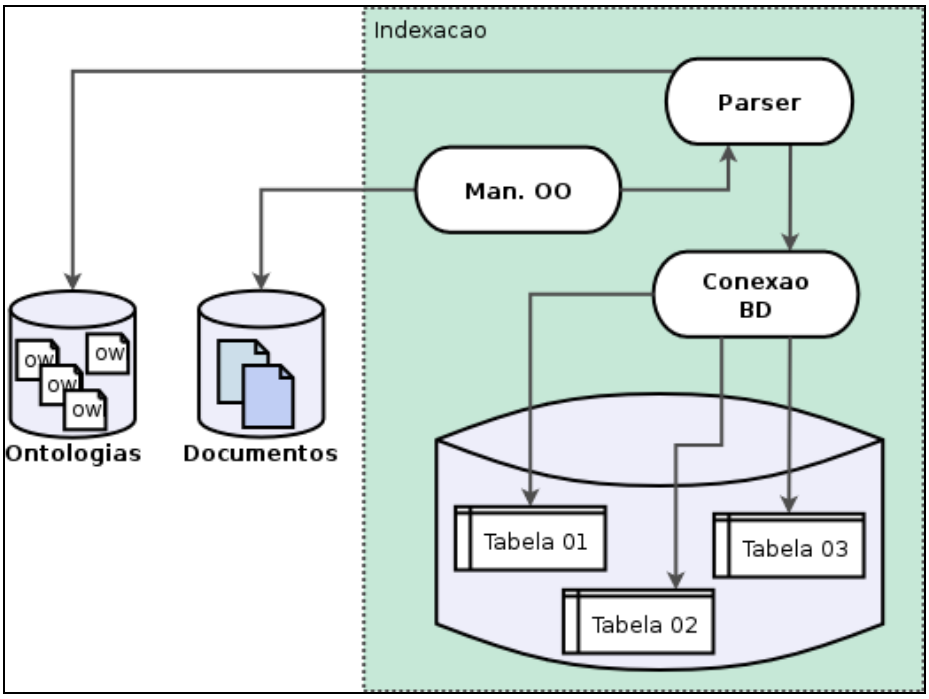

Figura 7 - Estrutura do Módulo Indexador.

Na Figura 7 os repositórios são mostrados fora da etapa de Indexação. Dentro desta etapa é apresentado o sub-módulo do *Parser*, o qual é responsável por ler uma anotação de um documento e capturar informações, como por exemplo:

- Namespace: informação essa que está inserida no cabeçalho da anotação;
- Ontologia: é uma URI que determina onde se descreve a ontologia relacionada com o documento corrente. Essa ontologia possui internamente a definição das classes e instâncias que provavelmente o documento deve referenciar;
- Arquivo: caminho que aponta para o local onde o arquivo está armazenado dentro do repositório de documentos;
- Classe: traz a URI que determina onde tal classe se faz descrita;
- Instância: traz também a URI que define onde está descrita essa instância.

Os demais sub-módulos chamados *Man. OO* e *Conexão BD* exercem as funções, respectivamente, de manipulação dos arquivos do aplicativo Open Office e execução de diretivas do banco de dados (conexão e execução de comandos SQL).

4.3.1 Analisador – Parser

Para fazer com que as informações contidas no repositório de dados possam ser entendidas pelo mecanismo geral de recuperação de informação é necessário saber do que trata cada um desses documentos. Para isso, existe um *Parser* responsável por traduzir, de modo semântico, o conteúdo que um documento tem internamente.

Tal conteúdo semântico é representado por uma anotação que vem embutida nas propriedades dos arquivos do repositório. A Seção 4.6 apresenta como se dá a anotação dentro de um documento. Sendo assim, existe um componente na arquitetura com o intuito de extrair a anotação do documento e entregar seu conteúdo ao *Parser*. Esse componente chama-se Manipulador de Arquivos OO e situa-se entre o módulo Indexador. O documento que terá sua anotação analisada e posteriormente manipulada situa-se no repositório. A Figura 7 apresenta tal componente estabelecendo uma ligação entre a fonte de informação (documento) e o módulo analisador.

Após o fornecimento da anotação, o *Parser* é responsável por extrair informações da ontologia, classes e instâncias que um determinado documento traz. Considerando a Figura 8 tem-se os três pontos principais analisados pelo *Parser*. O primeiro representa a informação que define onde está a ontologia referenciada ao documento. O segundo ponto principal é a definição das classes que são instanciadas nessa anotação. E, por último, tem-se a indicação das instâncias propriamente ditas.

```
<owl:Ontology rdf:about="file:/C:/edumenna/ontologia_educacao.owl">
</owl:Ontology>                                                          1

<owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Aluno">
</owl:Class>                                                             2

<rdf:Description rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Eduardo_Menna_da_Silva">
  <rdf:type>                                                                                  3
    <owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Aluno">
    </owl:Class>
  </rdf:type>
</rdf:Description>
```

Figura 8 - Exemplo do código de uma anotação.

Esse processo de análise sobre a anotação se faz importante, pois a partir dele é possível coletar as informações que povoam a base de dados em um segundo momento. Os itens 1, 2 e 3 apresentados na Figura 8, a qual apresenta um resumo de uma anotação, são diretamente passados para o banco de dados. Com isso, é possível construir uma rede de relações entre os componentes: documento, anotação, classe, instância e URI. Na Subseção 4.3.2 o diagrama esquemático do banco de dados é comentado.

4.3.2 Banco de Dados

O papel do banco de dados será de: (*i*) guardar as informações dos documentos do repositório (arquivos *OpenOffice*); (*ii*) guardar as anotações existentes dentro desses documentos; e, (*iii*) também registrará um apontador para a ontologia na qual a anotação se baseia. Na Figura 9 é apresentado o diagrama do banco de dados.

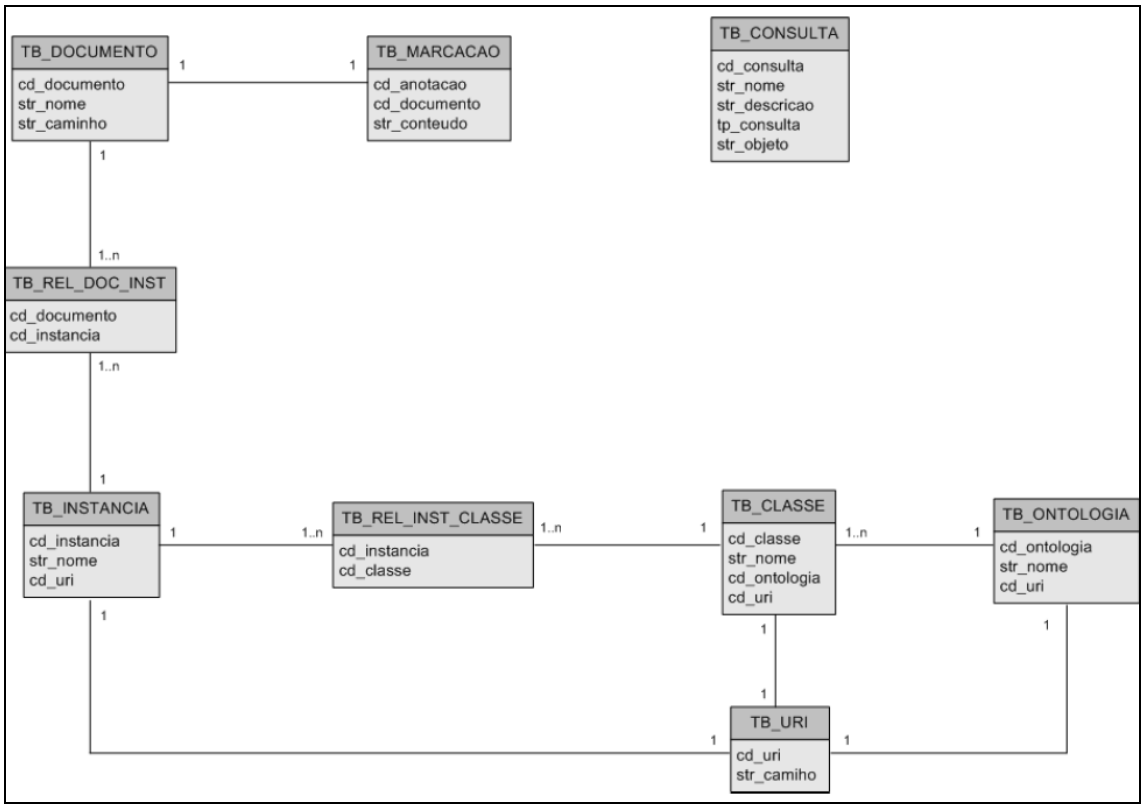

Figura 9 - Diagrama esquemático da Base de Dados.

No total, a base de dados proposta para esse trabalho é composta de seis tabelas. A tabela elementar dessa arquitetura é a *TB_DOCUMENTO*. A essa tabela, todas as outras terão relacionamentos diretos ou indiretos. Na *TB_DOCUMENTO* há informações sobre o arquivo existente dentro do repositório de arquivos. Os principais dados dessa tabela são: o nome do documento, podendo este ser repedido dentre os arquivos existentes no repositório; e, um caminho que define onde se encontra o arquivo. O caminho é único, pois só pode referenciar um documento.

À tabela *TB_DOCUMENTO* estão relacionadas as *TB_ANOTACAO* e *TB_INSTANCIA*. A primeira é responsável por apenas manter o registro de uma anotação, na íntegra, de um documento. Já a tabela *TB_INSTANCIA* é uma das mais importantes da arquitetura. Nela, ficam os registros de cada instância encontrada dentro das anotações. Essa tabela terá papel fundamental para o processo de recuperação de informação.

Após a definição da *TB_INSTANCIA* tem-se a descrição de outras duas tabelas relacionadas a ela. A primeira, chamada *TB_CLASSE*, é responsável por armazenar os registros das classes que uma anotação traz. Normalmente, as ferramentas que geram anotações não deixam explícitas a quais classes pertencem as instâncias. Porém, nesse trabalho, a ferramenta utilizada para gerar a anotação tem a característica de encapsular a definição da classe de uma instância. Com isso, não é preciso analisar a ontologia para descobrir a que classe pertence uma instância. Isso reflete em um ganho de performance. Todavia, as informações sobre a ferramenta de anotação serão comentadas no Capítulo 5.

A segunda tabela relacionada com a *TB_INSTANCIA* chama-se *TB_URI*. Essa tabela determina onde está definida uma instância propriamente dita. E não só a instância, mas também uma classe ou ontologia. No caso das instâncias, um registro da *TB_URI* referencia onde está descrita uma instância. A princípio a informação da URI de uma instância poderia ser um campo descritivo da *TB_INSTANCIA*. Porém, optou-se por separar essas informações, pois instâncias de ontologias distintas, ou classes distintas, podem apresentar o mesmo nome. Assim, somente pela URI é possível diferenciar uma das outras. A Figura 10 apresenta um caso onde instâncias possuem o mesmo nome.

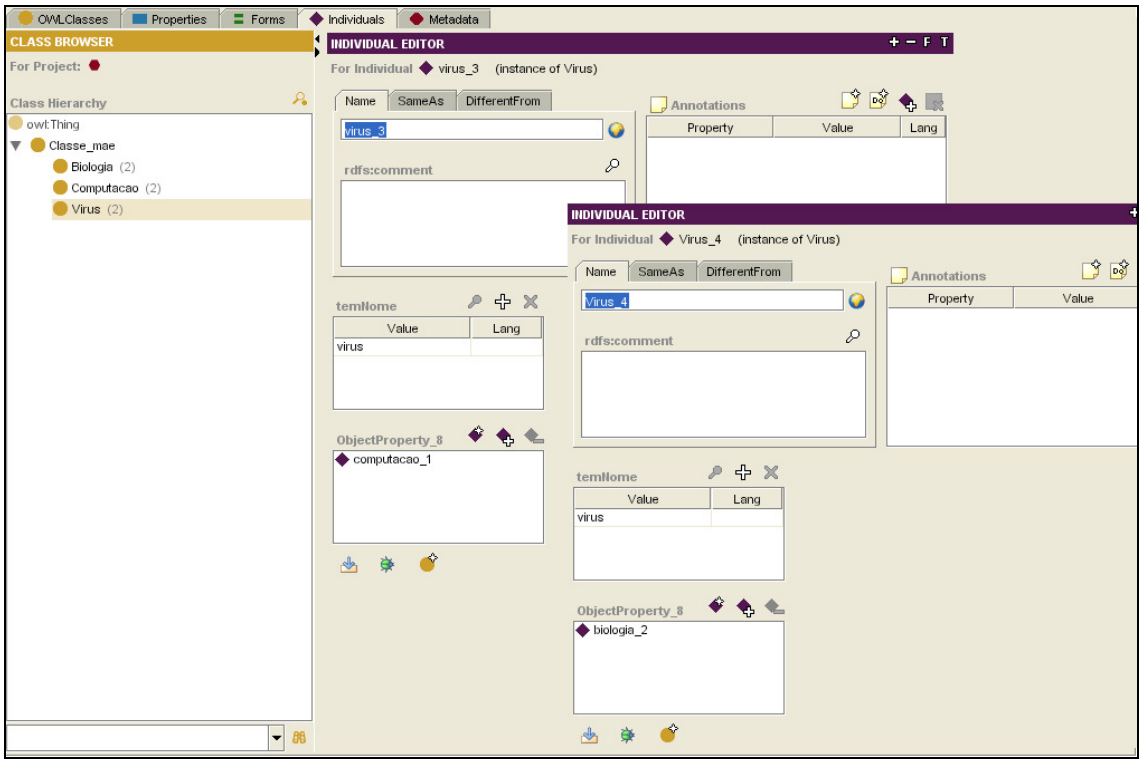

Figura 10 - Exemplo de instâncias com mesmo nome, porém pertencentes a classes distintas.

Na Figura 10 percebe-se que a existência de duas instâncias com nomes iguais e, além disso, com o nomes das classes a quem pertence, também iguais. Nesse caso, a URI é a única forma de distinguir cada um desses componentes.

Outra tabela que faz parte do diagrama do banco de dados chama-se *TB_ONTOLOGIA*. Essa tabela apresenta relações com *TB_CLASSE* e também *TB_URI*. A importância da relação com a segunda tabela se justifica pelo mesmo motivo no qual *TB_INSTANCIA* e *TB_CLASSE* também se relacionam com a *TB_URI*. Outro detalhe que fica visível nesse diagrama é a ligação entre as tabelas *TB_INSTANCIA*, *TB_CLASSE* e *TB_ONTOLOGIA*. Com isso, toda instância é referenciada a uma classe e esta por sua vez é referenciada a uma determinada ontologia.

O banco de dados terá papel fundamental para responder às consultas dos usuários. Por exemplo, nas situações em que as buscas dos usuários são questões superficiais (perguntas como: "*quais documentos tem instâncias de Universidade?*"), o banco de dados possui autonomia suficiente para retornar respostas corretas às consultas sem, necessariamente, ter que acessar a ontologia para responder qual questão.

Por outro lado, quando as buscas são mais avançadas, o banco de dados serve de base para se acessar uma ontologia quando a consulta é executada. Considerando a anotação mostrada na Figura 1 com as instâncias de cão e gato. Se a consulta for do tipo: "*retornar todos os documentos que falam sobre animais mamíferos"*. Nesse caso, o banco de dados, por si só, não tem capacidade de responder à consulta. Mas a partir do banco é possível descobrir que existe uma classe que referencia Mamíferos. Essa referência é feita através do *slot* "*temNome*" existente na ontologia e descrito como um campo na tabela Classe do diagrama da Figura 9. No segundo exemplo fica evidente a importância da definição de um *slot* que sirva de identificador dentro da ontologia.

4.4 Recuperação de Informação

As Seções 4.3 e 4.4 comentaram toda a parte que provê a análise das informações e disponibiliza ao terceiro componente de uma ferramenta de busca de informação. Essa última etapa chama-se Recuperação de Informação e depende diretamente do sucesso dos componentes descritos anteriormente.

A etapa de Recuperação tem uma inteligência da indexação. Esta é responsável por fornecer uma base de consulta sólida e com a maior capacidade possível de representação das informações contidas nos documentos provenientes do repositório de dados. Já a Recuperação utiliza de técnicas de consulta sobre as informações representadas na base, seja essa base o próprio banco de dados ou ainda uma ontologia que representa conceitos.

A Recuperação pode ser descrita em quatro sub-módulos que se complementam e sustentam toda a etapa de Recuperação. A Figura 11 apresenta a Recuperação de informação.

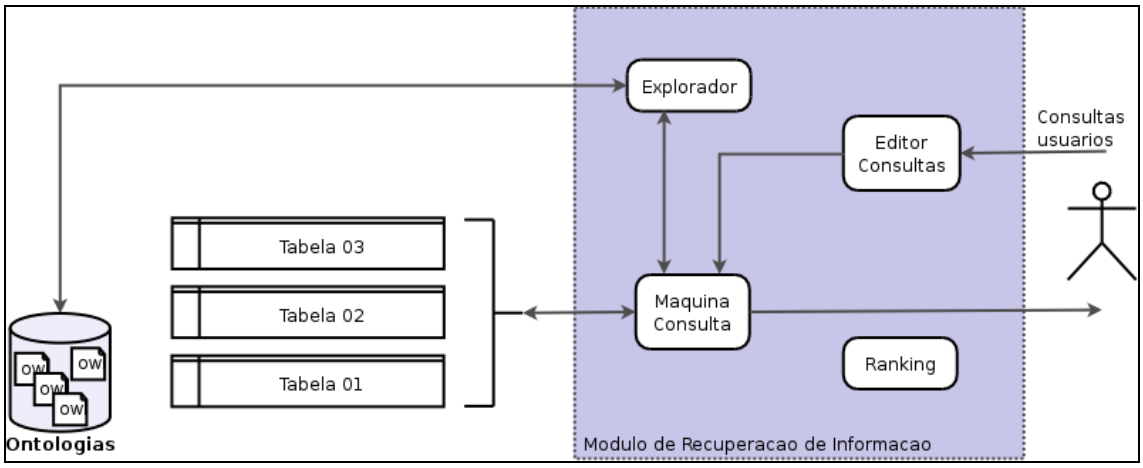

Figura 11 - Etapa de Recuperação de Informação.

O primeiro sub-módulo, que recebe uma seta externa, é o Editor de Consultas. Ele é responsável por prover uma interface de consulta onde usuários podem fazer requisições de informações à ferramenta de recuperação. Esse sub-módulo sozinho não atende as solicitações de quem utiliza a ferramenta e necessita pesquisar por documentos. Para isso, é necessária a existência da Máquina de Consulta e do sub-módulo denominado Explorador.

A Máquina de Consulta fica responsável por receber requisições do Editor de Consulta as requisições e efetuar a busca dentro da base de dados provida pela etapa de Indexação. É importante salientar que, dependendo da superficialidade das pesquisas, a Máquina de Consulta é capaz de responder corretamente as requisições dos usuários. Por exemplo, "*deseja-se saber quais são os documentos que contém anotação do Autor José Silva?*". Como apresentado anteriormente na Seção 2.2 esse tipo de consulta de Nível Estrutural é facilmente encontrado, pois só relaciona um registro de um autor na tabela de instâncias ao documento no qual se encontra relacionado.

Todavia, não só essas consultas servem para atender as requisições dos usuários. Se for considerada a situação onde é construída a seguinte pesquisa: "*Retornar todos os documentos escritos por professores das Instituições de Ensino Superior*". Na recuperação Sintática não é possível responder a essa pergunta, pois é necessário ter a representação do conhecimento descrito em uma ontologia. Nesse caso, faz-se

necessária a utilização do terceiro sub-módulo, o qual tem contato direto com o repositório de ontologias.

Para esse último exemplo de pesquisa, o sub-módulo Explorador executa a tarefa de recuperar a informação e, conseqüentemente, justifica sua importância. Nesse caso, o Explorador vai verificar se os filhos da Instância XX da classe Instituição apresentam outros filhos. Se existindo, verifica também se esses filhos apresentam classes ou sub-classes que representem Professores. Encontrando as instâncias dessas sub-classes, pode-se dizer que estas instâncias, por herança, são também instâncias da classe mãe chamada Instituição. Observa-se a ontologia da Figura 12 e a anotação de um documento de exemplo na Figura 13.

```
<rdf:Description rdf:about="file:/C:/ontoTeste/ontoInsituicaoProfessor.owl#Jose_da_Silva">
  <rdf:type>
    <owl:Class rdf:about="file:/C:/ontoTeste/ontoInsituicaoProfessor.owl#Professor">
    </owl:Class>
  </rdf:type>
</rdf:Description>
```

Figura 12 - Anotação de Professor baseado na ontologia.

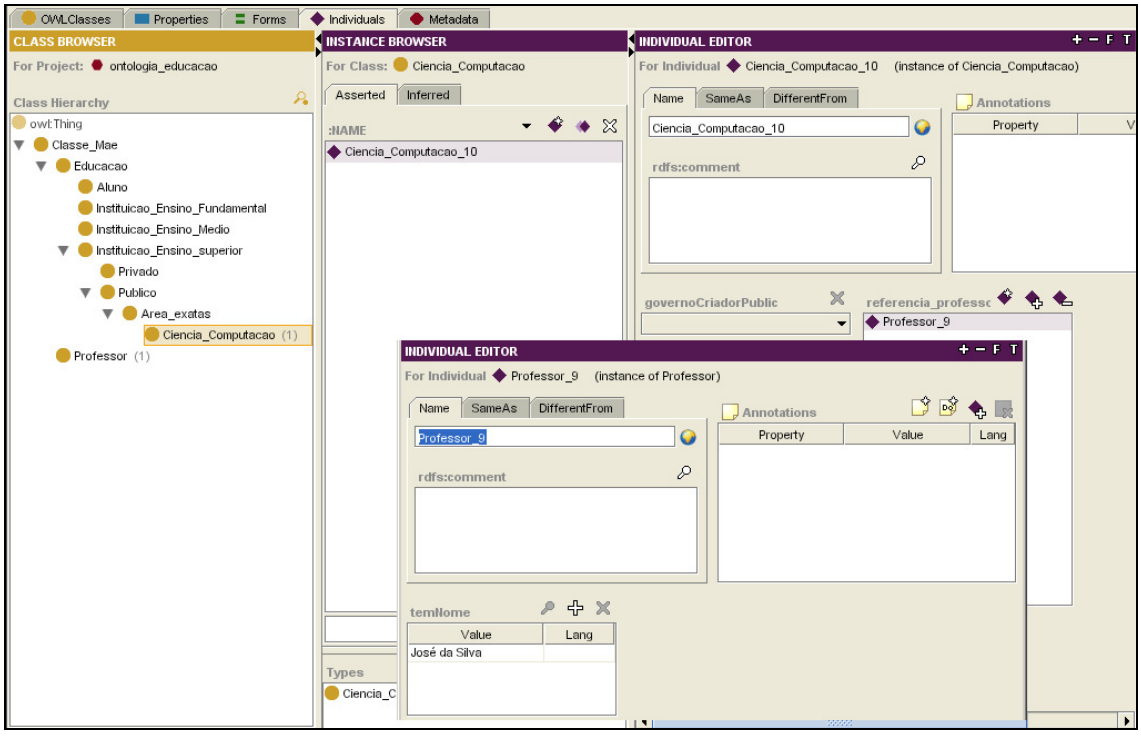

Figura 13 - Exemplo de Ontologia.

Percebe-se na Figura 12 que o documento apresenta a anotação de professor, porém esta não referencia diretamente a Instituição Superior. O que se vê é a anotação de professor como instância de uma Classe denominada *ciencia_da_computacao*. Em um primeiro momento se responderia que esse documento não é de um professor de ensino superior.

Todavia, em virtude da existência do sub-módulo Explorador, é possível descobrir, analisando a ontologia da Figura 13, que essa instância pertencente à classe Ciência da Computação, por herança é também uma instância da Classe Instituição de Ensino Superior. Isso porque a classe Ciência da Computação é filha da classe Área de Exatas e, a qual é filha da classe Pública e esta por sua vez, é filha da classe Instituição de Ensino Superior.

O quarto sub-módulo é responsável por classificar as respostas ás consultas efetuadas pelos usuários. Esse módulo cria uma lista decrescente de acordo com o maior grau de importância dos documentos retornados baseados na consulta. Todavia, esse sub-módulo não faz parte do escopo dessa arquiteutura e está somente representado na Figura 11.

4.5 O papel da Ontologia dentro da Arquitetura

Até esse momento nada foi comentado a respeito da Ontologia e sua contribuição para o trabalho. Isso foi feito de modo proposital, pois existem algumas ressalvas a serem feitas sobre a utilização dessa forma de representação do conhecimento. A seção atual comenta alguns detalhes.

A Ontologia é o cerne da Web Semântica. Ela é capaz de representar de forma muito organizada o conhecimento sobre alguma área, tema, linguagem, repartição, etc. Entretanto, a construção de uma ontologia não faz parte desse trabalho. Nessa arquitetura, parte-se do pressuposto que a ontologia já está construída e que também já

foi utilizada por algum usuário na definição das anotações relacionadas aos arquivos existentes no repositório.

Contudo, existe um detalhe importante imposto à construção de uma ontologia. A presente arquitetura é baseado na proposta de [ARASU et al., 2001], o qual define uma estratégia para a formulação de uma ontologia. Segundo [ARASU et al., 2001], ao se construir uma Ontologia é necessário definir um campo identificador para cada instância das classes de uma determinada Ontologia.

Esta política é bem aplicada para Ontologias a serem construídas. Mas se for considerada a situação onde os documentos anotados utilizam ontologias já existentes, a medida de atribuir um *slot* identificador para cada instância se tornaria inviável. Isso porque seria necessário acrescentar um *slot temNome*, por exemplo, a cada classe de uma determinada Ontologia. Além disso, seria também necessário atribuir um valor para cada Instância que herdaria o *slot temNome* de sua respectiva classe.

Assim, [GLONVEZYNSKI, 2005] apresenta uma alternativa que simplifica a criação de um *slot* identificador. A idéia consiste em criar uma Classe Mãe genérica onde todas as outras classes serão filhos da classe citada. Essa Classe Mãe tem um *slot* único, o qual recebe o nome de *temNome*, por exemplo. Desse modo, todas as classes filhos terão, por herança, um *slot* chamado *temNome* também. Isso soluciona o transtorno em que seria necessário alterar todas as classes e atribuir valores aos *slots* das instâncias.

A Figura 14 mostra a criação de uma classe Mãe e a estrutura hierárquica da Ontologia após a utilização do conceito de [GLONVEZYNSKI, 2005].

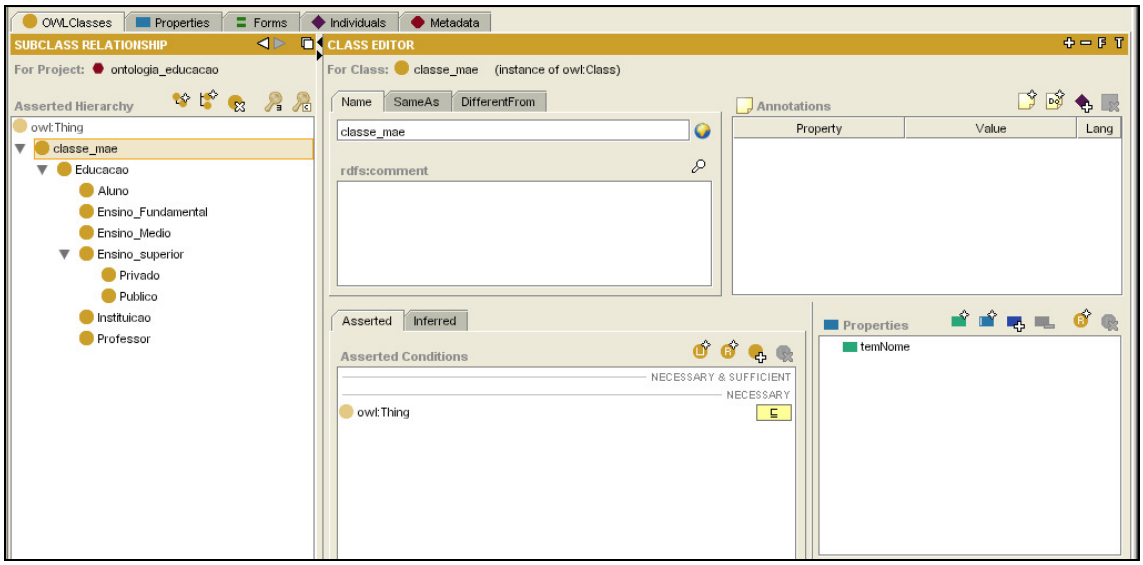

Figura 14 - Exemplo de criação de uma classe mãe genérica com o *slot temNome*.

A característica de toda instância ter um nome é necessária, pois permite que as consultas efetuadas sobre a base possam relacionar um registro de uma tabela a um indivíduo único da Ontologia. O elo de ligação que relaciona um registro da base a uma instância de uma ontologia será a URI.

Além disso, o *slot* identificador aqui rotulado como *temNome* também terá outro papel fundamental. O de servir como dicionário de sinônimos para uma instância. A Figura 15 representa tal situação.

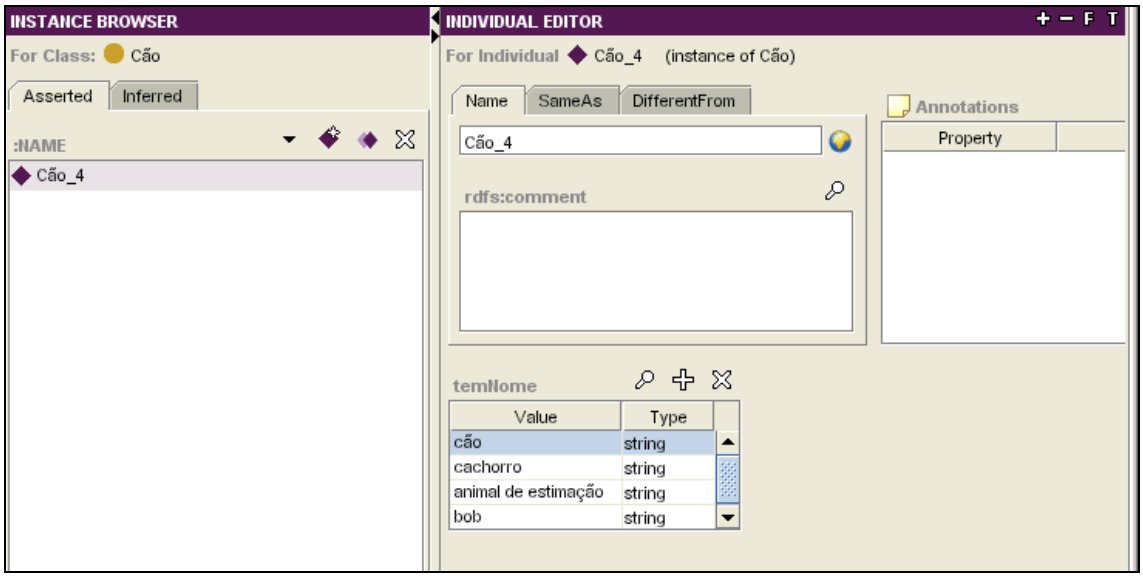

Figura 15 - Exemplo de um dicionário de sinônimos.

Na Figura 15 a instância de *Cão* pode ser vários nomes. Isso faz com que essa instância possa ser consultada através de várias outras palavras, o que define um conjunto, ou lista, de sinônimos da Instância. Os documentos que contém anotações de *Cão*, por exemplo, podem ser pesquisados e retornados como resultado a partir de palavras como: canino, cachorro, animal de estimação, bob, etc.

4.6 Fonte de informação

Serão utilizados como fonte de informação e fonte de consulta todos os arquivos do aplicativo *OpenOffice*. Esses arquivos são utilizados dentro do laboratório Compsem como materiais de referência do grupo. Esses arquivos estarão à disposição para consultas em um repositório central definido em algum servidor de arquivos.

A fonte de informação semântica, que é aquela que contém a anotação do documento com as instâncias e definições da ontologia, ficará dentro do próprio arquivo *OpenOffice*. Os arquivos desse aplicativo possuem vários campos para descrever suas propriedades. A anotação ficará dentro de um campo chamado Descrição.

A Figura 16 apresenta como ficará a estrutura do arquivo *OpenOffice* acrescido da anotação semântica.

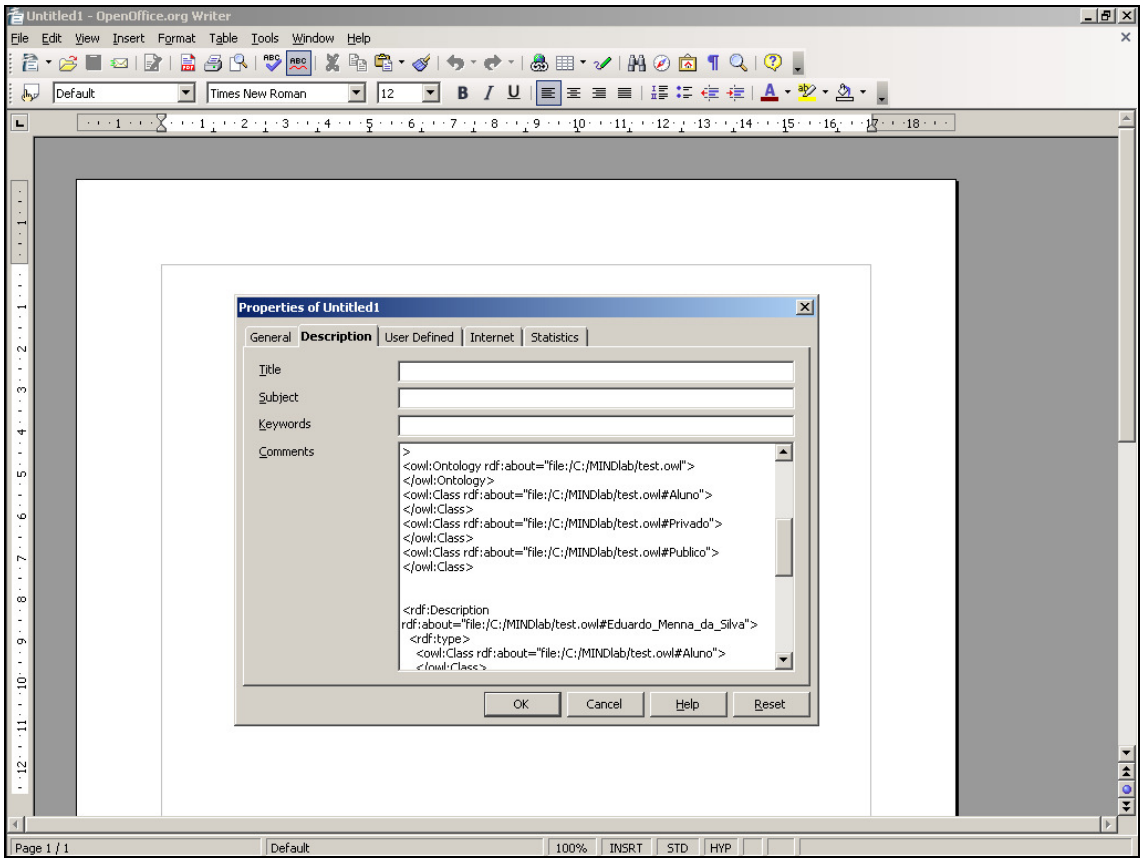

Figura 16 - Local onde constará a Anotação Semântica.

4.7 Anotação Semântica

A definição de uma anotação se faz importante, pois existirá um *parser* específico que a analisará. O primeiro ponto a ser definido é o tipo de linguagem de representação do conhecimento que será utilizada. Dentre as mais conhecidas, como por exemplo, XML puro, RDF e OWL, a opção mais aconselhada é a linguagem OWL. Isso se justifica pelo fato de que OWL é a linguagem padrão adotada pela W3C para a descrição de recursos na Web Semântica. Além disso, a OWL possui maior capacidade de representação de informações semânticas que a RDF ou RDF Schema.

O fato de a ferramenta proposta no trabalho manipular esse tipo de documento descritivo já torna a ferramenta interessante, pois, se a linguagem OWL comporta maior

capacidade de representação, maior será a capacidade do mecanismo de recuperação em indexar e buscar as informações que serão, futuramente, consultas pelos usuários.

Outro ponto de importante definição é a estrutura que terá a anotação de um documento. Para isso, há um estudo acerca de algumas ferramentas geradoras de anotação a fim de avaliar qual melhor se adapta a proposta desse trabalho.

Foram analisadas algumas ferramentas de anotação, como por exemplo, a ontoMat, a ferramenta de anotação desenvolvida por [GLONVEZYNSKI, 2005] para dar suporte à Cobse, Smore[1], entre outras. Dentre as ferramentas geradoras de anotação, optou-se por mesclar a característica de duas ferramentas. A Smore e a ferramenta de anotação desenvolvida para a Cobse.

A escolha se justifica por dois motivos. O primeiro é o fato de como a Smore constrói a anotação do documento. Se for analisada a estrutura da anotação poderá ser visto que, além de anotar as instâncias de cada classe, a Smore faz a anotação também da classe a qual pertence essa instância. A existência da classe na anotação permite a utilização de uma base de dados mais elaborada (a base será comentada adiante).

Já a opção de também utilizar a ferramenta de anotação do Cobse se justifica, pois tal ferramenta é destinada a criar anotações para documentos do pacote de escritório *Open Office*. Cabe ressaltar que a construção de uma ferramenta de anotação não faz parte do escopo desse trabalho. Assim, serão adotadas anotações já definidas dentro do padrão requerido para esse trabalho.

Como já foi dito anteriormente, a maioria das ferramentas de busca trabalha com uma tabela única, genérica, responsável por relacionar documentos a termos ou instâncias. Considerando uma situação hipotética, caso a ferramenta de busca quisesse incluir na tabela uma classe da instância, seria necessário acessar a ontologia ligada a essa instância e descobrir a qual classe pertence. Ou ainda, considerando que a base adotada é genérica e existe uma consulta que busca por instâncias de uma determinada classe.

[1] Ferramenta que permite anotar documentos HTML a partir de ontologias OWL.

Portanto, sendo essa a base genérica e não existindo ligações entre instâncias e suas classes, seria necessário, novamente, acessar a ontologia para descobrir a classe e depois voltar para o banco de dados e efetuar a consulta. Essas duas considerações resultariam em uma grande perda de performance por parte da ferramenta de busca.

A fim de ganhar performance, optou-se pela junção da anotação gerada pela ferramenta Smore ao modelo de anotação proposto em [GLONVEZYNSKI, 2005]. Ela provê todas as informações necessárias para construir uma base de dados mais completa o que reflete diretamente no processamento das consultas. A seguir é apresentado o código de uma anotação completa, a qual será analisada pela ferramenta de recuperação de informação aqui proposta.

```
<!DOCTYPE owl [ <!ENTITY owl  "http://www.w3.org/2002/07/owl#">
<!ENTITY xsd  "http://www.w3.org/2001/XMLSchema#"> <!ENTITY rdf
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"> <!ENTITY rdfs
"http://www.w3.org/2000/01/rdf-schema#"> ]> <rdf:RDF
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 xmlns:unnamed="http://www.owl-ontologies.com/unnamed.owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:test="file:/C:/edumenna/ontologia_educacao.owl#"
>
<owl:Ontology rdf:about="file:/C:/edumenna/ontologia_educacao.owl">
</owl:Ontology> <owl:Class
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Aluno">
</owl:Class> <owl:Class
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Privado">
</owl:Class> <owl:Class
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Publico">
</owl:Class>


<rdf:Description
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Eduardo_Menna_da_Silva">
  <rdf:type>
    <owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Aluno">
    </owl:Class>
  </rdf:type>
```

```
</rdf:Description>

<rdf:Description
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Universidade_Catolica_de_Pelotas">
  <rdf:type>
    <owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Privado">
    </owl:Class>
  </rdf:type>
</rdf:Description>

<rdf:Description
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Universidade_Federal_de_Santa_Catari
na">
  <rdf:type>
    <owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Publico">
    </owl:Class>
  </rdf:type>
</rdf:Description>

</rdf:RDF>
```

A ferramenta Smore trabalha sob os conceitos de triplas RDF. Ou seja, toda sentença representada dentro da anotação traz consigo uma referência à instância, tipo e classe. Isso pode ser verificado na seguinte declaração:

```
<rdf:Description
rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Eduardo_Menna_da_Silva">
  <rdf:type>
    <owl:Class rdf:about="file:/C:/edumenna/ontologia_educacao.owl#Aluno">
    </owl:Class>
  </rdf:type>
</rdf:Description>
```

A *tag* ***<rdf:Description>*** abre a tripla RDF fazendo uma referência a instância que está envolvida. No caso, a instância ***Eduardo_Menna_da_Silva***. O segundo componente da tripla RDF é representado pela *tag* ***<rdf:type>*** que define o tipo de relação existente entre a instância e a futura classe. Essa por sua vez é definida dentro da *tag* ***<owl:Class***. Sendo assim, fica definida dentro dessa estrutura RDF que a instância ***Eduardo_Menna_da_silva*** é do ***tipo*** que pertence a classe ***Aluno***.

Além disso, a anotação gerada pela ferramenta Smore pode ser dividida em três partes para uma melhor análise. Na primeira parte da anotação há o cabeçalho com a definição de todos os *namespaces* usados nessa anotação. Esse cabeçalho é semelhante para todas as anotações geradas a partir de ferramentas de anotação.

A segunda parte da anotação gerada pelo Smore é o diferencial no qual resultou pela opção de utilizar tal ferramenta. Observa-se que existe, logo após a definição da URI da ontologia, a definição das classes que possuem instâncias anotadas para um determinado documento. As classes definidas pelas URIs permitem a construção de uma base de dados mais abrangente do que a base genérica.

Por fim, a terceira parte da anotação é também semelhante às demais ferramentas. Nelas são anotadas as instâncias de classes que constam em um determinado documento.

Cabe ressaltar que a Ferramenta constituída juntamente a esse trabalho não inviabiliza a utilização de outras anotações, além da gerada pela Smore. Para isso, é preciso apenas adaptar o módulo analisador da anotação (*parser*) para a interpretação de outros modelos de anotação de documentos.

4.8 OWL-API

Para a recuperação de informações implícitas às existentes nas anotações a arquitetura descrita nesse capítulo utiliza uma biblioteca capaz de fazer inferências sobre ontologias do tipo OWL. Tal biblioteca chama-se OWL-API.

A OWL-API é uma biblioteca Java. Se resume a um conjunto de métodos Java implementados para a OWL W3C. Atualmente, a OWL-API está em sua versão 1.1 e encapsula as sub linguagens da OWL: Lite, DL e Full OWL. A OWL-API não está restrita somente a manipulação de documentos OWL. Permite também o trabalho com outros tipos de linguagens de representação, como o RDF, por exemplo.

A OWL-API foi escolhida para ser usada nesse trabalho, pois são métodos implementados em Java, a qual é a mesma linguagem de programação escolhida para o

desenvolvimento da ferramenta que valida a arquitetura. Além disso, OWL-API apresenta todos os requisitos necessários para manipulação de arquivos OWL. A forma de como a API trabalha com anotações é simples e apresenta rápida capacidade de resposta a consultas.

A API entende que uma anotação OWL é um objeto. Esse objeto recebe o nome de *OWLOntology* e provê vários acessos a seus axiomas. Não está limitado a axiomas lógicos, a exemplo da OWL-DL, mas sim inclui anotações que são elas mesmas representadas por axiomas. A partir de uma ontologia que é entendida como um objeto, a forma de manipular a ontologia e encontrar as informações dentro dela se torna tal qual é a manipulação de objetos criados dentro de estruturas de classe. A seguir é apresentado um exemplo de criação de um objeto *OWLOntology*:

```
1       try {
2           // Create the manager
3           OWLOntologyManager man = OWLManager.createOWLOntologyManager();
4
5           // Load the ontology
6           OWLOntology ont = man.loadOntologyFromPhysicalURI(URI.create(p_uri_ontologia));
7           System.out.println("Loaded: " + ont.getURI());
```

Figura 17 – Exemplo de código para criação de um objeto *OWLOntology*

A linha 3 apresentada na Figura 17 traz a criação de um objeto que é capaz de utilizar todos os métodos que a API provê. Já a linha 6 traz a criação do objeto que recebe a ontologia. Para isso, é preciso informar a URI da ontologia. A partir desde instante, o objeto *ont* permite a manipulação da ontologia. Desde criação de classes e instancias até consultas por informações explícitas e implícitas. Mais informações sobre a API podem ser encontradas em [HORRIDGE et al., 2007].

4.9 Considerações do capítulo

O capítulo atual abordou a arquitetura proposta neste trabalho, o qual é fruto de uma extensa revisão bibliográfica que objetiva solucionar um problema até então não contemplado pelas demais ferramentas de recuperação de informação.

5 Ferramenta implementada para validação da arquitetura

O capítulo 5 tem o objetivo de apresentar a ferramenta desenvolvida com a finalidade de validar a arquitetura do trabalho proposta e comentada no capítulo anterior. Aqui serão apresentadas as funções existentes na ferramenta através de telas extraídas do sistema.

5.1 Ferramenta MarSinWebs

A ferramenta MarSinWebs, acrônimo para Marcação e Recuperação Semântica na Web, foi a interface desenvolvida com o intuito de validar a arquitetura proposta nesse projeto. Tal ferramenta foi desenvolvida em Java [DEITEL; DEITEL, 2005].

A opção pela escolha da linguagem de programação Java se justifica pelo fato de que o MarSinWebs é um subprojeto de um projeto guarda-chuva denominado InWebs, o qual é um *framework* desenvolvido pelo grupo *Compsem* da UFSC. Esse *framework* é um apanhado de técnicas inerentes a Web Semântica e agrupa projetos responsáveis pela construção e manipulação de ontologias, criação de regras de negócio sobre ontologias, anotação de documentos, recuperação de informação e raciocínio sobre ontologias.

Ficou definido pelo grupo *ComSem*, durante a fase de especificação e arquitetura e do projeto geral, que todos os subprojetos desenvolvidos seguiriam a plataforma Java. Dessa forma, assim foi feito com o projeto descrito nesse trabalho.

A implementação da arquitetura apresentada no capítulo 4 segue o padrão de desenvolvimento MVC – *Model, View, Control*. Assim, as camadas de persistência de dados, regras de negócio e apresentação do conteúdo ao usuário ficaram independentes e repercutiram em uma melhor organização estrutural do trabalho.

A ferramenta MarSinWebs foi desenvolvida para a Web. Com isso, o usuário pode trabalhar de forma remota. Ademais, toda a aplicação é persistida no servidor, evitando alguns inconvenientes impostos por aplicações desenvolvidas sob a estrutura cliente-servidor.

Para permitir a aplicação voltada à Web foi necessário utilizar o publicador de conteúdo Web específico do Java, o qual é chamado de *TomCat*. Todas as páginas da ferramenta foram desenvolvidas de acordo com a tecnologia *JSP – JavaServer Pages*. A seção 5.2 irá mostrar visualmente a ferramenta.

5.2 MarSinWebs – forma visual

A tela inicial da ferramenta é acessada através do URL que encaminha o usuário à aplicação hospedada no publicador TomCat. A tela inicial da ferramenta é apresentada na Figura 18.

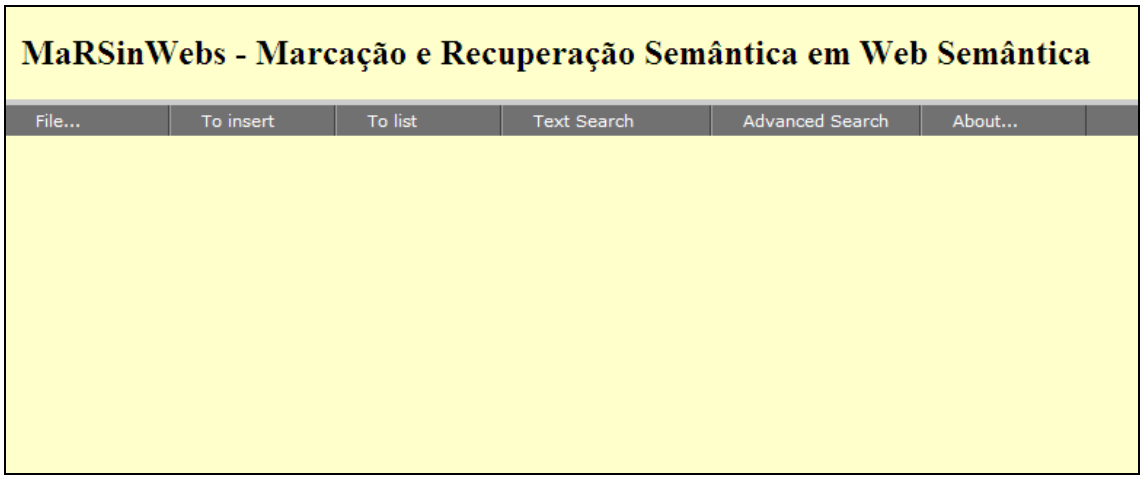

Figura 18 – Tela de boas-vindas da ferramenta.

Na tela inicial representada na Figura 18 é possível ver todos os menus que compõem a ferramenta. Ao todo são seis menus, cada qual com as seguintes opções:

1. *File...*
    a. *Save query*: nesta opção o usuário pode salvar as consultas que construiu e executou a partir das opções existentes em *Text search* ou *Advanced search*. Toda consulta salva deve ter atribuído um rótulo de identificação.
    b. *Use saved query*: permite que usuário visualize todas as consultas já salvas. Também podem ser executadas novamente a partir dos rótulos definidos pelos usuários.
2. *To insert*
    a. *Document in BD*: Esta opção permite que o usuário submeta arquivos para serem arquivados no repositório central da aplicação. Tais arquivos passaram pelo *parser* da ferramenta e as informações contidas nas anotações serão enviadas ao banco de dados.
    b. *Instance in ontology*: O item dá a possibilidade ao usuário de inserir instâncias às ontologias armazenadas no repositório de ontologias.
3. *To list*
    a. *Data base*
        i. *Class*: São listadas todas as classes existentes no banco de dados. Essas classes foram identificadas, preliminarmente, pelo *parser* no instante da submissão do arquivo.
        ii. *Instances*: São listadas todas as instâncias existentes no banco de dados. Tais instâncias foram identificadas, preliminarmente, pelo *parser* no instante da submissão do arquivo.
    b. *Ontology*
        i. *Class*: Lista, sub a estrutura de uma árvore, todas as clases de uma determinada ontologia selecionada pelo usuário.
        ii. *Instances*: Lista, sub a estrutura de uma árvore, todas as clases de uma determinada ontologia selecionada pelo usuário.

4. Text search
    a. *About class*: Permite que usuário efetue buscas textuais sobre quaisquer classes persistentes no banco de dados.
    b. *About instance*: Permite que usuário efetue buscas textuais sobre quaisquer instâncias persistentes no banco de dados.
    c. *About ontology*: Permite a consulta por documentos existentes no repositório central a partir da seleção de alguma ontologia.
5. *Advanced search*: Módulo que permite ao usuário efetuar consultas mais elaboradas dentro da ferramenta. Tal opção é que provê a funcionalidade de consulta por informações implícitas, as quais ficam somente são representadas pelas ontologias.
    a. *Super & subclass*: Opção que permite a consulta por documentos relacionados a super classes e subclasses da classe consultada.
    b. *Neighbour class*: Permite a consulta por documentos de classes que estão no mesmo nível da classe que está sendo consutada.
6. *About*: Contêm informações acerca da ferramenta desenvolvida.

A Figura 19 apresenta a opção de submissão de arquivos. Os arquivos são enviados para o modulo indexador, o qual é responsável por chamar o sub-módulo *parser* que analisará a anotação contida no documento. Em seguida, o arquivo será salvo no repositório local.

Figura 19 – Opção de submissão de arquivo para o repositório central da ferramenta.

A próxima interface apresentada na Figura 20 – Listagem de todas as classe persistentes no banco de dados.traz o resultado da listagem de todas as classes persistentes no banco de dados. Cada classe retornada na consulta é um *link* que encaminha o usuário para outra tela que identifica todos os documentos relacionados a determinada classe.

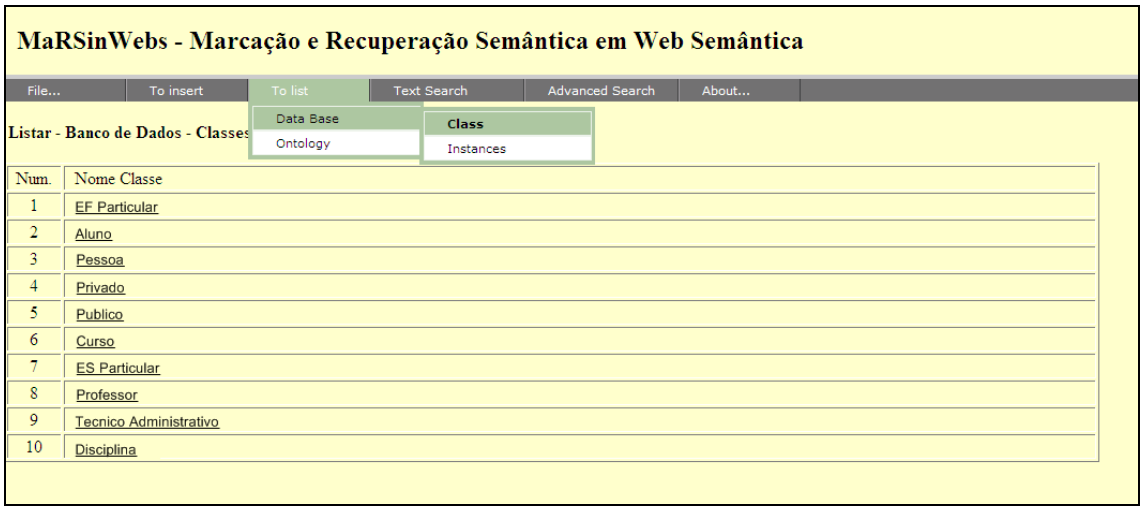

Figura 20 – Listagem de todas as classe persistentes no banco de dados.

Assim como a listagem de todas as classes existentes no banco de dados, a ferramenta também permite a listagem de todas as instâncias armazenadas na base. A Figura 21 apresenta a interface de listagem das instâncias.

Figura 21 – Listagem das instâncias do banco de dados.

Assim como na listagem das classes, também é possível ser redirecionado para a página que apresenta todos os documentos submetidos e que estão relacionados com as instâncias listadas. Cabe lembrar que, todos os documentos relacionados a alguma instância implica que tal arquivo possui uma anotação da instância dentro de sua anotação.

Ainda dentro do módulo To List existe o item *Ontology*, opções *Class* ou *Instances*. Essas duas opções são responsáveis por listar dados inerentes a alguma ontologia. A Figura 22, por exemplo, apresenta o cenário onde um usuário deseja listar as classes existentes em uma ontologia cadastrada no sistema e localizada no Repositório de Ontologias. A ontologia usada nesse exemplo chama-se *Educação.owl*, e descreve as instituições de ensino e recursos humanos ligados a essas instituições.

Figura 22 – Representação em árvore das classes pertencentes a ontologia *Educação.owl*.

O quarto item do menu superior, chamado *Text Search*, possui três formas de pesquisa. A primeira delas é apresentada na Figura 23 e é responsável por procurar por documentos que contenham anotações de classes a partir de termos informados no campo *Class*. Também há um segundo campo que fornece opções para restringir a busca pelos termos citados. As opções para seleção são mostradas na caixa localizada no lado direto da Figura 23. Tais opções são:

- *Contains*: busca por documentos onde as classes anotadas contenham o termo informado pelo usuário no campo *Class*;
- *Does not contain*: busca por documentos onde as classes anotadas não contenham o termo informado pelo usuário no campo *Class*;
- *Is*: busca por documentos onde as classes anotadas tenham o nome exatamente igual ao termo informado pelo usuário no campo *Class*;

- *Is not*: busca por documentos onde as classes anotadas apresentam o nome totalmente diferente do termo informado pelo usuário no campo *Class*;
- *Begins with*: busca por documentos onde no nome das classes anotadas comece com o termo informado pelo usuário;
- *Ends with*: busca por documentos onde no nome das classes anotadas termine com o termo informado pelo usuário.

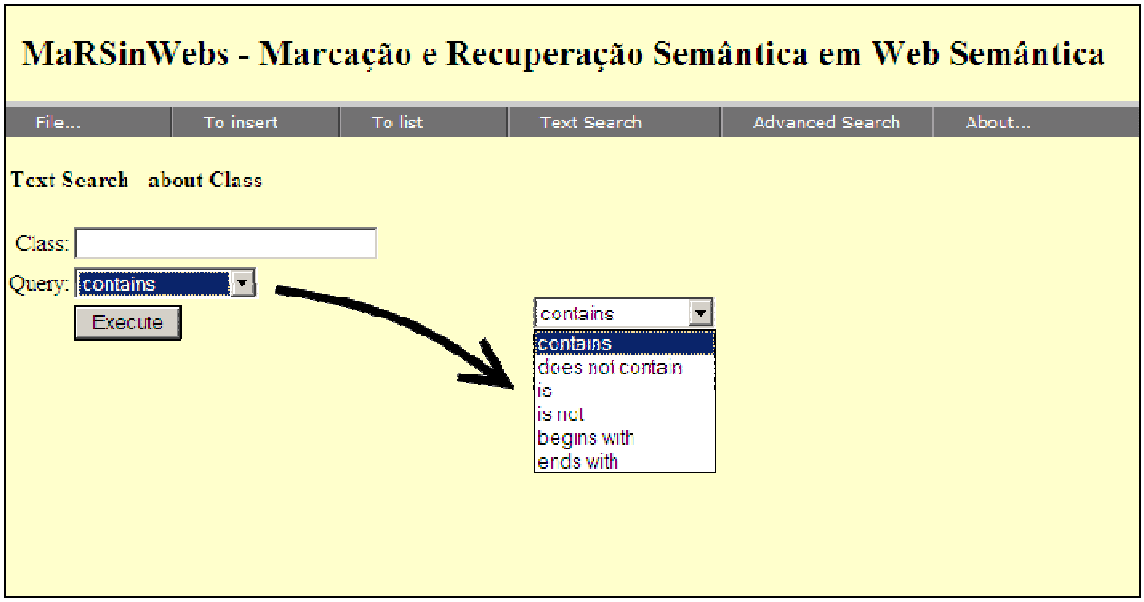

Figura 23 – Busca por documentos que possuem classes nomeadas a partir de termos pesquisados.

É importante registrar que todas as três opções do item do menu *Text Search* efetuam buscas a partir do banco de dados. As consultas que utilizam Ontologias como base são executadas a partir do menu ao lado chamado de *Advanced Search*.

Caso o usuário queira consultar por documentos com o nome da classe *Aluno*, deve ser informado no campo *Class* tal termo. Se tiver certeza de que quer receber, como resposta do sistema, documentos somente anotados com essa classe, basta escolher a opção *Is* da caixa *Query*. Em tal cenário, a resposta do sistema é apresentada na Figura 24.

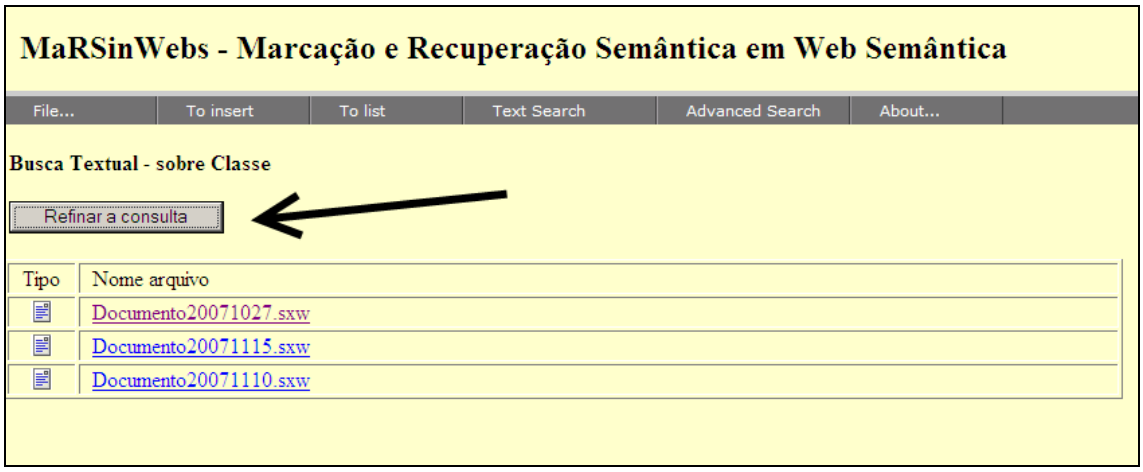

Figura 24 – Resultado da consulta por documentos anotados com a classe *Aluno*.

Na Figura 24 pode-se ver a lista de documentos anotados com a classe *Alunos*. Ao lado do *link* de cada documento existe um ícone que identifica o qual de tipo do documento dentre todos suportados pelo pacote de escritório OpenOffice.

Ainda na Figura 24 é possível notar que a resposta à consulta traz todos os documentos marcados com a classe *Aluno*, independentemente das instâncias marcadas em sua notação. Caso o usuário queira aprofundar a consulta e, por exemplo, procurar por documentos que além de serem anotados com a classe *Aluno*, também contenha alguma instância específica, existe a opção de *Deepen Search*. Tal opção encaminha o usuário a uma outra interface onde é possível definir o termo para alguma instância. A Figura 25 apresenta a interface.

Figura 25 – Resultado da busca aprofundada.

A Figura 25 mostra a busca e o resultado no quadrado ao lado. Nesse exemplo se buscou, além de documentos anotados com a classe *Aluno*, documentos que também contêm anotações de instâncias com o nome *Eduardo*. O quadrado ao lado mostra três documentos do repositório anotados com a classe *Aluno* e instância *Eduardo*.

O cenário apresentado entre as Figura 23 e Figura 25 pode ser executado a partir de pesquisas por instâncias e aprofundamento usando nomes de classes.

Ao usuário é dada a opção de salvar alguma consulta que julgar importante. Para isso, deve-se construir uma pesquisa, como a citada entre as Figura 23 e Figura 25, e após, na opção *File...* do menu superior, escolher o item *Save Query*. A Figura 26 apresenta a interface onde o usuário pode salvar as consultas.

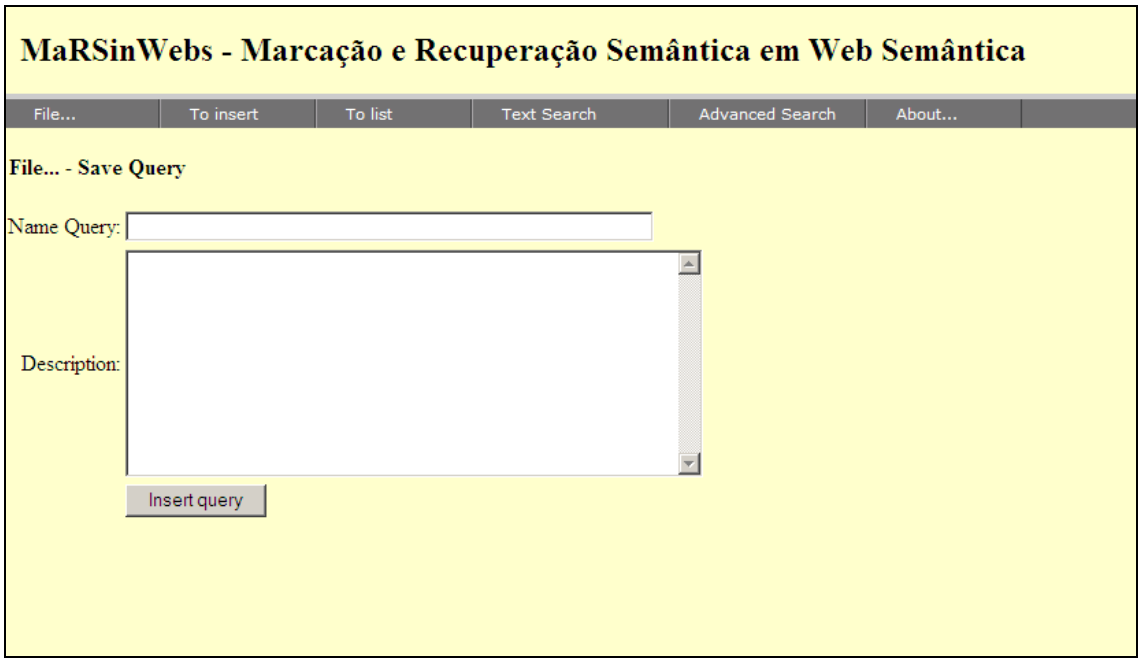

Figura 26 – Interface para cadastro de pesquisas realizadas pelos usuários

Na tela apresentada na Figura 26 o usuário deve atribuir um nome e uma descrição para a consulta construída. Assim, o sistema armazenará a consulta do usuário para que mais tarde possa ser executá-la novamente a partir do caminho *File... -> Saved Query*.

O item *Saved Query* traz uma lista das consultas salvas no sistema. Ao clicar sobre o nome de uma consulta o usuário deve ser direcionado para uma página que apresenta a lista de documentos que atendem a consulta salva.

A opção *Advanced Search*, localizada no menu superior da página, é responsável por executar as pesquisas mais complexas da ferramenta. As buscas efetuadas por esse módulo utilizam ontologias no auxílio para descoberta dos documentos importantes para os usuários. A Figura 27 apresenta um exemplo de pesquisa que pode ser executada a partir desta opção. Tal exemplo diz respeito às consultas por documentos relacionados a super e subclasses da classe consultada.

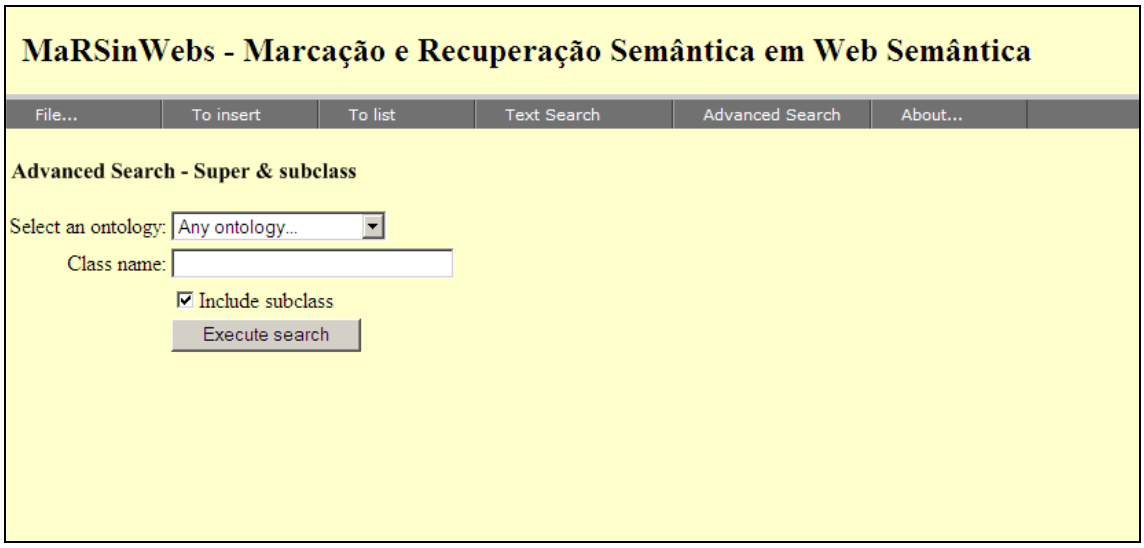

Figura 27 – Interface de pesquisa avançada da ferramenta (*super & subclass*).

A Figura 27 apresenta um exemplo simples do funcionamento do item *Advanced Query*. No cenário mostrado o usuário deseja receber como resposta do sistema a lista de documentos que contém anotação da classe *Pessoa*. É possível ver que além dos documentos anotados com a classe em questão, o sistema retornou outros documentos anotados com a classe *Servidor*. Isso porque o item *Advanced Query* uma a ontologia para identificar tal classe possui outra subclasse. Assim, a consulta considera os documentos anotados da classe e também se sua subclasse.

Tal flexibilidade aumenta a abrangência da consulta e auxilia o usuário no encontro de determinados documentos onde não se sabe exatamente por qual classe consultar. A Figura 28 apresenta o resulta da busca por documentos da classe Pessoa, aplicada a qualquer ontologia persistente no Repositório de Ontologias.

Figura 28 – Resulta da pesquisa avança para a classe *Pessoa*

Na Figura 28 é possível verificar que a pesquisa vasculha todas as ontologias existentes. São procurados documentos anotados com as classes: *Pessoa*, pois é a origem da consulta; e suas subclasses dentro de cada ontologia. Assim, há uma separação no resultado retornado ao usuário.

Além da consulta hierárquica mostrada na Figura 28 – Resulta da pesquisa avança para a classe *Pessoa*, a ferramenta permite outro tipo de consulta avançada. A busca por materiais relacionados a classes que estão no mesmo nível que a classe consultada. Por exemplo: um usuário deseja consultar por documentos de todos os tipos

de alunos do curso de um determinado curso de computação. Sabe-se que tal curso de computação tem três tipos de alunos. Os regulares, ouvintes, especiais. Porém, o usuário que deseja fazer a consulta conhece apenas os alunos do tipo regular. Como esse usuário poderá fazer uma consulta para obter documentos ligados a qualquer tipo de aluno do curso de computação? A Figura 29 apresenta a interface que pode responder ao questionamento anterior.

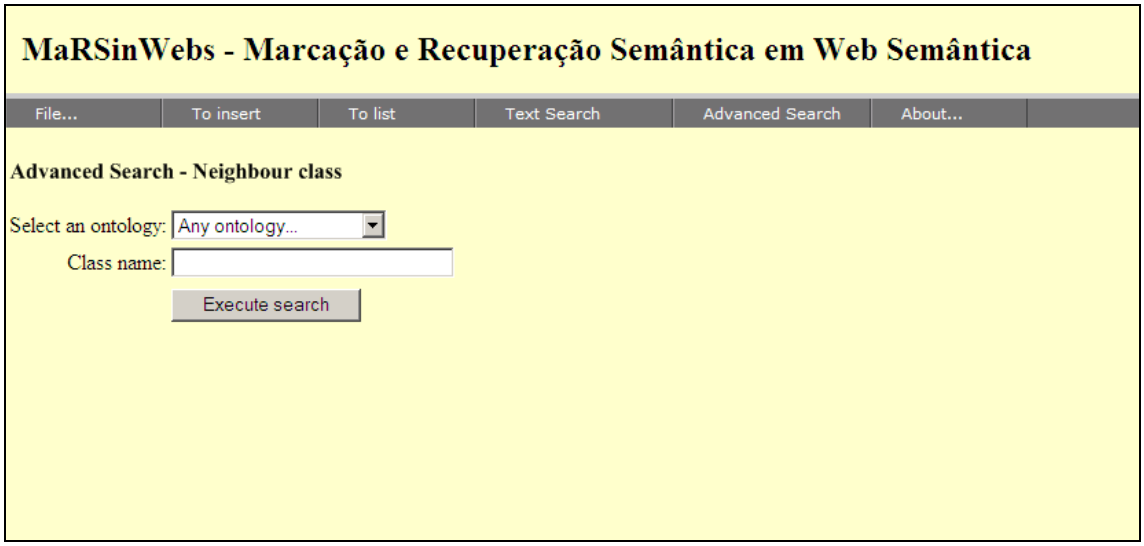

Figura 29 – Interface de pesquisa avançada da ferramenta (neighbour class).

Na interface apresentada na Figura 29 o usuário deve selecionar a Ontologia e informar uma sequência de caracteres como nome da classe, a qual quer que o sistema retorne todos os documentos relacionados às classes sobre o mesmo nível. De acordo com o questionamento usado no parágrafo anterior, o resultado da busca é apresentado na Figura 29.

Figura 30 – Resultado da consulta por documentos de classes do mesmo nível à *Regular*.

Pro fim, o módulo *About* do menu superior traz informações acerca da arquitetura e ferramenta desenvolvidos dentre do projeto.

5.3 Considerações do capítulo

O presente capítulo apresentou a ferramenta desenvolvida com o intuito de validar a arquitetura proposto nesse trabalho. Foram comentadas as funcionalidades do sistema, como: (*i*) armazenamento de consultas; (*ii*) busca textual sobre documentos; *(iii)* buscas avançadas usando ontologias para suporte a descoberta de informação; etc.

A Subseção 5.2 teve o objetivo de servir como um manual de utilização da ferramenta para os usuários que desejam manipular o sistema.

6 Conclusões e trabalhos futuros

Esse capítulo apresenta as conclusões retiradas após a fase de construção da arquitetura e implementação da ferramenta como recurso de validação de tal arquitetura. São comentados os objetivos alcançados e as lacunas em aberto que dá margem a continuação da pesquisa sob forma de trabalhos futuros.

6.1 Conclusões

O trabalho desenvolvido e descrito nesse documento atendeu aos objetivos a que se propôs. A recuperação de informação apoiada pela Web Semântica torna mais simples a tarefa de descoberta de informação, por parte do usuário.

No presente trabalho procurou-se utilizar, de forma mais avançada, a ontologia adotada como referência no instante de definir as anotações que compõem as anotações dos documentos. Como observado os trabalhos correlatos comentados no decorrer da Subseção 3.6, a grande maioria das ferramentas de recuperação de informação trabalha sobre dados textuais. A abordagem defendida pela MarSinWebs visa pesquisar dos dados com o suporte da Web Semântica.

Tal característica já a torna diferente das demais abordagens estudadas. Contudo, uma abordagem dentre as comentadas nos trabalhos correlatos se aproxima, conceitualmente, da ferramenta MarSinWebs. Todavia, a ferramenta descrita nesse trabalho está estruturada sobre o tipo de busca em nível Semântico. Assim, permite que informações implícitas as anotações dentro das anotações sejam recuperadas e apresentadas aos usuários.

Quanto à arquitetura do sistema, a utilização de um repositório para documentos e outro para ontologias agregou maior poder organizacional das informações. Porém, acarretou em uma limitação, pois as ontologias que forem alteradas fora da ferramenta

não têm como serem atualizadas dentro do repositório, se não por intervenção de algum usuário administrador.

O MarSinWebs dá ao usuário a possibilidade de executar buscas textuais ou utilizar os recursos das ontologias para efetuar tal tarefa. Além isso, a opção de salvar as consultas já efetuadas simplifica a vida do usuário que não precisa construir uma consulta novamente, toda vez que precisar consultar algo que já tenha sido executado.

6.2 Trabalhos futuros

O presente trabalho abre uma possibilidade muito grande para a realização de atividades futuras. A primeira oportunidade de atividade diz respeito a mudanças na arquitetura. Atualmente, o sistema MarSinWebs é reativo, ou seja, age a partir de estímulos gerados pelos usuários. Por exemplo, para que o sistema tenha documentos e ontologias armazenadas em seus respectivos repositórios, é necessário que o usuário submeta um documento ao sistema. Assim, tal documento é salvo e sua anotação é analisada para o futuro povoamento do banco de dados.

Um avanço sobre a atual arquitetura seria automatizar essa tarefa, fazendo com que o *Crawler* da ferramenta executasse a procura dos documentos a serem inseridos no repositório. Com isso, a tarefa de enriquecimento da base de informação dependeria da própria ferramenta, e não do usuário final, pois pode não haver a colaboração necessária na junção do material de consulta.

Uma outra restrição da arquitetura diz respeito ao tamanho do repositório central de documentos e ontologias. De acordo com a concepção atual, o sistema MarSinWebs conta com um repositório local de dados. Todos os documentos, inclusive as ontologias, são persistidos nesses repositórios locais.

Seria interessante expandir essa restrição da arquitetura, fazendo com que a ferramenta suportasse consultas na Web, ou até mesmo, em Web *Services*. Os documentos que fazem parte do repositório de consulta poderiam ser distribuídos

geograficamente, ou então, poderiam ser carregados e salvos no repositório local, a partir de outros repositórios instalados em demais laboratórios e ou universidades.

As ontologias usadas para consulta dentro do MarSinWebs também são, obrigatoriamente, objetos persistentes no sistema. Seria possível modificar essa estrutura dependente do repositório central e determinar que o módulo responsável por carregar as ontologias, procurasse tais dados em outros repositórios espalhados pela rede. Tal alteração agregaria mais informação ao repositório de dados.

A arquitetura apresentada nesse trabalho visa manipular documentos de um específico pacote de escritório, o OpenOffice. Nesse caso também haveria a possibilidade de se desenvolver um trabalho a fim de permitir que, não somente documentos do OpenOfffice fossem incorporados ao repositório, mas sim documentos de outros pacotes, como por exemplo, Microsoft Office, StarOffice, entre outros.

Referências bibliográficas

[ANTONIOU; VAN HARMELEN, 2004] ANTONIOU, Grigoris; VAN HARMELEN, Frank. **A Semantic Web Primer**. The MIT Press, 2004.

[ARASU et al., 2001] ARASU, Arvind; CHO, Junghoo, GARCIA-MOLINA, Hector, PAEPCKE, Andréas; RAGHAVAN, Sriram. **Searching the Web**. ACM Transactions on Internet Technology. Páginas 2-43. Agosto de 2001.

[BARROSO et al., 2003] BARROSO, L.A.; DEAN, J.; HOLZLE, U. **Web search for a planet: The Google cluster architecture.** IEEE Micro, Yorktown Heights, NY 10598, USA, Volume 23, n.2, Páginas: 22-28, Março de 2003.

[BAX, 2001] BAX, Marcelo Peixoto. **Introdução às Linguagens de Marcação**. Revista Ciência da Informação, Brasília, Volume 30, n. 1, páginas 32-38, Jan/Abr de 2001.

[BERNERS-LEE, 1998a] BERNERS-LEE, Tim. **Semantic Web Road Map**. Disponível em http://www.w3.org/DesignIssues/Semantic.html, acessado em 26/07/2006, 1998.

[BERNERS-LEE, 1998b] BERNERS-LEE, Tim. **Universal Resource Identifiers in WWW**. Internet Engineering Task Force Request for Comments RFC1630, Disponível em http://www.ietf.org/rfc/rfc1680.txt?number=1680, acessado em 24/01/2007. 1998.

[BERNERS-LEE et al., 1998] BERNERS-LEE, Tim; FIELDING, R.; MASINTER, L.. **Uniform Resource Indentifiers (URI): Generic Syntax**. Internet Engineering Task Force Request for Comments RFC2396, Disponível em http://www.ietf.org/rfc/rfc1680.txt?number=2396, acessado em 13/02/2007. 1998b.

[BERNERS-LEE, 2000] BERNERS-LEE, Tim**. Semantic Web**. Slides apresentados no congresso XML 2000, Washington, DC. Disponível em http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html. Acessado em 25/07/2005. Junho, 2000.

[BERNERS-LEE et al., 2001] BERNERS-LEE, Tim.; HENDLER, J.; LASSILA, O.. **The Semantic Web**. Revista Scientific American, Maio de 2001.

[BIANCHINI et al., 2006] BIANCHINI, Devis; DE ANTONELLIS, Valeria; PERNICI, Bárbara; PLEBANI, Pierluigi.. **Ontology-based methodology for e-service discovery**. Inf. Syst. Elsevier Science Ltd, Oxford, UK, Volume 31, n. 4, páginas 361-380, 2006.

[BOAG et al, 2003] BOAG, S; CHAMBERLIN, D.; FERNÁNDEZ, F. M.; FLORESCU, D.; ROBIE, J.; SIMÉON, J.. XQuery 1.0: An XML query language. Disponível em http://www.w3.org/TR/xquery/. Acessado em 06/2007. 2003.

[BOUQUET et al., 2004] BOUQUET, P.; GIUNCHIGLIA, F.; HARMELEN, F.; SERAFINI, L; STUCKENSCHMIDT, H.. **Contextualizing Ontologies**. Lecture Notes in Computer Science - Journal of Web Semantics - ISWC 2003. Springer Berlin / Heidelberg , volume 2870/2003, Setembro de2004.

[BRICKLEY et al., 2004] BRICKLEY, D.; GUHA, R. V.; MCBRIDE, B.. RDF **Vocabulary Description Language 1.0: RDF Schema**. Disponível via http em: http://www.w3.org/TR/rdf-schema/, acessado em Janeiro de 2007, 2004.

[CHEN et al., 2003] CHEN, Zheng; LIU, Shengping; WENYIN, Liu; PU, Geguang; MA, Wei-Ying. **Building a web thesaurus from web link structure**. SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. ACM Press, New York, NY, USA , páginas 48-55, 2003.

[CUSUMANO, 2005] CUSUMANO, M. A.. **Google: what it is and what it is not**. Commun. ACM. ACM Press, volume 48, n. 2, páginas 15-17, New York, NY, USA, 2005.

[DACONTA et al., 2003] DACONTA, M. C.; LEO, J.; OBRST, L. J.; SMITH, K. B.. **The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management**. Wiley, Maio de 2003.

[DAVIES et al., 2003] DAVIES, J.; FENSEL, D.; HARMELEN, F.. **Towards the Semantic Web: Ontology-drive Knowledge Management**. Chapter 5 - Sesame, A Generic Architecture for Storing and Querying RDF and RDF Schema. John Wiley & Sons, Ltd, páginas 71-89, 2003.

[DEITEL et al., 2003] DEITEL, H. M; DEITEL, P. J.; NIETO, T. R.. **XML: Como Programar**. Bookman, volume 1, n. 974, páginas 972, Porto Alegre, Março de 2003.

[DEITEL; DEITEL, 2005] DEITEL, H. M; DEITEL, P. J**.. Java: Como Programar**. Pearson, volume 6, páginas 1152, Porto Alegre, Junho de 2005.

[DAML, 2000] DAML.ORG: Sítio oficial da linguagem DAML. Disponível em http://www.daml.org/, acessado em 25/07/2007, 2000.

[DOAN et al., 2002] DOAN , AnHai; MADHAVAN , Jayant; DOMINGOS, Pedro; HALEVY, Alon. **Learning to map between ontologies on the semantic web**. WWW '02: Proceedings of the 11th international conference on World Wide Web, ACM Press, páginas 662-673, 2002.

[EDWARDS, 1997] EDWARDS, M.. **XML: Data the Way You Want It**. Publicado via http em: http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnarxml/html/xmldata.asp, acessado em Janeiro de 2007. Outubro de 1997.

[EIKVIL, 1999] EIKVIL, L**.. Information Extraction from World Wide Web - a Survey**. Norweigan Computing Center. Tecnical report, 1999.

[FENSEL, 2003] FENSEL, D.. **Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce**. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2003.

[FERREIRA, 2006] FERREIRA, A. B. H.. **Novo dicionário Aurelio versão 5.0 edição revista e atualizada**. Positivo. Terceira Edição. Curitiba, 2006.

[FREITAS, 2003] FREITAS, F.. **Ontologias e a Web Semântica**. Anais do XXIII Congresso da Sociedade Brasileira de Computação. Jornada de Mini-Cursos em Inteligência Artificial. Ed. Campinas: Sociedade Brasileira de Computação (SBC), volume 8, páginas 1-52, 2003.

[GLONVEZYNSKI, 2005] GLONVEZYNSKI, Régis A.. **Modelo de anotação de documentos para a codificação do conteúdo semântico no processo de autoria**. Dissertação de Mestrado apresentada para conclusão do Programa de Pós-Graduação em Ciência da Computação. UFSC, SC, 2005.

[GÓMEZ; BENJAMINS, 1999] GÓMEZ-Pérez, A.; BENJAMINS, V. R.. Overview of knowledge sharing and reuse components: Ontologies and problem-solving methods. International Joint Conference on Artificial Intelligence(IJCAI-99). Workshop on Ontologies and Problem-Solving Methods (KRR5). Stockolm, Sweden, 1999.

[GRAU, 2004] GRAU, Bernardo C.. **A Possible Simplification of the Semantic Web Architecture**. Proceedings of the 13th international conference on World Wide Web. Páginas 704-713. 2004.

[GRUBER, 1993] GRUBER, Thomas R.. **A translation approach to portable ontology specifications**. Knowledge Acquisition, 5:199–220, 1993.

[GUHA et al., 2003] GUHA, R.; MCCOOL, Rob; MILLER, Eric. **Semantic Search**. Proceedings of the twelfth international conference on World Wide Web. Budapest, Hungary. Páginas 20-24. 2003.

[HARMELEN; HORROCKS, 2000] HARMELEN, F.; HORROCKS, I.. **Questions and answers on OIL: the Ontology Inference Layer for the semantic web**. IEEE Intelligent Systems, volume 15, n. 6. páginas 69-72, Dezembro de 2000.

[HARRUSI, 2006] HARRUSI, S.; AVERBUCH, A.; YEHUDAI, A.. **XML syntax conscious compression**. Data Compression Conference. Sch. of Comput. Sci., Tel Aviv Univ., Israel. Março de 2006.

[HATALA, 2005] HATALA, M.; WAKKARY, R.; KALANTARI, L.. **Rules and Ontologies in Support of Real-Time Ubiquitous Application**. Journal of Web Semantics, páginas 5-22, Julho de 2005.

[HEFLIN; HENDLER, 2000] HEFLIN, J. and HENDLER, J**.. Searching the Web with SHOE**. Artificial Intelligence for Web Search. Menlo Park, CA, páginas 35-40, 2000.

[HEFLIN, 2001] HEFLIN, J.. **Towards the Semantic Web: Knowledge Representation in a Dynamic, Distributed Environment**. University of Maryland. College Park, 2001.

[HERMAN et al., 2004] HERMAN, I.; SWICK, R.; BRICKLEY, D.. **Resource Description Framework (RDF)**. Disponível via http em: http://www.w3.org/RDF/, acessado em Janeiro de 2007, 2004.

[HORROCKS et al., 2000] HORROCKS, I.; FENSEL, D.; BROEKSTRA, J.; DECKER, S.; ERDMANN, M.; GOBLE, C.; HARMELEN , F.; KLEIN, M.; STAAB, S.; STUDER, R.; MOTTA, E.. **The Ontology Inference Layer OIL**. Disponível via http em: http://www.ontoknowledge.org/oil/TR/oil.long.html, acessado em Janeiro de 2007, 2000.

[HORROCKS et al., 2005] HORROCKS, I; PATEL-SCHNEIDER, P. F.; BECHHOFER S.; TSARKOV D.. **OWL Rule: A Proposal and Prototype Implementation**. Journal of Web Semantics, páginas 23-40, Julho de 2005.

[JANSEN et al., 2003] J. JANSEN, Bernard; SPINK, Amanda; PEDERSEN, Jan.. **An analysis of multimedia searching on AltaVista**. MIR '03: Proceedings of the

5th ACM SIGMM international workshop on Multimedia information retrieval. ACM Press, New York, NY, USA, páginas 186-192, 2003.

[KARVOUNARAKIS, 2003] KARVOUNARAKIS, Greg. Disponível por http://139.91.183.30:9090/RDF/RQL/. Acessado em 06/2007. 2003.

[KIRYAKOV et al., 2003] KIRYAKOV, Atanas. POPOV, Borislav. OGNYANOFF, Damyan. **Semantic Annotation, Indexing, and Retrieval**. 2nd International Semantic Web Conference (ISWC2003), Florida, USA. Páginas 484 a 499, Outubro, 2003.

[KOIVUNEN; MILLER, 2001] KOIVUNEN, M.; MILLER, E.. **W3C Semantic Web Activity**. Disponível por http em: http://www.w3.org/2001/12/semweb-fin/w3csw, acessado em Março de 2007, Fevereiro de 2001.

[KRAFT et al., 2006] KRAFT, Reiner; CHANG, Chi Chao; MAGHOUL, Farzin; KUMAR, Ravi.. **Searching with context**. WWW '06: Proceedings of the 15th international conference on World Wide Web. ACM Press, New York, NY, USA, páginas 477-486, 2006.

[LASSILA, 1998] LASSILA, O.. **Web metadata: a matter of semantics**. IEEE Internet Computing. Páginas 30-37, 1998.

[LI; LI, 2004] LI, Hong; LI, Lei.. **A DTD-conscious sparse numbering scheme**. The Fourth International Conference on Computer and Information Technology. N. xxi+1168, páginas 295-302, Setembro de 2004.

[MANBER et al., 2000] MANBER, U.; PATEL, A.; ROBISON, J.. **Experience with personalization of Yahoo!**. Commun. ACM. ACM Press, New York, NY, USA, volume 43, n. 8, páginas 35-39, 2000.

[MCCUE, 1999] MCCUE, C.. **Exploring the World with Yahoo!**. Hungry Minds, Incorporated. 1999.

[MCGUINNESS; HARMELEN, 2004] MCGUINNESS, D. and HARMELEN, F.. **OWL Web Ontology Language – Overview**. Disponível via http em: http://www.w3.org/TR/owl-features/, acessado em Janeiro de 2007, 2004.

[MCILRAITH et al., 2001] MCILRAITH, S.A.; SON, T.C.; ZENG, H.. **Semantic Web services**. Intelligent Systems, IEEE. Knowledge Syst. Lab., Stanford Univ., CA, USA, páginas 46-53, 2001.

[MILLER, 2001] MILLER, L.. RDF Squish query language. Disponível em em: http://ilrt.org/discovery/2001/02/squish/. Acessado em 06/2007. 2001.

[MORAIS; SOARES, 2004] MORAIS, E. F.; SOARES, M. B.. **Web Semântica para Máquinas de Busca**. Seminário da disciplina Projeto e Análise de Algoritmos - PPGCC/UFMG, Maio de 2004.

[MOURA, 2001] MOURA, Ana Maria de C.. **A Web Semântica: Fundamentos e Tecnologias**. Congreso Internacional de Ciencias de la Computación – CICC 2001. Universidad de Aquino. Aquino, Bolivia. 2001.

[HORRIDGE et al., 2007] HORRIDGE, Matthew; BECHHOFER, Sean; NOPPENS, Olaf. **Igniting the OWL 1.1 Touch Paper: The OWL API. OWLED 2007**. 3rd OWL Experienced and Directions Workshop, Innsbruck, Austria, Junho 2007.

[SILVA, 2003] SILVA, T. M. S.. **Extração de Informação para a Busca Semântica na Web Baseada em Ontologias**. Universidade Federal de Santa Catarina. Dissertação de mestrado, Florianópolis, Julho de 2003.

[NOY et al., 2001] NOY, N. F.; SINTEK, M.; DECKER, S.; CRUBÉZY, M.; FERGERSON, R. W.; Musen, M. A.. **Creating Semantic Web Contents with Protégé-2000**. IEEE Intelligent Systems - The Semantic Web, volume 16, páginas 60-71, Med. Inf. Lab., Stanford Univ., CA, USA, Março/Abril de 2001.

[OFFICE, 1998] **OFFICE.MICROSOFT.COM**. Sítio oficial. Microsoft Corporation. Disponível em http://office.microsoft.com/. Acessado em 25/07/2005, 1998.

[ONTOKNOWLEDGE.ORG/OIL, 2000] **ONTOKNOWLEDGE.ORG/OIL**. Sítio oficial da linguagem OIL**.** Disponível em http://www.ontoknowledge.org/oil/. Acessado em 05/12/2007, 2000.

[OPENOFFICE, 2002 a] **OPENOFFICE.ORG**. Sítio oficial. Sun Microsystems. Disponível em http://www.openoffice.org/. Acessado em 25/07/2005.

[OPENOFFICE, 2002 b] **OPENOFFICE.ORG XML FILE FORMAT 1.0**. Technical Reference Manual. Version 2. Sun Microsystems, dezembro 2002.

[PAGELS, 2006] PAGELS, M.. **The DARPA Agent Markup Language Homepage**. Disponível via http em: http://www.daml.org/, acessado em Janeiro de 2007. Tecnical report, 2006.

[PROTÉGÉ, 1995] **PROTÉGÉ ONTOLOGY EDITOR**. Sítio oficial. Stanford University School of Medicine. Disponível em http://protege.stanford.edu/. Acessado em 20/07/2005, 1995.

[RILOFF; LEHNERT, 1994] RILOFF, Ellen; LEHNERT, Wendy.. **Information extraction as a basis for high-precision text classification**. ACM Trans. Inf. Syst. ACM Press, New York, NY, USA, volume 12, n. 3, páginas 296-333, 1994.

[RUSSEL; NORVIG, 2003] RUSSEL, S.; NORVIG, P.. Inteligência **Artificial, a Modern Approach**. Prentice Hall series in Artificial Inteligence. Segunda edição, 1100 páginas, 2003.

[SANDERSON, 1994] SANDERSON, M.. **Word sense disambiguation and information retrieval**. Proceedings of SIGIR-94. 17th ACM International Conference on Research and Development in Information Retrieval, páginas 49-57, 1994.

[SWOOGLE, 2005] **SWOOGLE.UMBC.EDU**. Sítio oficial. Disponível em http://swoogle.umbc.edu/. Acessado em 25/011/2007, 2005.

[UREN et al., 2005] UREN, V.; CIMIANO, P.; IRIA, J.; HANDSCHUH, S.; VARGAS-VERA, M.; MOTTA, E.; CIRAVEGNA, F.. **Semantic annotation for knowledge management: Requirements and a survey of the state of the art**. Journal of Web Semantics, 4(1): 14-28, January 2005.

[ZAGO, 2005] ZAGO, R. F.. **Modelo de Recuperação e Indexação de Conhecimento em Documentos Corporativos Anotados Semanticamente**. Dissertação de Mestrado apresentada para conclusão do Programa de Pós-Graduação em Ciência da Computação. UFSC, SC, 2005.

[W3C/OWL, 2004] **W3.ORG/OWL Web Ontology Language Overview**. Sítio oficial. World Wide Web Consortium. Disponível em http://www.w3.org/TR/owl-features/. Acessado em 08/02/2007, 2004.

[W3C/SW, 2001] W3C Semantic Web Activity. Sítio official. World Wide Web Consortium. Disponível em http://www.w3.org/2001/12/semweb-fin/w3csw. Acessado em 08/02/2007, 2001.