ANAUÊ PEREIRA DA COSTA
FABIO SENDODA YAMATE

Semantic Lattes: uma ferramenta de consulta de informações acadêmicas da base Lattes baseada em ontologias

São Paulo
2009

ANAUÊ PEREIRA DA COSTA
FABIO SENDODA YAMATE

Semantic Lattes: uma ferramenta de consulta de informações acadêmicas da base Lattes baseada em ontologias

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo como Conclusão de Curso de Graduação em Engenharia de Computação.

São Paulo
2009

ANAUÊ PEREIRA DA COSTA
FABIO SENDODA YAMATE

Semantic Lattes: uma ferramenta de consulta de informações acadêmicas da base Lattes baseada em ontologias

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo como Conclusão de Curso de Graduação em Engenharia de Computação.

Área de concentração:
Engenharia da Computação

Orientador:
Prof. Dr. Jaime Simão Sichman

Co-orientador:
Profa. Dra. Anarosa Alves Franco Brandão
Profa. Dra. Renata Wassemann

São Paulo
2009

FICHA CATALOGRÁFICA

Costa, Anauê Pereira da

Semantic Lattes: uma ferramenta de consulta de informações acadêmicas da base Lattes baseada em ontologias / A.P. da Costa, F.S. Yamate. – São Paulo, 2009. 97 p.

Trabalho de Formatura — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais..

1. Ontologia 2. Web Semântica I. Yamate, Fabio Sendoda II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais.. III. t.

Às nossas famílias,
pelo apoio e compreensão em mais
uma etapa de nossas vidas.
Aos mestres, pelo conhecimento e pela
apoio em nossos caminhos.

AGRADECIMENTOS

Ao Professor Jaime Sichman, pela orientação e confiança na concretização deste trabalho.

À Professora Anarosa Brandão pelo apoio técnico e pelo auxílio na realização deste trabalho.

À Professora Renata Wassermann pela ajuda no entendimento dos conceitos necessários para a conclusão deste trabalho.

À Escola Politécnica da Universidade de São Paulo, que muito nos ensinou nesta importante etapa de nossas vidas.

Ao Departamento de Engenharia de Computação e Sistemas Digitais, pelo conhecimento adquirido em nossa formação como Engenheiros no Curso Cooperativo de Engenharia da Computação.

RESUMO

A Internet revolucionou a forma como criamos conteúdo e trocamos informações. Entretanto, a Internet é livre de regras: qualquer um pode desenvolver um conteúdo sem rigor e o papel do computador é unicamente o de apresentar tal informação. Podemos classificar a Web atual como sintática e, o processo de interpretação do conteúdo é de responsabilidade dos usuários. A Web Semântica propõe transformar este volume de informações em documentos formatados, de modo que computadores passem a ter a capacidade de interpretá-los. Essa etapa é feita pela criação de uma ontologia, que tem como papel conceitualizar um domínio. Com base neste conceito de Web Semântica, será desenvolvido, como projeto de formatura, a descrição de uma ontologia para o domínio de currículos Lattes, mantido pelo CNPq. Assim, o objetivo é melhorar a busca de informações em currículos acadêmicos.

ABSTRACT

The Internet has dramatically changed the way we create content and exchange information. However, the Internet is rule free: anyone can produce any content, without any rigorour, and the computer function is uniquely to present this information. We can classify the current Web as syntactic data, and the process to interpret it is delegated to the users. The Semantic Web proposes to convert these information in formatted documents so that computers are able to interpret them. The process of attachy semantic values to the contents, is done by creating an ontology, which is the conceptualization of a domain. Based on this concept of the Semantic Web, our final undergraduation project is the development of an ontology in the domain of Lattes Curriculum, maintained by CNPq. The objective is to improve information search on academic curriculums.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

W3C World Wide Web Consortium

XML Extensible Markup Language

DTD Document Type Definition

RDF Resource Description Framework

RDFS RDF Schema

OWL Web Ontology Language

URI Uniform Resource Identifier

SPARQL Simple Protocol and RDF Query Language

LISTA DE SÍMBOLOS

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação Do Sistema

O projeto consiste na elaboração de um sistema capaz de realizar consultas sobre os currículos Lattes, de uma forma que pretenda ser mais eficiente do que o atual sistema de busca encontrado no Portal Lattes[1]. O sistema do CNPq utiliza uma forma tradicional de buscas baseado em formulários com inúmeros campos a serem preenchidos e selecionados, se assemelhando a um questionário, no qual cada "pergunta" está relacionada a filtragem de certas informações em toda a base persistida em bancos de dados relacionais.

Do ponto de vista do usuário, esta abordagem é pouco intuitiva e imprecisa. Seria muito mais conveniente se ele pudesse realizar a busca de interesse através de perguntas mais específicas, como por exemplo:

> Quais são os professores do Departamento de Engenharia de Computação da EPUSP?

ou

> Quais são as publicações dos professores do Departamento de Engenharia de Computação da EPUSP no ano de 2009?

Infelizmente, certas informações não estão explicitamente declaradas em banco de dados relacionais e não existe uma propriedade que relacione semanticamente um Professor a um Departamento de forma clara, como *trabalhaNoDepartamento*. Tradicionalmente, este tipo de modelo de informação seria modelado por diversas tabelas em banco de dados relacionais, que não são bons para representar conhecimento, pois a sua função é essencialmente de armazenamento de dados.

[1] http://buscatextual.cnpq.br/buscatextual.

Figura 1.1: *A interface atual de busca do Lattes.*

O sistema desenvolvido neste projeto atuará como uma plataforma de consulta à currículos de forma mais próxima daquilo que o usuário deseja, utilizando para isso questões definidas no domínio de currículos Lattes com uma abordagem em ontologias, descrevendo os conceitos dos domínios usados, onde os dados são estruturados como grafos utilizando mecanismos de inferência para obter conclusões lógicas sobre o conhecimento existente.

1.2 Escopo

O sistema proposto nesse projeto, nomeado Semantic Lattes, consiste em uma ferramenta de busca de informações sobre um conjunto de currículos Lattes, provenientes do Portal Lattes, utilizando as relações semânticas existentes entre os dados dos currículos como forma de melhoria na precisão das buscas.

As buscas e inferências de informações curriculares ficarão mais precisas com a utilização de conceitos e técnicas de Web Semântica. As aplicações desses conceitos envolverão algumas práticas como a criação e utilização de ontologias dos domínios Lattes e Qualis, a utilização

de um motor de inferências (conhecido como *reasoning engine*), e a utilização de uma base de inferências apropriada para armazenar as instâncias dos dados.

As ontologias estarão descritas em documentos elaborados em OWL, uma linguagem específica para descrição de ontologias. Os domínios sobre os quais essas ontologias serão baseadas foram divididos e limitados ao universo das informações acadêmicas do Lattes e do Qualis.

1.2.1 Portal Lattes

A base Lattes contém indexados atualmente mais de um milhão de currículos em seus sistemas, e a análise desses dados demandaria muito tempo de processamento. Por esse motivo o Semantic Lattes realizará as inferências das informações acadêmicas sobre um conjunto bem menor e limitado de currículos, que sofrerão a análise ontológica no momento em que forem incorporados ao sistema.

O Semantic Lattes, não acessará diretamente a base Lattes, tendo em vista que o portal Lattes não oferece outra interface pública para consultas de currículos além da página de busca textual, disponível no portal. Além do extenso número de currículos existentes, o Lattes disponibiliza à cada usuário o acesso ao seu próprio currículo Lattes que, através de um login no site, pode ser baixado em um arquivo no formato XML. E será justamente este arquivo que o Semantic Lattes utilizará como fonte de dados para a realização das inferências.

Os scripts de carregamento dos arquivos fazem parte de uma ferramenta auxiliar, com o desenvolvimento também previsto na execução deste projeto, que realizará a tradução dos currículos em XML para as instâncias das ontologias em OWL.

1.2.2 Portal Qualis

O Portal Qualis disponibiliza informações sobre o impacto de publicação da produção intelectual no país. Segundo a descrição disponível no próprio Portal, disponibiliza uma lista com a classificação dos veículos utilizados pelos programas de pós-graduação para a divulgação da sua produção. O sistema Semantic Lattes utilizará então essas estratificações na elaboração de resultados, relacionando cada artigo e publicação referenciada nos currículos Lattes com as classificações Qualis.

1.3 Objetivo

A manutenção da qualidade do sistema de ensino de nível superior no país depende fortemente da atuação do CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), uma instituição ligada ao MEC que atua na avaliação e classificação dos cursos de pós-graduação ligados ao SNPG (Sistema Nacional de Pós-Graduação). Esse processo de avaliação de cursos envolve, dentre outras aspectos, o levantamento das publicações acadêmicas pertencentes ao curso em avaliação. Para tanto, uma das formas de determinar quais produções pertencem a um curso específico pode ser feita através de consultas nos currículos da Plataforma Lattes.

Na base de dados da Plataforma Lattes está concentrado um gigantesco volume de currículos originados principalmente dos membros das instituições de ciência e tecnologia. Nessa base estão apresentados currículos estruturados segundo um formato definido pela junção dos formatos utilizados nos principais centros acadêmicos no país, visando a uniformização e facilidade de integração e portabilidade dos currículos entre estes centros.

Em geral, as avaliações de cursos superiores são conduzidas por comissões de consultores de alto nível, ligados a instituições das diferentes regiões do país que precisam determinar se as produções expostas na plataforma Lattes são publicações de caráter reconhecido. Outra tarefa seria verificar se tais produções efetivamente existem, visto que esse currículos são mantidos pelos próprios usuários e não existem formas de validação de todos esses documentos. Para efetuar uma avaliação mais completa e elaborada, é necessário acessar uma outra base de dados também disponibilizada e mantida pelo CNPq, chamada plataforma Qualis. O Qualis avalia e classifica periódicos e conferências nacionais de acordo com a importância na área de pesquisa no qual elas são divulgadas, do modo que um mesmo periódico pode possuir diferentes classificações em diferentes áreas de pesquisa. Logo, os artigos e publicações pertencentes a um determinado periódico ganham a importância do seu periódico no processo de avaliação daqueles cursos de nível superior.

Nesse contexto, o projeto de formatura apresentado neste documento objetiva o desenvolvimento de uma ferramenta que auxilie e agilize o processo de consulta de informações dos currículos disponibilizados na base Lattes e na base Qualis. Para este fim, serão criadas ontologias para estas duas bases, e essa ferramenta deve utilizar conceitos e hierarquias semânticas existentes nestas ontologias.

1.4 Organização do Texto

Este documento está estruturado em 6 capítulos, conforme descrito a seguir: No capítulo **1 - Introdução** são apresentadas as motivações que levaram à criação do sistema de consultas proposta no projeto, contextualizando o sistema final no processo de avaliação de cursos de Pós-Graduação. São apresentadas também as funcionalidades que o sistema irá oferecer e aquelas não serão consideradas neste projeto. No capítulo **2 - Web Semântica** são descritos os conceitos básicos e as linguagens envolvidas em um sistema de Web Semântica. No capítulo **3 - Semantic Lattes** é apresentado o sistema resultante do projeto, apresentando sua arquitetura e funcionalidades. No capítulo **4 - Metodologia e Desenvolvimento** são detalhadas as formas adotadas de elaboração do projeto, focando no desenvolvimento das ontologias que compõe o núcleo do sistema. No capítulo **5 - Implementação e Teste** são apresentadas os conceitos e as ferramentas utilizadas na realização do projeto e na execução do sistema. E finalmente, no capítulo **6 - Conclusões** são analisadas os resultados do projeto e apresentadas algumas perspectivas futuras.

2 WEB SEMÂNTICA

Na construção de uma ferramenta que realize consultas semânticas em uma base de dados, é preciso ter muito claros os conceitos básicos sobre os quais a execução das tarefas se apoiam. Deste modo, serão apresentadas neste capítulo as principais idéias e as teorias que fundamentam a elaboração e execução deste projeto.

A Internet contemporânea, no moldes do *World Wide Web* (www) idealizado por Tim Berners-Lee, sempre viveu um constante processo de evolução e crescimento. O que no início eram apenas apresentações estáticas de documentos em *hypertexto*, armazenados em poucos servidores na rede, acabou se tornando uma entidade de colossais proporções no qual as informações disponibilizadas cresce de maneira exponencial e são geradas dinâmicamente de forma colaborativa nos inúmeros servidores espalhados na rede.

Essa quantidade quase inimaginável de documentos acessíveis na rede é composta de dados e informações de quase qualquer área do conhecimento humano e, no entanto, ainda não existem meios eficientes de aproveitar todo esse conteúdo disponível. Nesse contexto, a Web Semântica surge como uma extensão, e evolução, da Rede Mundial usando conceitos que primam por formas de organização desse conhecimento.

2.1 Definições Básicas

Muito do contéudo disponível hoje na Rede Mundial está em um formato que foi criado para a leitura humana, e não para manipulação por computador (BERNERS-LEE; HENDLER; LASSILA, 2001). Assim, uma das propostas da Web Semântica é organizar informações para possibilitar que máquinas processem conteúdos além da simples forma de bits de dados. Para isso, admite-se o uso de metadados estruturados de modo que seja possível categorizar as informações visando um aumento de desempenho na realização de buscas e na utilização desses resultados. A partir desses metadados, as ontologias são estabelecidas através da conceitualização de domínios de conhecimento, provendo relacionamentos semânticos às informações atualmente

existentes.

Apesar de a Web Semântica estar profundamente ligada com os conceitos e as metodologias da Inteligência Artificial (IA), isto não significa que aquela tem os mesmos propósitos desta. O que se propõe com uma Internet semântica não é algo mágico que tornará as máquinas capazes de compreender a linguagem e conhecimento humano (BREITMAN, 2005), mas ocorre sim uma utilização de técnicas de IA para criar soluções parciais satisfatórias na utilização humana do conhecimento disponível nos sistemas de informação.

2.1.1 Domínio

Pode-se definir domínio de conhecimento como algum segmento do universo que pode ser especificado na forma de um conhecimento (RUSSELL; NORVIG, 2003). Dessa forma, a representação de conhecimento em si necessariamente implica na existência de um domínio que delimita determinada área do saber. Na lógica de predicados, utilizada na execução de inferências, pode-se ainda definir o conceito de "domínio de discurso", ou "universo de discurso", como sendo um conjunto de objetos indicados na representação formal declarativa de um conhecimento de um domínio.

Um domínio muitas vezes pode ser visto como um contexto no qual se está inserido os seus elementos. Os significados, o tipo de relacionamento existente e a importância são inteiramentes dependentes do domínio em que se está. Por exemplo, temos a palavra Tanque, que possui diferentes significados dependendo da contexto no qual está sendo utilizado: no domínio das armas de combate, a palavra tanque está se referindo a um veículo blindado, já na área de conhecimento da hidraúlica, tanque significa uma forma de armazenamento de água.

2.1.2 Ontologia

Ontologias são os elementos centrais na Web Semântica, pois modelam um domínio de conhecimento e possibilitam armazenar em documentos uma forma de representação da semântica existente. Tal forma permite um aumento no nível de comunicação entre humanos e máquinas.

2.1.2.1 Conceito

Uma ontologia pode adquirir diferentes significados e representações dependendo da área em que é utilizada. Guarino e Giaretta em 1995 identificaram algumas categorias de interpretações para o termo "ontologia" (GUARINO; GIARETTA, 1995):

1. Ontologia como uma disciplina da Filosofia;
2. Ontologia como um sistema conceitual informal;
3. Ontologia como uma proposta semântica formal;
4. Ontologia como uma especificação de uma conceitualização;
5. Ontologia como uma representação de um sistema conceitual por meio de uma teoria lógica:
   (a) Caracterizada por propriedades formais, ou;
   (b) Caracterizada apenas para propósitos específicos;
6. Ontologia como um vocabulário usado por uma teoria lógica;
7. Ontologia como um meta-nível de especificação de uma teoria lógica.

Todas essas interpretações, com exceção da primeira relacionada com o conceito de Ontologia de Aristótoles, tem conceitos relativamente parecidos e inclusive podem ser agrupadas em:

- interpetrações que definem uma ontologia como uma entidade conceitual semântica (interpretações 2 e 3);
- interpretações que definem uma ontologia como um objeto concreto em nível sintático (interpretações 4 a 7).

Esse segundo grupo é o que melhor se adequa aos interesses da Web Semântica, especificamente a interpretação 4, proposta por Gruber em (GRUBER, 1993), é a mais popular das definições e mais problemática segundo Guarino e Giaretta, pois o seu significado é inteiramente dependente da interpretação dos termos "especificação" e "conceitualização". Gruber mais tarde reformulou essa definição em (GRUBER, 2009)

"Uma ontologia é uma explícita especificação de uma conceitualização"

E uma possível interpretação dos termos dessa definição é apresentada em (BREITMAN, 2005):

> "*Conceitualização* representa um modelo abstrato de algum fenômeno que identifica os conceitos relevantes para o mesmo; *explícita* significa que os elementos e suas restrições estão claramente definidos; *formal* (especificação) significa que a ontologia deve ser passível de processamento automático"

2.1.2.2 Formalização

O papel das ontologias é contextualizar o vocabulário utilizado e fornecer um padrão para o compartilhamento da informação e conhecimento. Na Web Semântica, a formalização desse vocabulário é feita através da definição de termos e seus relacionamentos, assim como outros elementos existentes para o domínio. Já o conhecimento é representado através da criação de axiomas que descrevem as restrições deste domínio. Assim, para que seja possível realizar inferências com a ontologia e obter conteúdo das informações, é necessário que essa representação seja composta por:

- classes que descrevem os conceitos;
- relacionamentos entre as classes;
- propriedades das classes, e seus valores permitidos;
- axiomas que representam as sentenças para restringirem o conhecimento;
- instâncias de elementos individuais das classes para compor a base de conhecimento.

Todas essas informações podem ser modeladas através de um conjunto de três linguagens complementares entre si: o RDF (Resource Description Framework), o RDFS (RDF Schema), e o OWL (Web Ontology Language).

A grau de formalização de uma ontologia pode variar de um nível mínimo, com a utilização de linguagem natural para representação dos conceitos, até níveis mais altos com a utilização de uma lógica formal. Uschold em (USCHOLD et al., 1999) afirma que em curto prazo, ontologias que carregam menos significados do conhecimento podem ser mais fáceis de aplicar em sistemas funcionais como um sistema de busca, o que não é verdade, por exemplo, em ferramentas de tradução intersistemas. Porém, existem muitos benefícios nas busca com ontologias "mais pesadas", no caso de um repositório de informações semânticamente bem estruturado a precisão da busca é maior.

Em suma, o conceito de um ontologia na Web Semântica e IA é a representação de parte de um universo que permite a compreensão humana e seu processamento por computadores.

Para tanto, isso precisa ser feito de maneira concisa e formal para evitar ambiguidades e indecidibilidade computacional.

2.1.2.3 Proposições

As ontologias na Web Semântica são elaboradas e utilizadas segundo as suposições de que o conhecimento é distribuído ao longo da Web, e por isso podem possuir diferentes definições e referências para um mesmo conceito. Levando em conta essas diferenças, as seguintes proposições foram adotadas segundo (HEBELER et al., 2009):

Proposição de Mundo Aberto Afirma que a veracidade de uma declaração é independente se ela é conhecida ou não, ou seja, não saber que uma declaração é explicitamente verdadeira não implica que ela é falsa. Essa proposição tem um impacto no modo como as informações são modeladas e interpretadas dentro dos sistemas, pois a proposição tradicional, de Mundo Fechado, é exatamente o oposto disso, em que o não conhecimento da verdade de uma declaração determina, necessariamente, a sua falsidade.

Proposição de Nomes Não Únicos Afirma que não é possível assumir que recursos que são identificados por diferentes URIs são obrigatoriamente diferentes, a menos que esse fato seja explicitamente declarado. O que também difere da abordagem tradicional, em que o conceito de chaves primárias em uma base de dados designa, necessariamente, registros diferentes.

2.2 Linguagens de Descrição

Atualmente as tecnologias disponíveis para a construção de aplicações para a Web Semântica ainda não atingiram todo o seu potencial, mas o W3C de Tim Berners-Lee está trabalhando arduamente para isso. Desde a sua fundação em 1994, o consórcio W3C vem definindo padrões de linguagens e protocolos na tentativa de melhorar as formas de troca de dados na rede mundial.

Algumas definições foram criadas pensando nas aplicações com o apelo semântico, ao passo que outras definições, de aplicação mais geral da Internet, servem de base para as primeiras. Nos próximos tópicos serão apresentadas as tecnologias da Web Semântica utilizadas pelo sistema Semantic Lattes ao longo desse projeto.

2.2.1 XML

É um formato baseado em texto utilizado para a representação de informações estruturadas. É originado do padrão SGML, o mesmo padrão que deu origem ao formato HTML, e foi concebido com a intenção auxiliar na caracterização do formato do dados trafegados pela Internet.

Por possuirem um ancestral em comum, o XML e o HTML são semelhantes em alguns aspectos tais como: ambas são linguagens de marcação e utilizam tags para descrever uma informação, além de serem facilmente compreensíveis na leitura humana. Mas a grande diferença entre as duas linguagens é a inexistência de uma estrutura para as informações contidas nos documentos HTML. Por essa razão, o XML é adotado em inúmeras aplicações que envolvem a troca de dados entre sistemas, entre pessoas, entre máquinas, etc.

Uma amostra de uma especificação no formato XML é dada a seguir:

```
<aplicacao>
  <nome>Semantic Lattes</nome>
  <categoria>Buscador</categoria>
</aplicacao>
```

Existem duas maneiras possíveis de representar a definição da estrutura de um documento XML: o DTD (*Document Type Definition*) e o XSD[1] (*XML Schema Definition*), sendo que este último é mais recente e possui mais recursos de formatação e especificação de tipos de dados (*datatypes*). Ambos são definidos pelo W3C e oferecem uma forma de definir propriedades dos dados que podem ser utilizados entre as tags, delimitando características como formato, condição e alcance que os valores podem assumir em determinado campo[2].

Apesar da restrições possíveis na estrutura do XML, a falta de palavras reservadas ou de um vocabulário de termos faz dessa linguagem flexível o suficiente para ser usada em uma ampla variedade de aplicações. Exatamente por ser tão flexível, o XML não consegue atender sozinho as necessidades existentes no compartilhamento de dados semânticos pela Web.

2.2.2 RDF

O RDF é uma linguagem criada para ser utilizada na representação de recursos usando um vocabulário formal. As propriedades desses recursos são especificadas através de declarações (*statements*) que são expressadas através de triplas. Uma tripla RDF obrigatóriamente é

[1] http://www.w3.org/XML/Schema/.
[2] http://www.w3.org/2002/xmlspec/.

formada por um *sujeito*, que faz referência ao recurso em si, um predicado representando a *propriedade* do recurso a ser definida, e um *objeto* determinando o valor dessa propriedade.

Neste caso, o termo *recurso* faz referências a qualquer elemento, abstrato ou não, que se deseja manusear ou representar, podendo receber também a denominação de *entidade*. A *propriedade* da tripla é um tipo especial de recurso dentro do RDF, que descreve as relações entre os demais recursos. Já o *objeto* está representando uma forma de satisfazer a relação estabelecida pela propriedade, podendo tomar a forma de um recurso ou um valor literal.

Ainda, cada recurso RDF deve estar associado a um identificador, uma URI que seja capaz de fornecer uma identificação global única ao recurso, o que é extremamente necessário em um ambiente de conhecimento distribuído como a Internet. Porém, uma URI como

http://www.semanticlattes.com.br/curriculo#Ricardo

pode ser bem extensa, e, por esse motivo, nas notações em RDF normalmente se utiliza abreviações para os nomes em comum. Usando desse artifício, o exemplo anterior poderia ser representado na forma simplificada:

xmlns:base=”http://www.semanticlattes.com.br/curriculo”
...
#Ricardo (prefixo implícito)

Existem algumas formas de se codificar um documento RDF, sendo as principais o XML, Turtle e N3. Apesar do W3C inicialmente ter publicado a definição do RDF em XML[3], o formato Turtle também já é disponibilizado na definição do W3C[4]. Não existem, no entanto, diferenças na capacidade de representação dessas notações. No projeto do Semantic Lattes, a forma RDF/XML foi adotada por ser a mais compatível com as ferramentas utilizadas. Assim uma declaração de que Ricado está sendo orientado por João pode assumir a seguinte forma em RDF/XML:

```
<rdf:RDF xmlns="http://www.semanticlattes.com.br/curriculo#"
         xmlns:base="http://www.semanticlattes.com.br/curriculo"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="#Ricardo">
    <ehOrientadoPor rdf:resource="#João" />
  <rdf:Description>
</rdf:RDF>
```

[3]http://www.w3.org/TR/rdf-syntax-grammar/.
[4]http://www.w3.org/TeamSubmission/turtle/.

É possível ainda utilizar uma representação gráfica das relações existentes entre os recursos RDF. Para isso são utilizados grafos na representação de uma tripla, onde o sujeito e o objeto são nós e a propriedade é o vetor que aponta do sujeito para o objeto. A figura 2.1 apresenta um exemplo do grafo de uma tripla.

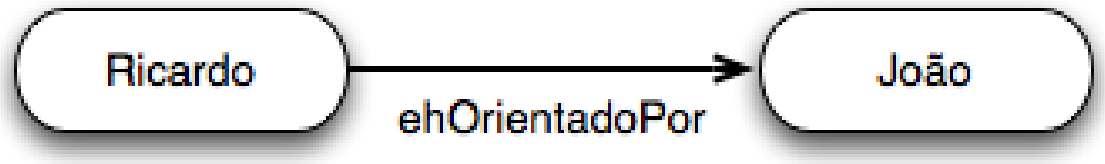

Figura 2.1: *Um exemplo de grafo de tripla RDF/XML.*

O modelo do RDF é muito poderoso na descrição de informações, tornando possível dizer qualquer coisa sobre qualquer coisa (HEBELER et al., 2009). Porém, por ser justamente muito flexível e expressivo, não há um suporte explícito para a especificação do significado semântico por trás dessas descrições.

2.2.3 RDFS

Visando melhor propiciar a expressão semântica das descrições em RDF, o RDFS[5] (*RDF Schema*) foi concebido para fornecer um vocabulário específico para a criação de taxonomias de classes e propriedades, além de possibilitar especificações simplificadas de domínio (domain) e alcance (range) das propriedades. O RDFS é constituído, então, por um conjunto de recursos cujos significados são muito bem definidos e usados de maneira consistente para descrever outros recursos.

Utilizou-se aqui o termo *classe* para referenciar um conjunto de elementos que impõe restrições sobre o que pode ser declarado em um documento RDF (ANTONIOU; HARMELEN, 2008). Uma classe em RDFS se assemelha a uma classe do paradigma de programação de orientação a objetos, pois é possível ter hierarquias de especialização/generalização de classes e criar objetos como instâncias dessas classes. Entretanto, na orientação a objetos, para se adicionar uma nova propriedade é necessário modificar uma classe. Já em RDFS, as propriedades são definidas globalmente de forma que não há um encapsulamento destas como atributos nas definições das classes, o que torna possível a definição de uma propriedade de uma classe sem necessáriamente alterá-la.

As principais classes utilizadas no RDFS são:

[5] http://www.w3.org/TR/rdf-schema/.

- **rdfs:Resource**, é a classe de todas os recursos;
- **rdfs:Class**, é a classe de todas as classes;
- **rdfs:Literal**, é a classe de todos os literais (*string*);
- **rdf:Property**, é a classe de todas as propriedades;
- **rdf:Statement**, é a classe de todas as declarações representadas;

Alguns dos termos presentes na codificação das relações semânticas em RDFS são:

- **rdf:type**, que relaciona um recurso a uma classe, declarando-o como uma instância daquela classe;
- **rdfs:subClassOf**, é quem relaciona uma classe com alguma de suas superclasses. Em função da natureza da especificação da classe, é possível a uma classe possuir mais de uma classe pai;
- **rdfs:subPropertyOf**, que relaciona uma propriedade com algumas das suas superpropriedades.

Existem ainda as restrições semânticas das propriedades como:

- **rdfs:domain**, que especifica um domínio de uma dada propriedade, ditando que qualquer recurso que tem essa propriedade é uma instância das classes referenciadas nesse domínio;
- **rdfs:range**, que especifica o alcance da propriedade, limitando que os valores dessa propriedade são instâncias das classes referenciadas nesse alcance.

Como as declarações em RDFS utilizam os recursos do modelo RDF, ainda é válida a apresentação na forma de grafos. Um exemplo ilustrando alguns dos conceitos explicitados aqui pode ser observado na figura 2.2.

Apesar de proporcionar algumas definições semânticas entre as classes, o RDFS é uma linguagem útil para definir a semântica de domínios particulares de conhecimento, fazendo do RDF Schema uma linguagem primitiva de ontologias.

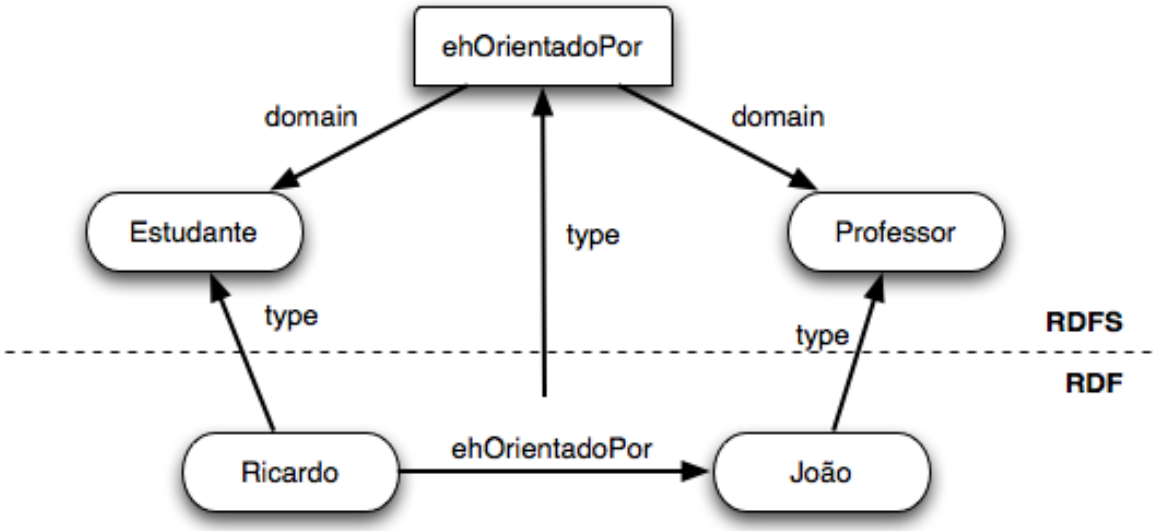

Figura 2.2: *Camadas RDF e RDFS.*

2.2.4 OWL

O OWL[6] (*Web Ontology Language*) é uma extensão do vocabulário RDFS, no qual são incluídos recursos que permitem a construção de ontologias mais complexas voltadas para a Web. São introduzidas novas restrições na estrutura e no conteúdo dos documentos RDF, com o objetivo de tornar as inferências computacionalmente decidíveis. É uma recomendação do W3C e nasceu da união das linguagens de modelagem para ontologias DAML-ONT e OIL, chamada DAML+OIL[7].

O OWL surgiu então com a necessidade de se superar as limitações de expressividade do RDF e do RDF Schema, e de definir uma linguagem mais poderosa que o próprio DAML e OIL (ANTONIOU; HARMELEN, 2008). Algumas propriedades inexistentes no modelo RDF/RDFS que foram listadas por Antoniou são:

- **Escopo local de propriedades**, são restrições específicas no alcance de uma propriedade de uma classe em particular;
- **Disjunção de classes**, uma terminologia que permite declarar que duas classes semelhantes são na verdade disjuntas;
- **Combinações booleanas de classes**, utilização dos operadores como união, intersecção e complemento entre classes na definição de outras classes;

[6] http://www.w3.org/TR/owl-features/.
[7] http://www.daml.org/2001/03/daml+oil-index.html.

- **Restrições de cardinalidade**, são restrições no número máximo ou mínimo de valores possíveis, obrigatórios ou não, em uma propriedade;

- **Características especiais de propriedades**, são asserções como a de que uma propriedade é transitiva, única, ou inversa de outra propriedade.

Os requesitos que uma linguagem de ontologia deve atender e que parecem ser impossíveis de se cumprir são: suporte de *reasoning* eficiente e alta capacidade de facilidade para expressão em uma linguagem tão poderosa quanto uma combinação do RDF Schema com uma lógica completa (full logic). Desses requisitos, o OWL nasceu como uma linguagem tão extensa e com tantas inúmeras estruturas complexas que o Web Ontology Working Group (Grupo de Trabalho de Ontologia Web) do W3C dividiu o OWL em três sub-linguagens, em que cada uma foi arquitetada segundo diferentes perspectivas dos requisitos propostos:

- **OWL-Full** É a linguagem OWL completa e utiliza todas as primitivas da linguagem. É possível ainda com essa linguagem, modificar e combinar essas primitivas com RDF e RDF Schema. A sua principal vantagem é que ela é totalmente compatível com RDF, tanto sintaticamente quanto semânticamente. Sua desvantagem é que a linguagem é tão indecidível quanto é poderosa.

- **OWL-DL** (Description Logic) Pensando em se obter eficiência computacional, uma sub-linguagem da OWL-Full foi formada com restrições na maneira como os construtores do OWL e RDF podem ser usados, assegurando, dessa forma, que a linguagem formada corresponda a uma lógica de descrição. A sua vantagem é a existência de um suporte para *reasoning* eficiente. Já a desvantagem é a perda da compatibilidade total com o RDF, apesar de um documento OWL-DL ainda ser um documento RDF válido.

- **OWL-Lite** Um conjunto ainda mais restrito que o OWL-DL, em que não há classes enumeradas, declaração de disjunção, nem cardinalidade arbitrária. A vantagem é que essa sub-linguagem é mais fácil de compreender (para usuários) e mais fácil de implementar (nas ferramentas). A desvantagem é a sua baixa expressividade.

A escolha de qual das linguagens utilizar, depende da aplicação e da ontologia a ser criada. A escolha entre uma OWL completa e a versão *Lite* depende, por exemplo, da necessidade de expressividade das relações na ontologia, do desempenho da aplicação ou da necessidade de se realizar manutenção periódica.

2.3 Inferências

2.3.1 Princípios Básicos

Inferência basicamente é o processo de extrair novas informações de um conjunto de informações pré-existentes. Na Web Semântica e na Inteligência Artificial esse processo é executado pelos *reasoners*, implementações de software que, são geralmente conectados à um framework ou outras ferramentas, e que utilizam uma base de conhecimento para criar as novas afirmações lógicamente válidas.

A implementação de um sistema de inferência consiste em um conjunto de triplas RDF que normalmente são compostas como na forma:

SE <*um conjunto de triplas RDF*> contém certas triplas
ENTÃO adiciona para <*esse conjunto de triplas RDF*> triplas adicionais

A primeira parte da regra é a condição para a inferência, no qual são analisadas as asserções e os relacionamentos existentes entre as instâncias das classes determinadas pelas triplas RDF. Já a segunda parte é a conclusão da regra, em que são criadas novas relações e/ou asserções a respeito dessas instâncias identificadas na primeira parte. Para ilustrar o conceito, são apresentados alguns exemplos do tipos de inferências possíveis, retirados de (SEGARAN; TAYLOR; EVANS, 2009):

Simples e Determinística

SE (Pedra1 Pesa 1 kilo)
ENTÃO (Pedra1 Pesa 2,2 libras)

Baseada por Restrições

SE (Pessoa1 Tem 17 anos) E (Pessoa1 Vive em São Paulo)
ENTÃO (Pessoa1 Não Pode Dirigir)

Classificação

SE (EmpresaA Está em Santos) OU (EmpresaA está no Rio de Janeiro)
ENTÃO (EmpresaA Está no Litoral)

Conceitos Vagos

SE (Pessoa1 Mede 1,70)
ENTÃO (Pessoa1 É Alta)

Serviços Online

SE (RestauranteA possui Endereço)
ENTÃO (GPS Traça Rota para RestauranteA)

Do ponto de vista matemático, até mesmo realizar consultas em uma base de dados é uma forma de inferência, visto que é possível obter resultados de um massa de dados. A idéia principal é que usando regras com algum conhecimento apropriado e serviços externos, podem-se extrair novas afirmações a partir do conjunto de afirmações corrente.

2.3.2 Inferência em Banco de Dados

2.3.2.1 Estruturação de Dados Tradicional

Numa abordagem utilizando bancos de dados relacionais, teríamos tabelas para representar as entidades e tabelas para explicitar os relacionamentos. Teríamos também aspectos como relacionamentos de um-para-um, um-para-muitos e muitos-para-muitos, sendo que essas relações são implementadas através de chaves primárias e estrangeiras. Todas essas formas de modelar dados são amplamente conhecidas e tradicionalmente consideradas as melhores soluções para o armazenamento de dados.

Num cenário em que temos um professor que trabalha num departamento, normalmente poderíamos buscar esta informação em banco de dados relacional utilizando uma chamada em SQL (*Structured Query Language*), por exemplo:

```
SELECT pessoa.nome FROM pessoa
  INNER JOIN departamento_pessoa dp ON dp.id = pessoa.id
  INNER JOIN departamento ON departamento.id = dp.id
WHERE departamento.name = 'nome do departamento'
```

Caso seja necessário representar a relação com uma Instituição é preciso observar que um departamento pertence a uma instituição e um professor deve estar relacionado com algum departamento dessa instituição, logo ele tem vínculo com ela.

Assim, se quisermos obter quais vínculos institucionais a pessoa possui, teríamos de partir dos departamentos que são pertencentes a uma instituição. Para isso teríamos de incluir mais um `JOIN` que relacionasse departamentos com instituições. Isto é extremamente custoso, pois sabemos que o uso de `JOIN` é uma operação que envolve muito processamento, principalmente se tivermos um grande volume de registros, pois a busca, simplificadamente, tende a varrer todos os registros, sem pensar no conceito de índices.

Para modelar esse conhecimento poderíamos adicionar uma tabela `instituicao_pessoa`, para eliminarmos a necessidade de termos um `JOIN` com departamento. No entanto, perderíamos a relação semântica entre departamento e instituição apesar de nada nos impedir de mantê-la. Note que isso acaba gerando muitos relacionamentos dificultando a manutenção destes registros.

O exemplo acima não possui muita complexidade de conhecimento, mas exige uma complexidade em sua modelagem. Desta forma, uma abordagem mais eficiente seria a utilização de grafos, que veremos na próxima seção.

2.3.2.2 Representação de Conhecimento na Forma de Grafos

O problema apresentado na seção 2.3.2.1 poderia ser modelado na forma de um grafo conforme apresentado na figura 2.3.

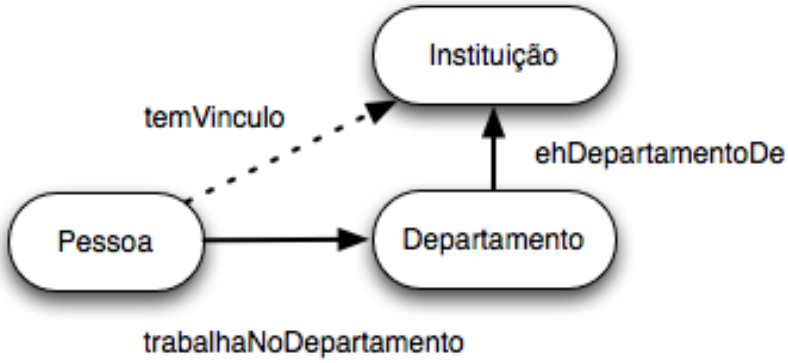

Figura 2.3: *Grafo para um conjunto de relações e consequências.*

Note que não precisaríamos explicitar que uma pessoa está vinculada a instituição se ela trabalhar no departamento. Esta informação poderia ser *concluída* a partir das descrições de que um departamento é de uma instituição e, por consequência, quem trabalha nele está vinculada a uma instituição.

Com a geração dos grafos, uma relação direta é estabelecida entre as entidades, o que elimina a necessidade de operações de `JOIN`s. Basta partir de um nó, no caso pessoa, e navegar pelas relações do grafo, como *temVinculo*, que é uma consequência lógica, e então obter seus vínculos instituicionais.

Na Web Semântica, essas declarações são estruturadas em triplas utilizando a tecnologia RDF da W3C, onde temos o objeto, propriedade e valor como sendo o nó, relação e destino, respectivamente.

2.3.3 Linguagens de Consulta

2.3.3.1 SPARQL

SPARQL[8] é uma linguagem de consulta para a Web Semântica, independente do banco de dados subjacente. Diferentemente de uma linguagem de busca em XML, que é focada em um nível mais baixo de abstração, o SPARQL consegue trabalhar com as outras tecnologias da Web Semântica, incluíndo RDF, RDF Schema e OWL. É capaz de compreender não apenas a sintaxe, mas também o modelo de dados do RDF e a semântica dos vocabulários do RDFS e OWL.

Existem outras linguagens de busca que também reunem tais qualidades, sendo inclusive mais velhas e maduras em suas implementações como o SeRQL (*Sesame RDF Query Language*) e o RDQL (*RDF Data Query Language*), mas não existem, atualmente, muitas ferramentas que as utilizam. Além disso, o SPARQL é uma recomendação do W3C[9], tem um amplo suporte da comunidade de Web Semântica e, apesar da sua publicação oficial ter ocorrido somente em 2008, já existe um número considerável de *endpoints* públicos disponíveis. Um *endpoint* no caso, é um serviço que aceita e processa consulta SPARQL e retorna os resultados em formatos que dependem da forma solicitada.

Na Web Semântica, SPARQL é uma linguagem de busca ao mesmo tempo em que é um protocolo, ambos são definidos usando uma implementação baseada em WSDL[10]:

- O procolo define a forma de comunicação entre uma aplicação cliente SPARQL com um *endpoint* SPARQL.
- Na linguagem de busca são definidas as sintaxes necessárias para a construção das consultas. Esta é a característica interessante neste trabalho.

A linguagem de busca SPARQL é apoida no reconhecimento de padrões de grafos, logo, o mais simples deles é uma única tripla. Essa tripla é semelhante a uma tripla RDF mas com a possibilidade de uma variável no lugar de algum sujeito, predicado ou objeto. Combinando essas triplas da *query*, obtém-se o padrão de grafo que se deseja buscar. Para ilustrar esse conceito, considere o exemplo de uma query SPARQL a seguir.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

[8] http://www.w3.org/TR/rdf-sparql-query/.
[9] http://www.w3.org/TR/rdf-sparql-protocol/.
[10] http://www.w3.org/TR/wsdl/.

```
PREFIX sl: <http://www.semanticlattes.com.br/curriculo#>
SELECT ?a ?e
WHERE
{
  ?a rdf:type sl:Professor ;
  sl:ehOrientadorDeDoutorando ?e.
}
```

Esta *query* obtém todos os grafos com triplas que possuírem um predicado `rdf:type` relacionado ao objeto `sl:Professor` e um predicado `sl:ehOrientadorDeDoutorando` relacionado a qualquer recurso, ou seja, serão retornados todos os professores da base de dados que são orientadores de Doutorado. É importante notar que, assim como é feito no RDF, é possível adotar abreviações para representar *namespaces* extensos, fato que facilita a leitura e compreensão da consulta.

Não há, no entanto, um compromisso do SPARQL em suportar a semântica das informações. Assim, a busca de padrões de grafos é somente tratar os dados disponibilizados pelo sistema por trás do *endpoint*, que pode ser ou não capaz realizar as inferências sobre seus dados. Em caso positivo o resultado esperado será alcançado, e em caso contrário apenas os registros que foram explicitamente declarados serão retornados.

Uma consulta SPARQL expressa objetivos de alto-nível e são mais fáceis de se estender para bases de dados que não são inicialmente previstas. Essa tecnologia é capaz de superar as limitações das buscas locais e dos formatos simples.

2.4 Conclusões

Este capítulo apresentou, de forma resumida, os principais aspectos relevantes ao projeto.

Nos capítulos seguintes, será especificado o sistema Semantic Lattes, que utiliza conceitos desta área para a consulta de informações acadêmicas na base Lattes.

3 SEMANTIC LATTES

3.1 Arquitetura do Sistema

A arquitetura do Semantic Lattes está apresentada na figura 3.1, que segue um modelo de três camadas divididos em apresentação, aplicação e dados.

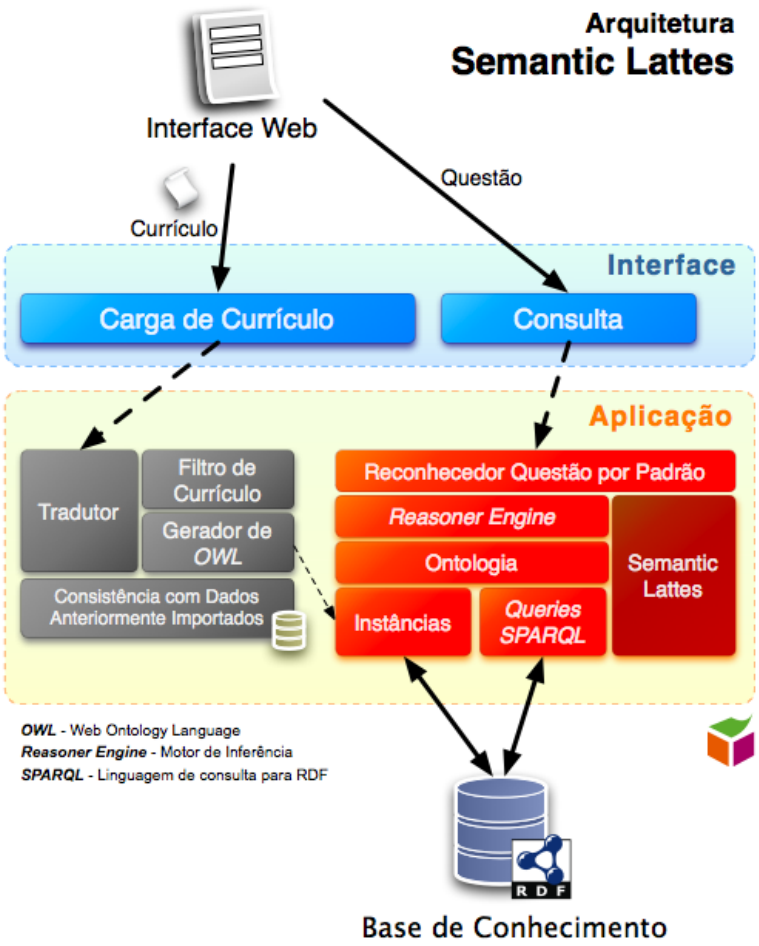

Figura 3.1: *Arquitetura do Semantic Lattes.*

- A camada de **Apresentação** corresponde às telas do sistema para a entrada das perguntas, apresentação dos resultados da consulta e importação de currículos.

- A camada de **Aplicação** reúne funções como traduzir as informações dos currículos para a geração das instâncias, realizar as inferências com os dados obtidos dessas instâncias e reconhecer as consultas recebidas na camada da interface.
- A camada de **Dados** corresponde ao armazenamento dos conjuntos de dados gerados dos currículos, contendo todas as informações originais.

3.1.1 Camada de Apresentação

Na camada da Apresentação, a interface Web permite que consultas de interesse sejam requisitadas ao sistema para então serem processadas na aplicação. A exibição dos resultados corresponde a outra tela do sistema onde são listados todas as instâncias encontradas na base de conhecimento.

Outra tela da interface provê a carga de currículos em XML por meio do envio destes documentos que serão então processados na aplicação para gerar novas instâncias, e então adicionados à base de conhecimento.

3.1.2 Camada de Aplicação

A camada de Aplicação é formada por um conjunto de módulos que se inter-relacionam visando o tratamento das relações semânticas entre os dados dos currículos.

Podemos dividir esta camada em dois grandes módulos busca e importação, como ilustrados na figura 3.1.

3.1.2.1 Módulo de Busca

Reconhecedor Questão por Padrão É o módulo que recebe as requisições de consultas provenientes da camada de Apresentação. Neste módulo as questões são resolvidas extraindo-se os parâmetros de entrada e então repassadas por meio da sua consulta SPARQL associada.

Motor de Inferência (Reasoner Engine) O módulo central do sistema é o motor de inferências, que é responsável por determinar novos relacionamentos entres as instâncias existentes na base de dados, descritas de forma semântica segundo as Ontologias definidas no sistema.

Ontologias Armazenam as definições e os relacionamentos existentes nos domínios de conhecimento sobre o qual o sistema está trabalhando no formato OWL.

Instâncias Contêm as instâncias das ontologias presentes no sistema relativas às informações dos currículos e dos resultados das inferências realizadas pelo Reasoner Engine.

Queries SPARQL Correspondem ao conjunto de consultas construídas para cada tipo de consulta definida para o sistema.

Semantic Lattes É o módulo responsável pela integração de todos os módulos apresentados anteriormente. Aqui há um encapsulamento da ferramenta Jena, que provê os interfaces de comunicação com o motor de inferência Pellet e a ferramenta de armazenamento de dados TDB, que serão apresentadas na seção 5.1.

3.1.2.2 Módulo de Importação

O módulo Tradutor é responsável pela conversão dos documentos no formato XML para instâncias em OWL de acordo com as descrições da ontologia. Neste ponto, algumas consistências são feitas como persistência de identificadores URI de instâncias para possibilitar que não sejam recriadas instâncias iguais. Abaixo temos a divisão de tarefas para este módulo:

- **Filtro de Currículo:** o filtro é apenas uma seleção dos dados mais relevantes do currículo para este projeto. Em termos de publicações existem diversos tipos, mas apenas os de importância para este projeto foram escolhidos, como especificados na seção 5.2.
- **Gerador de OWL:** gera os arquivos de saída em OWL contendo as instâncias da ontologia, geradas a partir das informações relevantes do currículo.
- **Consistência de Dados (simplificado):** neste módulo existe uma verificação simples das entidades já carregadas no sistema de modo a fazerem uso dos URI corretamente. Por exemplo, dois currículos distintos apresentam referências para uma pessoa de nome João da Silva. Nesta ocorrência, seria esperado que o mesmo URI fosse gerado para representar a mesma entidade. A consistência utiliza parâmetros dos dados do currículo para minimizar esses erros, ainda que não seja uma técnica muito eficiente.

3.1.3 Camada de Dados

A camada de Dados tem função de armazenamento das instâncias geradas a partir dos currículos, segundo a representação em grafos utilizadas para representar os conjuntos de dados. O sistema de armazenamento utiliza um tipo de banco de dados específico para armazenar as triplas RDF chamado TDB. Essa base de dados, que possui uma estrutura diferente dos tradicionais bancos relacionais, é um projeto dentro do próprio Jena e é integrado por meio de uma API em Java.

3.1.4 Fluxos de Informações

Na figura 3.2, são apresentados os fluxos de informações do sistema, por todos os módulos apresentados anteriormente, para as funcionalidades de importação de currículos e consulta ao sistema.

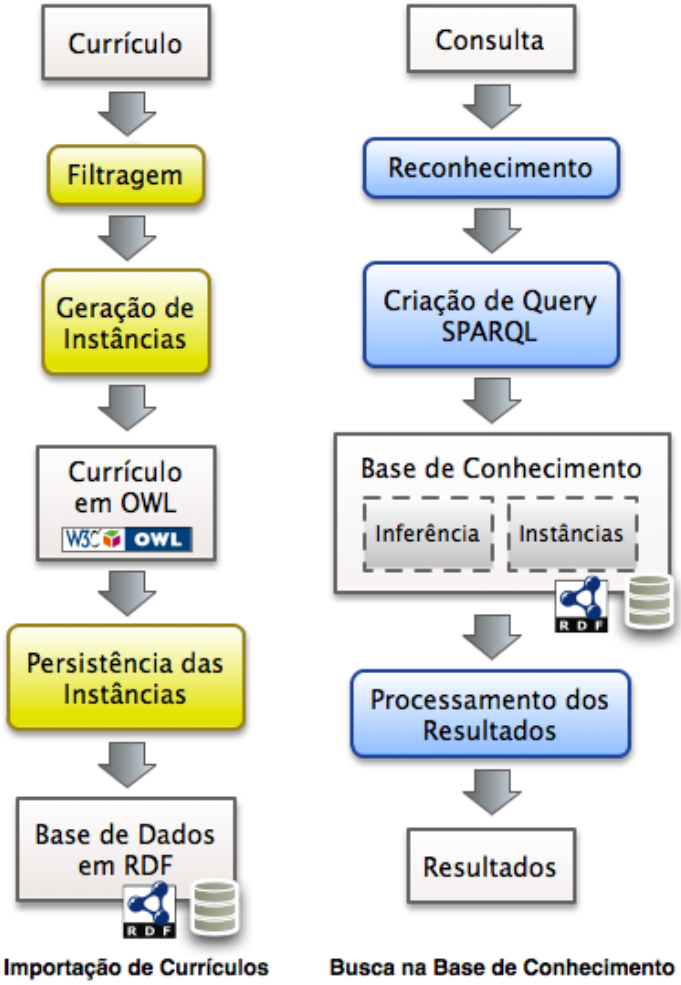

Figura 3.2: *Fluxos de cada módulo da aplicação.*

3.2 Detalhamento das Funcionalidades

O sistema Semantic Lattes possui as seguintes funcionalidades:

- Carga de currículos ao sistema a partir de documentos XML;
- Reconhecimento das questões inseridas pelo usuário;
- Consulta à base de currículos.

3.2.1 Carga de Currículos no Sistema a partir de Documentos XML

Essa primeira funcionalidade diz respeito à forma como as novas informações serão carregadas no sistema. A única forma possível é através da introdução de currículos escritos em XML, estruturados de acordo com as padrões estabelecidos pela Comunidade CONSCIENTIAS-LMPL[1].

A partir destes documentos, um mecanismo tradutor realiza a identificação dos elementos relevantes ao sistema e gera um documento OWL contendo apenas as instâncias das classes definidas na ontologia. Por fim, essas informações são incorporadas à base de conhecimento existente no Semantic Lattes na forma de triplas RDF. O currículo poderá ser carregado e, em caso de novas instâncias ou relacionamentos, estes serão incorporados na base de conhecimento. Contudo, a manutenção destas informações não está prevista como funcionalidade deste sistema.

A tradução será um processo de um passo sem conferência das informações. Este é um dos principais problemas hoje dos currículos. As informações não possuem um vínculo direto com outros currículos. No caso de publicações, por exemplo, não existe um identificador único que possa ser utilizado, sendo necessário o uso nome da publicação como um parâmetro de identidade. Entretanto, cada autor do currículo pode escrever livremente o nome da publicação, ou seja, existirá uma inconsistência de informações e as referências serão consideradas distintas quando equivalentes.

Este problema de inconsistência de informações constitui outro aspecto do sistema a ser abordado, mas que vai além do escopo deste projeto, podendo fazer parte de futuras implementações do sistema.

[1] http://lattesextrator.cnpq.br/lmpl.

3.2.2 Reconhecimento das Questões Inseridas Pelo Usuário

A segunda funcionalidade do Semantic Lattes refere-se à maneira que será realizado o reconhecimento da pergunta inserida pelo usuário. Essas perguntas foram pré-definidas com base na lista de questões levantadas na seção 4.2. São perguntas na forma:

> Quais são as publicações em conferências pelo professor do departamento A no ano 2009?

Para esta questão temos como parâmetros os termos 'A' e '2009'. O reconhecimento da questão é feito por meio de expressões regulares de forma direta, sem possibilidades de variação da pergunta, como por exemplo a eliminação do termo "são", que manteria o mesmo valor semântico da pergunta.

O reconhecimento de linguagem natural seria outra extensão do projeto para este sistema. Na possibilidade de ampliação da gama de variações para um mesma pergunta, o sistema poderia se tornar muito mais poderoso naquilo em que se propõe a fazer, que é a precisão na consulta de interesse. Porém, seria um trabalho extremamente complexo que, novamente, não cabe a ser tratado neste projeto.

3.2.3 Consulta à Base de Currículos

A última funcionalidade corresponde ao elemento principal do sistema que é a obtenção dos resultados de acordo com a pergunta realizada pelo usuário. Essa funcionalidade necessita das funcionalidades anteriores, pois fará uso da base de conhecimento e do reconhecimento da pergunta.

Uma vez reconhecida a pergunta, a consulta correspondente será associada à sua query em SPARQL de consulta, à base de conhecimento e aos resultados obtidos que serão exibidas ao usuário na interface Web do sistema.

4 METODOLOGIA E DESENVOLVIMENTO

4.1 Introdução

Na metodologia adotada de desenvolvimento das ontologias foi tomada como referência o modelo apresentado em (BRANDãO; LUCENA, 2002), cujos passos estão listados abaixo:

- busca por ontologias que pudéssemos reutilizar;
- definição das questões de competência;
- definição dos conceitos básicos da nossa ontologia;
- definição de conceitos mais refinados a partir dos conceitos básicos;
- definição dos relacionamentos entre os conceitos;

A primeira recomendação baseia-se no princípio da Web Semântica de que o conhecimento é distribuído e compartilhado, e que as ontologias podem ser tomadas como referências em outros domínios, além do seu original. No entanto, na etapa de construção das ontologias desse projeto não foi utilizado nenhum conceito originário de outra ontologia, pois não havia um conjunto de classes suficientemente notável que justificasse a utilização de outras ontologias.

Em seguida, a definição das questões de competência se deve a limitação do escopo da ontologia, o que é extremamente importante em domínios muitos extensos e que possuem inúmeros conceitos e relações como a do Lattes, por exemplo.

Os últimos três passos consistem na construção gradual do modelo da ontologia, atendendo os requisitos identificados com as questões de competência. Começando com a definição dos conceitos básicos, seguindo pelo refinamento destes e terminando com o tratamento dos relacionamentos existentes entre esses conceitos.

4.2 Questões de Competência

As questões de competência estão relacionadas com a limitação do escopo da ontologia dentro do domínio, ou seja, a ontologia representa apenas a parte do domínio que atende a resolução das perguntas elaboradas. Esse tipo de abordagem é extremamente útil e torna o construção das ontogias mais eficiente, principalmente quando localizadas em domínios muito extensos como é o caso do Currículo Lattes.

4.2.1 Processo de Elaboração

A elaboração das perguntas relevantes ao sistema foi feita com a ajuda dos especialistas do domínio, que levantaram as questões pensando nos relatórios de produtividade acadêmica dos Departamentos.

O papel dos especialistas do domínio, neste caso, foi dado aos orientadores desse projeto, observando a extensa experiência que eles adquiriram com a utilização do sistema e do seu profundo conhecimento das relações existentes entre os conceitos embutidos nos documentos do sistema Lattes.

4.2.2 Lista das Questões

Algumas questões de competência que devem ser respondidas pelas ontologias de domínio são as seguintes:

1. Quais são os professores/pesquisadores do Departamento <departamento>?
    - Quais professores/pesquisadores do Departamento <departamento> possuem registro de patente?
    - Quais professores/pesquisadores do Departamento <departamento> publicaram livros?
    - Quais professores/pesquisadores do Departamento <departamento> publicaram capitulos de livros?
2. Quais professores/pesquisadores do Departamento <departamento> publicaram no ano <ano>?
    - Quais professores/pesquisadores do Departamento <departamento> publicaram em conferências no ano <ano>?

- Quais professores/pesquisadores do Departamento <departamento> publicaram em conferências internacionais no ano <ano>?
- Quais professores/pesquisadores do Departamento <departamento> publicaram em conferências nacionais no ano <ano>?

3. Quais são os professores/pesquisadores do Departamento <departamento> que publicaram em co-autoria com o <nome pesquisador/professor>?
4. Quais são as publicações dos professores/pesquisadores do departamento <departamento>?
5. Quais são as publicações em periódicos dos professores/pesquisadores do departamento <departamento>?
6. Quais são as publicações em que o <nome pesquisador/professor> figura como 1o autor?
7. Quais são as publicações em periódicos do professor/pesquisador <nome pesquisador/professor>
8. Quais são as publicações em conferências dos professores/pesquisadores do Departamento <departamento>?
   - Quais são as publicações em conferências internacionais dos professores/pesquisadores do Departamento <departamento>?
   - Quais são as publicações em conferências nacionais dos professores/pesquisadores do Departamento <departamento>?
9. Quais são as publicações em conferências internacionais do professor/pesquisador <nome pesquisador/professor>?
   - Quais são as publicações em conferências nacionais do <nome pesquisador/professor>?
10. Quem escreveu a publicação <nome publicação>?
11. Quem são os orientandos do <nome pesquisador/professor>?
    - Quem são os orientandos de IC do <nome pesquisador/professor>?
    - Quem são os orientandos de Mestrado do <nome pesquisador/professor>?
    - Quem são os orientandos de Doutorado do <nome pesquisador/professor>?
12. Quais alunos/orientandos possuem publicações com seu orientador?
13. Quais são os livros publicados por professores/pesquisadores do Departamento <departamento>?

14. Quais são os livros publicados pelo <nome pesquisador/professor>?

15. Quais são os capítulos de livros publicados por professores/pesquisadores do Departamento <departamento>?

16. Quais são os capitulos de livros publicados pelo <nome pesquisador/professor>?

17. Quais são as teses de doutorado finalizadas sob orientação dos professores/pesquisadores do Departamento <departamento>?

    - Quais são as teses de doutorado em andamento sob orientação dos professores/pesquisadores do Departamento <departamento>?

18. Quais são as teses de doutorado finalizadas sob orientação do <nome pesquisador/professor>?

    - Quais são as teses de doutorado em andamento sob orientação do <nome pesquisador /professor>?

19. Quais são as dissertações de mestrado finalizadas sob orientação dos professores/pesquisadores do Departamento <departamento>?

    - Quais são as dissertações de mestrado em andamento sob orientação do professores/pesquisadores do Departamento <departamento>?

20. Quais são as dissertações de mestrado finalizadas sob orientação dos <nome pesquisador /professor>?

    - Quais são as dissertações de mestrado em andamento sob orientação do <nome pesquisador/professor>?

21. Quais são os artigos Qualis <estrato Qualis> publicados entre o ano <ano> e ano <ano>?

Essas questões também terão outros propósitos, além da restrição na elaboração da ontologia. Elas servirão de base também para a padronização da entrada do sistema final, em que o usuário do sistema fará requisições diretamente relacionadas com os conteúdo questões acima.

4.3 Desenvolvimento das Ontologias

4.3.1 Abordagem Adotada de Questões

No desenvolvimento do sistema foram descritas duas ontologias para a representação dos conceitos do domínio do currículo Lattes do CNPq e do domínio das avaliações do Qualis da CAPES. A construção das ontologias foram orientadas pelas questões definidas na seção 4.2, sendo o ponto de partida para a definição dos conceitos e relações necessárias.

Seguindo os passos da metodologia exposta em (BRANDãO; LUCENA, 2002), esta etapa consiste em partir das perguntas de competência para limitar a criação das ontologias a parte do domínio que será útil nas tarefas do sistema. O processo consiste em identificar, classificar e relacionar os conceitos do domínio de atuação.

Esse processo inicia-se com a identificação das entidades expostas na questões, procurando representar essas entidades em classes que possuem propriedades em comum. Em seguida, se possível, são dividas em grupos e, então, os relacionamentos e as ligações semânticas entre essas classes são determinados.

Para explicitar melhor a conexão existente entre os conceitos dessa ontologia e as perguntas elaboradas, é exemplificada logo abaixo a definição de parte da ontologia a partir de algumas das questões elaboradas.

Sejam as questões de competência:

1 Quais são os Professores/Pesquisadores do Departamento <departamento>?

2 Quais Professores/Pesquisadores do Departamento <departamento> publicaram no ano <ano>?

4 Quais são as Publicações dos Professores/Pesquisadores do Departamento <departamento>?

Para esse conjunto de questões, pode-se identificar por simples inspeção, as entidades presentes como sendo: Professor, Pesquisador, Departamento, Publicação e Ano. Essas entidades básicas já são candidatas a classes na ontologia.

Observando essas entidades, é possível já criar algumas classificações que reflitam os conceitos existente no domínio, e, quando necessário, até inserir novas entidades que ajudem a formar uma taxonomia. No exemplo, é possível criar uma classe-pai denominada Pessoa que tenha os filhos Professor e Pesquisador.

Uma vez gerada a taxonomia, são definidas as relações entre essas entidades, novamente observando os conceitos e relações existentes no domínio em questão, conforme mostra a figura 4.1. Dessa forma, na ontologia estarão representados:

- A primeira relação já está explicita na questão 1. Um Departamento está conectado com um Professor e um Pesquisador através da relação *possuiFuncionarios* e a relação inversa *trabalhaNoDepartamento*;
- Uma Publicação é ligada a um Professor e um Pesquisador através da relação *publicadoPor* e a inversa por *ehAutorDe*;
- Uma Publicação possui ainda uma *data property* chamada Ano;

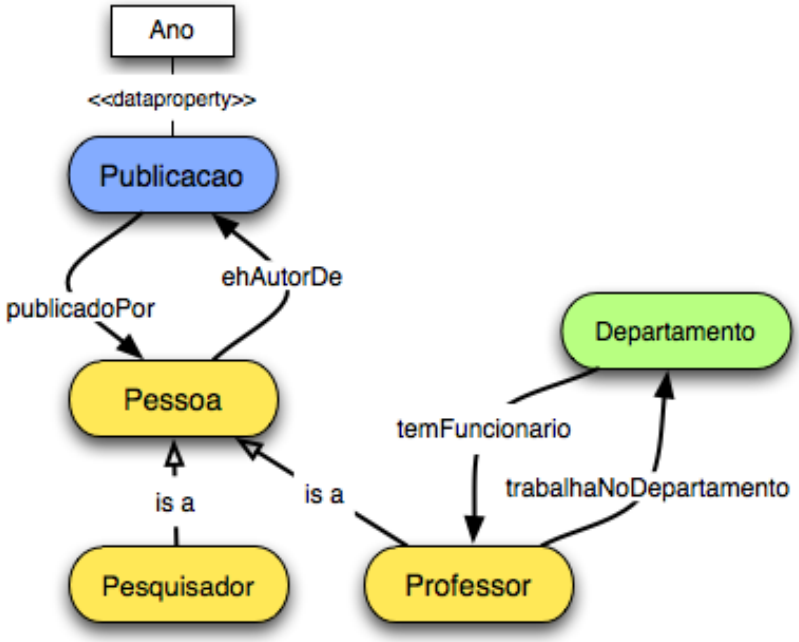

Figura 4.1: *Relações entre conceitos de um domínio em questão.*

Apenas com essas definções e relações na ontologia, o motor de inferências já é capaz de encadear essas regras e inferir sobre os dados para a realização das algumas consultas específicas para a resolução das questões originalmente propostas.

O processo de criação do restante da ontologia se dá com a execução dessas etapas, aplicadas à lista completa de questões de competência apresentada na seção 4.2.2. Uma descrição dos principais conceitos das ontologias desenvolvidas é apresentada a seguir:

4.3.2 Ontologias do Lattes

Para a construção da ontologia da base Lattes, foi necessário o envolvimento dos especialistas do domínio, que neste caso são os orientadores do projeto. Tendo isto como premissa, foram levantados os conceitos chaves do domínio a fim de limitar o escopo inicial e focar naquilo que teria maior prioridade inicial. Assim, foram escolhidos os seguintes elementos do currículo:

- Autor do currículo;
- Trabalhos em Conferências/Eventos;
- Artigos Publicados em Periódicos;
- Livros e Capítulos;
- Patentes;
- Orientações Concluídas e Em Andamento.

Autor do Currículo Corresponde aos dados gerais do autor do currículo. Aqui são obtidas as informações como o nome completo, nome para citação, vínculos institucionais, departamentos de atuação e títulos.

Os valores possíveis para um título são: *Graduado*, *Mestre* e *Doutor*. Na definição desses conceitos foi preciso tomandar cuidado para não assumir uma metonímia como uma definição. Por exemplo, Doutor pode vir a ser usado como uma generalização da classe Pessoa e não como um título que foi obtido obtido durante seu histórico acadêmico.

O autor é uma pessoa e pode exercer diferentes papéis como sendo um *Estudante*, *Professor* e *Pesquisador*. Dentre as diferentes possibilidades para uma generalização da classe Pessoa, a adotada neste domínio foi a das atividades que o indivíduo exerce.

Para a modelagem do conceito de Pesquisador, basta que ele possua ao menos uma publicação acadêmica. Esta restrição foi definida pois não é clara nos currículos a definição de "pesquisador", logo foi considerado que qualquer um que gere um produto de pesquisa, por exemplo um trabalho acadêmico, é um pesquisador.

Com relação à Professor, foi possível extrair a informação a partir de seus vínculos profissionais. Porém, para o professor que está trabalhando em algum departamento, caso não esteja explícito no seu currículo a sua atuação profissional, pode ser inferido ele é um professor por justamente por estar relacionado a um departamento.

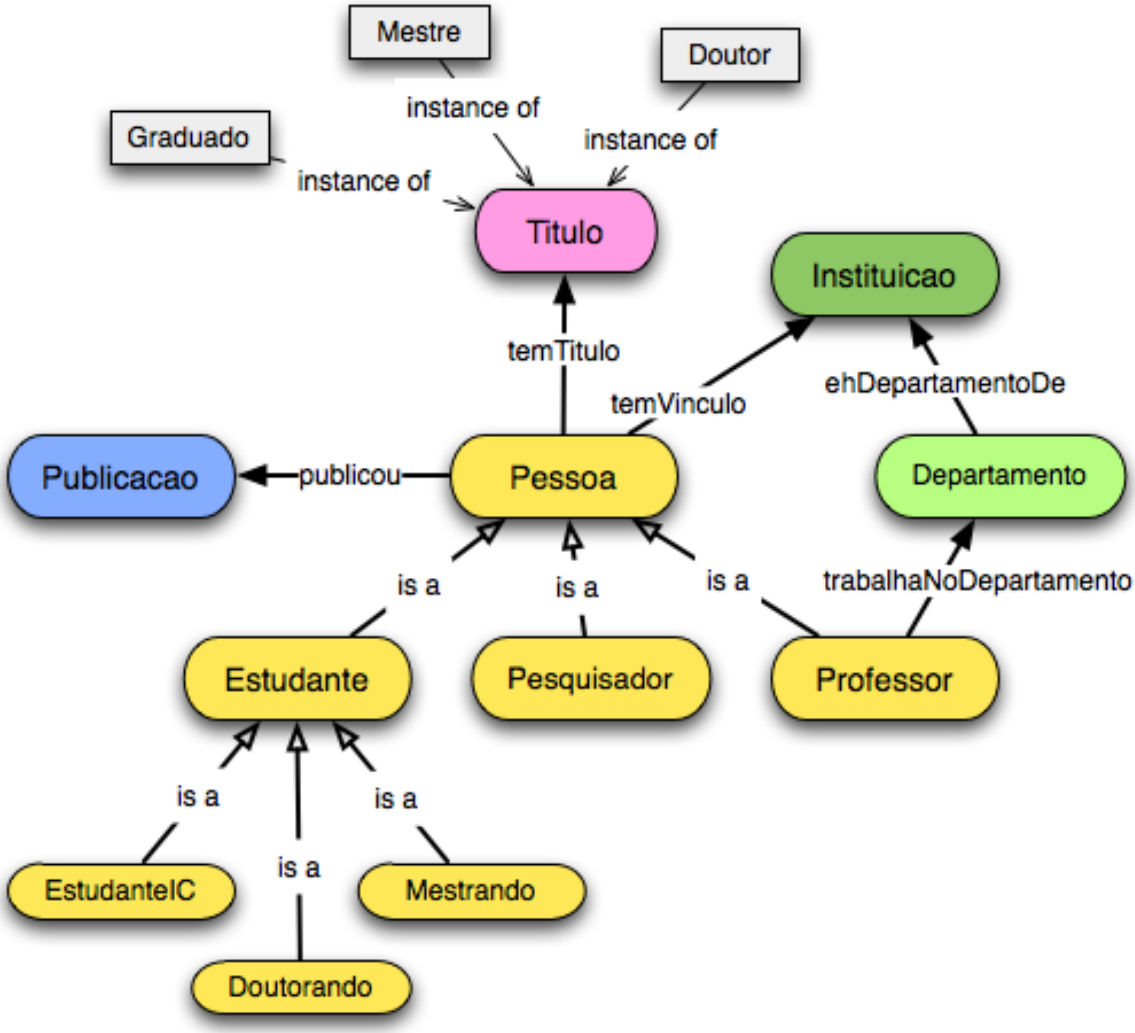

Figura 4.2: *Ontologia Lattes: Conceitos relacionados à Pessoa.*

No caso do conceito Estudante, as informações são obtidas dos currículos de seus orientadores, visto que o trabalho ainda não concluído (dissertação de mestrado ou tese de doutorado) não é declarado no currículo do estudante. Logo, o processo também pode se dar por inferência através da verificação de que uma pessoa está sob alguma orientação em andamento, como veremos mais adiante.

Todo autor de currículo possui publicações que foram definidas como: artigos em periódicos, artigos em conferências (anais), livros e capítulos, patentes, organizações de eventos e entre outros, como detalhado a seguir.

Trabalhos em Conferências/Eventos Deste elemento após uma extensa elaboração, elegeram-se os conceitos: *Conferência*, *Edições de Conferência*, *Anais* e *Artigos em Anais*.

O conceito de uma Conferência faz referência ao evento que pode ocorrer todos os anos, na medida em que cada ocorrência é identificada como sendo uma Edição de Conferência. Esses dois conceitos estão relacionados pelas propriedades *possuiEdicaoConferencia* e *ehEdicaoDaConferencia* que permitem a navegação pelas edições a partir de uma conferência e a determinação de uma conferência a partir de suas edições.

Seguindo o mesmo raciocínio, temos os Anais (*Proceedings*) representando as publicações de cada um das Edições de Conferência, e os Artigos em Anais representam os artigos publicados em cada um dos Anais. Assim, cada Artigo em Anais se encontra relacionado com seu respectivo Anais pelas propriedades *possuiArtigoPublicadoEmAnais* e *ehPublicadoEmAnais*. Os conceitos de Anais podem ser classificados como uma generalização da classe *Publicação*.

Ainda, o conceito Anais e Edição de Conferência estão ligados por *ehPublicadoEmEdicaoConferencia* e *possuiAnais*. Todos esses relacionamentos podem ser observados na figura 4.3.

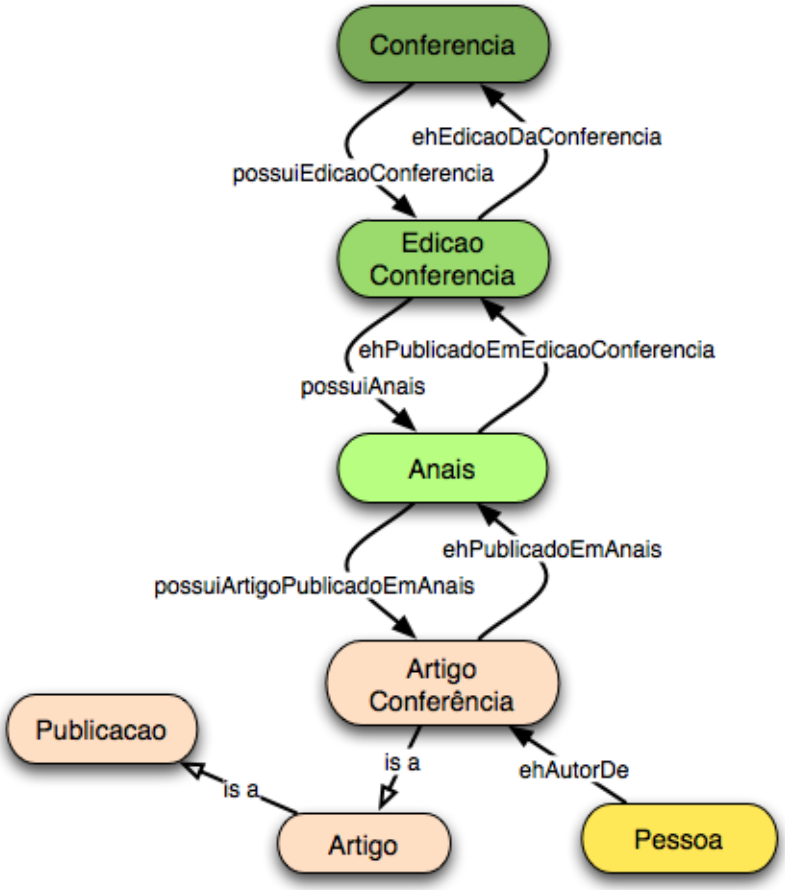

Figura 4.3: *Ontologia Lattes: Conceitos relacionados à publicações em Conferências.*

Artigos Publicados em Periódicos Neste item, foram identificados os conceitos *Periódicos*, *Edições de Periódico* e *Artigos*.

Analogamente aos Trabalhos em Conferências, um Periódico foi modelado como tendo uma frequência fixa, e que um gera um produto denominado *Edição de Periódico*. Nas Edições de Periódico estão publicados os Artigos.

A relacionamento entre um Periódico e uma Edição de Periódico é dado por *possuiPeriodico* e *ehEdicaoDePeriodico*. As relações *ehPublicadoEmEdicaoPeriodico* e *possuiArtigoPublicadoEmEdicaoPeriodico* relaciona Edição de Periódico e Artigos, conforme ilustrado na figura 4.4.

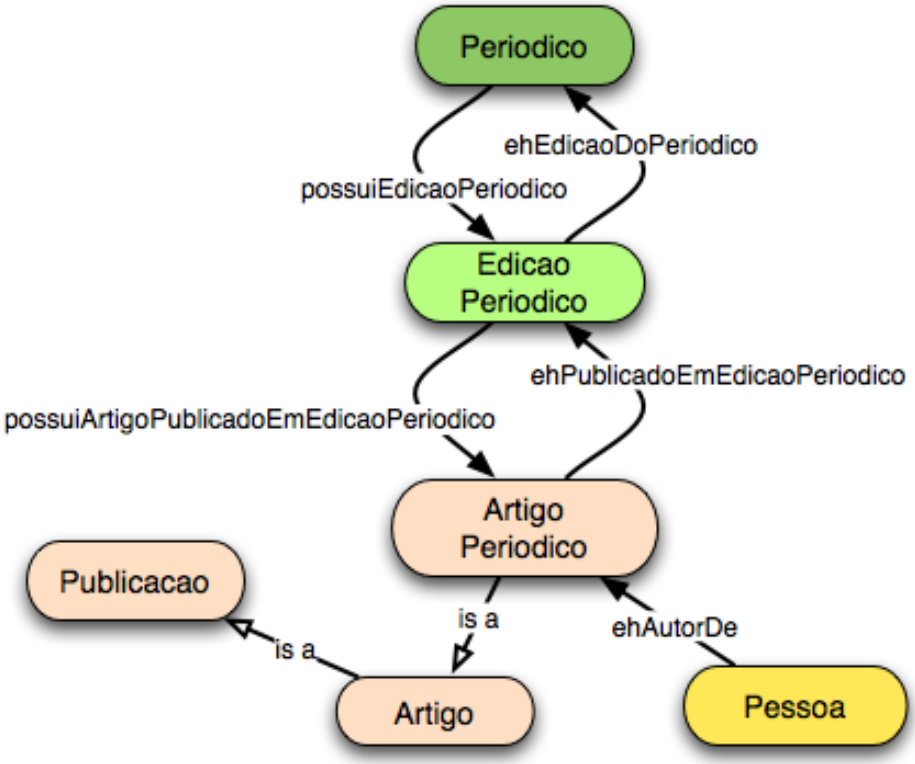

Figura 4.4: *Ontologia Lattes: Conceitos relacionados à publicações em Periódicos.*

Livros e Capítulos Os *Livros* e *Capítulos* são outro tipo de publicação e, portanto, são generalizações da classe *Publicação*. Em geral, livros podem ter autores ou editores. Neste segundo caso, os editores são responsáveis por escolher e organizar os diferentes capítulos. Sendo que cada capítulo pode ter autor(es) distinto(s) dos editores, representados na figura 4.5.

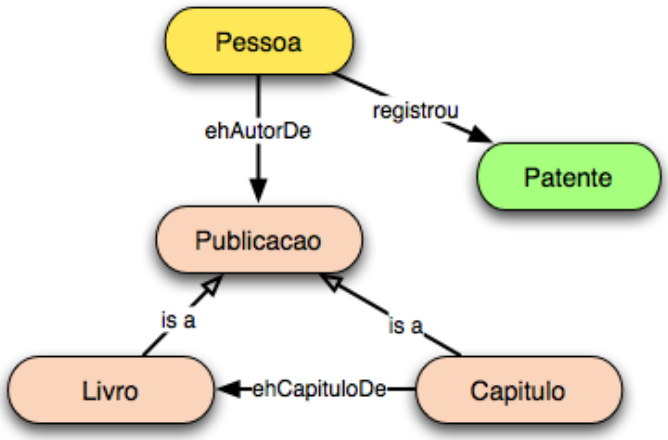

Figura 4.5: *Ontologia Lattes: Conceitos de Livro, Capítulo e Patente.*

Patentes A classe *Patente* é outro produto possível de pesquisa, e é, neste caso, irmã da classe *Publicação*. Estes produtos são relacionados com a classe *Pessoas* através das propriedades *registradoPor* e *registrou*.

Orientações Concluídas e Em Andamento As orientações foram modeladas como um conceito que relaciona diversos outros conceitos como *Orientador*, Orientando ou Orientado e o *Trabalho* em orientação.

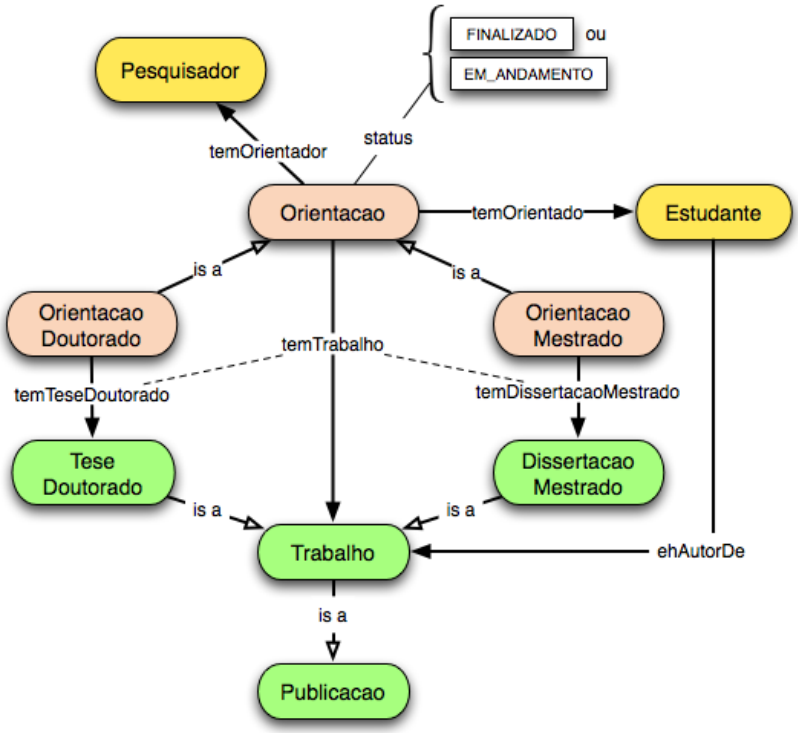

Figura 4.6: *Ontologia Lattes: Conceitos relacionados à Orientação.*

O Orientador não é um conceito explícito na ontologia, é uma informação obtida a partir do conceito central que é a Orientação. Foram distinguidos dois tipos de orientações para este trabalho, Orientações de *Mestrado* e de *Doutorado*. As orientações podem estar finalizadas ou em andamento. Segundo pode-se observar na figura 4.6.

No caso de uma orientação de mestrado, o trabalho corresponde a uma *dissertação de mestrado* e uma *tese de doutorado* para orientação de doutorado.

4.3.3 Ontologia do Qualis

A ontologia do Qualis foi projetada para definir os estratos de qualificação de uma publicação, que neste caso são os periódicos e conferências, para uma certa área. Esta área pode ser administração, ciências e entre outros. Para este trabalho foram consideradas as áreas Ciência da Computação e Engenharia IV.

A implementação desta ontologia não apresenta relação alguma com a do Lattes. Contudo, existem conceitos que podem ser "traduzidos" entre elas, como Anais e Periódicos, possibilitando assim a construção de novas questões a partir da união de diferentes domínios. Como pode-se observar na figura **??**.

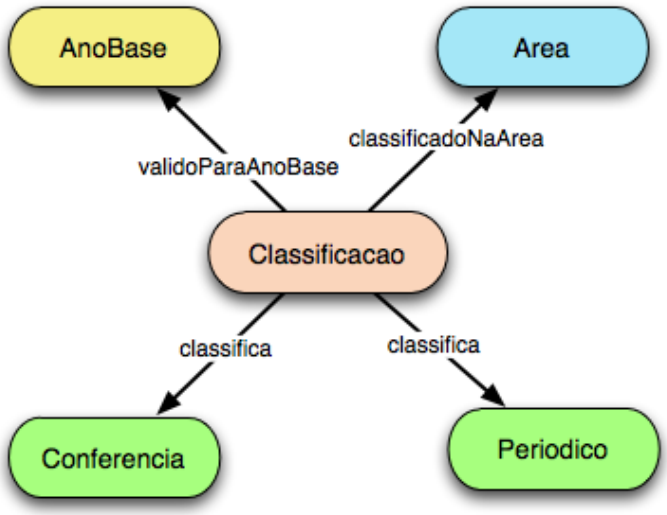

Figura 4.7: *Ontologia Qualis: Conceitos envolvidos com Classificação.*

Os conceitos na ontologia do Qualis são poucos. Temos um elemento central que corresponde a Classificação, neste caso aplicada a Conferência e Periódico. Cada classificação é válida segundo o Ano Base que foi atribuído.

4.4 Desenvolvimento das Consultas

As consultas correspondem às queries em SPARQL para obtenção dos resultados desejados de acordo com as questões de competência levantadas na especificação. Como as questões serviram de base para a construção da ontologia, as relações necessárias para obtenção da consulta foram diretas. Por exemplo, seja a consulta para uma questão do tipo:

> Quais são os professores do departamento de <nome do departamento>?

A query correspondente para esta consulta é dada por:

```
PREFIX lattes: <http://www.semanticlattes.com.br/curriculo#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
SELECT *
WHERE {
?pessoa rdf:type lattes:Professor;
    lattes:nome ?pessoa_nome;
    lattes:publicou ?publicacao;
    lattes:trabalhaNoDepartamento ?departamento.
?departamento lattes:nome ?departamento_nome.
?publicacao rdf:type lattes:Publicacao;
    lattes:titulo ?publicacao_titulo;
    lattes:ano ?publicacao_ano.
OPTIONAL { ?publicacao lattes:idioma ?publicacao_idioma }
FILTER regex(?departamento_nome, <nome_do_departamento>, 'i')
}
ORDER BY DESC(?publicacao_ano)
```

Pode-se facilmente observar os relacionamentos envolvidos nesta consulta. Deseja-se as publicações de professores, e logo pega-se todas as pessoas que são professores na ontologia (*?pessoa rdf:type lattes:Professor*) . Em seguida obtem-se aquelas que trabalham no departamento passado na questão ([?pessoa][1] *lattes:trabalhaNoDepartamento ?departamento*). E por último, todas as publicações que cada pessoa publicou são selecionadas ([?pessoa] *lattes:publicou ?publicacao*). As demais informações são os dados que serão exibidos ao usuário, como título, ano de publicação, idioma e entre outros.

Outro exemplo de query SPARQL seria o uso das duas ontologias desenvolvidas, Lattes e Qualis. Veja abaixo um exemplo:

[1] Elemento implícito. O ';' é usado para encadear declarações.

Quais são os artigos Qualis <estrato> já publicados entre <ano1> e <ano2>?

```
PREFIX lattes: <http://www.semanticlattes.com.br/curriculo#>
PREFIX qualis: <http://qualis.capes.gov.br/qualis-capes.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
SELECT *
WHERE {
  ?periodico rdf:type lattes:Periodico;
    lattes:titulo ?periodico_titulo;
    lattes:ISSN ?periodico_ISSN;
    lattes:possuiEdicaoPeriodico ?periodico_edicao.
  ?artigo rdf:type lattes:Artigo;
    lattes:titulo ?artigo_titulo;
    lattes:ehPublicadoEmEdicaoPeriodico ?periodico_edicao;
    lattes:temAutor ?pessoa;
    lattes:ano ?artigo_ano.
  ?pessoa lattes:nome ?pessoa_nome.
  _:qualis_area rdf:type qualis:Area;
    qualis:nome_area ?area_nome.
  _:qualis_periodico rdf:type qualis:Periodico;
    qualis:ISSN ?periodico_ISSN.
  ?c rdf:type qualis:Classificacao;
    qualis:classificadoNaArea _:qualis_area;
    qualis:estrato ?periodico_estrato;
    qualis:classifica _:qualis_periodico.
  OPTIONAL {
    ?pessoa lattes:trabalhaNoDepartamento ?departamento.
    ?departamento lattes:nome ?departamento_nome.
  }
  FILTER (str(?area_nome) = 'ENGENHARIAS IV')
  FILTER regex(?periodico_estrato, '<estrato>', 'i')
  FILTER (str(?artigo_ano) >= '<ano1>'
          && str(?artigo_ano) <= '<ano2>')
}
```

Podemos observar que relacionamos o ISSN dos periódicos para poder obter sua classificação do Qualis. Note que foi adicionado um novo prefixo para a ontologia do qualis.

Todas as consultas seguiram o mesmo raciocínio em suas construções.

5 IMPLEMENTAÇÃO E TESTES

5.1 Ferramentas Utilizadas

Na execução do projeto, foram necessárias ferramentas que apoiassem o ambiente de desenvolvimento necessário para o uso de tecnologias como RDF, RDFS, OWL e SPARQL (seção 2.2), motores de inferência (seção 2.3), persistência dos dados (seção 2.3.2.2) e, por último, ferramentas para desenvolvimento de aplicações Web.

Nesta seção, veremos as ferramentas utilizadas para o desenvolvimento do Semantic Lattes conforme as descrições apresentadas no capítulo 3, sobre o sistema.

5.1.1 Protégé-OWL

O Protégé[1] é uma plataforma open-source desenvolvida na Faculdade de Medicina da Universidade de Stanford, e foi concebida para auxiliar no processo de criação de modelos de domínio com ontologias. Esta plataforma provê duas maneiras de modelar ontologias: através do editor Protégé-Frames e do editor Protégé-OWL. Neste projeto utilizou-se o último, pois precisamos da construção de ontologias para os domínios tratados. O Protégé permite a exportação em vários formatos incluindo RDF, RDFS, OWL, N3 e entre outros.

Com esta ferramenta é possível modelar desde a mais simples ontologia, como uma taxonomia e classificação, até descrições complexas do OWL-Full.

O Protégé está na versão 4 e possui características como:

- Compatibilidade com OWL 2.0;
- É desenvolvida na plataforma Java;
- Inclui o FaCT++ (motor de inferência) nativo no próprio editor;

[1] http://protege.stanford.edu/.

- Permite a manipulação de várias ontologias no mesmo editor;
- Suporte a plugins de terceiros.

Em particular, o suporte a plugins permite a integração do Protégé com o Pellet, que é o motor de inferência adotado neste projeto, veja seção 5.1.2. Esta facilidade, permite que a validação dos conceitos da ontologia no próprio editor, facilitando a manutenção e correção de possíveis inconsistências.

A interface do Protégé-OWL permite trabalhar separadamente com as definições do OWL, com abas exclusivas para a manipulação das classes, propriedades do objeto, propriedades de dado e instâncias. Na figura 5.1 podemos visualizar a interface gráfica do Protégé e as abas que separam cada elemento da ontologia.

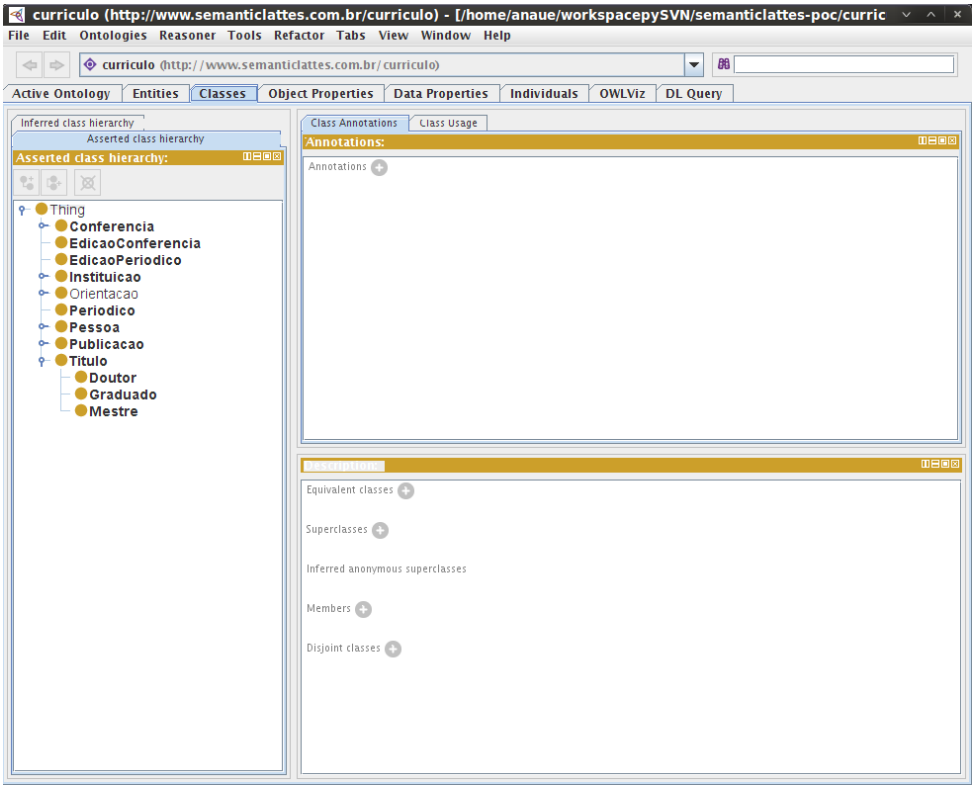

Figura 5.1: *Interface gráfica do Protégé.*

5.1.2 Pellet

O Pellet[2] é um motor de inferência desenvolvido em Java publicado sob uma licença *open-source* e uma proprietária. Provê suporte ao OWL 2, da W3C, e é mantido e desenvolvido pela *Clark&Parsia*. É a principal escolha para sistemas que necessitem de inferência em OWL

[2]http://clarkparsia.com/pellet.

DL. Um motor de inferência como o Pellet, é o componente central em aplicações baseadas em gerenciamento de dados por ontologias, apresentados na seção 2.3.

A ferramenta Pellet é bem popular em aplicações semânticas, pois é facilmente integrável com a ferramenta Jena e o editor Protégé.

5.1.3 Jena

O Jena[3] é um framework em Java e open-source[4] sob desenvolvimento em conjunto com o *HP Labs Semantic Web Programme*, da Hewlett-Packard. É uma ferramenta voltada para o desenvolvimento de aplicações de Web Semântica, provendo um ambiente de desenvolvimento para RDF, RDFS, OWL e SPARQL, incluindo um motor de inferência baseado em regras (seção 2.3).

A ferramenta Jena inclui:

- Uma API[5] para RDF;
- Leitura e escrita em RDF para RDF/XML, N3 and N-Triples;
- Uma API para OWL;
- Armazenamento em memória e disco;
- Motor de consulta em SPARQL.

O Jena inclui diferentes outros projetos internos com o ARQ, TDB e SDB, e motores de inferência próprios e o Joseki. O ARQ é uma implementação para a linguagem de consulta SPARQL. O TDB e SDB são sistemas para persistência de banco de dados, onde o primeiro é uma implementação própria e otimizada de um banco de dados, e o segundo é uma interface para utilização de bancos de dados relacionais. O Jena inclui sua própria implementação de motores de inferência, mas possibilita, por meio de interfaces, agregar diferentes implementações de terceiros, como o Pellet, citado anteriormente. O Joseki é uma implementação do protocolo SPARQL em HTTP para a realização de consultas à base de conhecimento.

Em resumo, o Jena atua como integrador das diferentes ferramentas envolvidas no desenvolvimento de aplicações de Web Semântica.

[3] http://jena.sourceforge.net/.
[4] http://www.openjena.org.
[5] Application Programming Interface.

5.1.3.1 Conceitos de Programação do Jena

O Jena é um abstração dos conceitos da Web Semântica em classes Java. Os artefatos da Web Semântica são representações de dados, enquanto que para o Jena são objetos contendo dados e métodos de acesso à eles. Na tabela 5.1 pode-se observar uma analogia entre os elementos da Web Semântica em comparação com as do Jena obtidos em (HEBELER et al., 2009).

Artefato	Web Semântica	Classe Java no Jena
Subjeito, Predicado e Objeto	URI	Resource, Property
Declaração	Statement	Statement
Dados	Ontologia e instâncias de dados	Graph e Model
Consulta e resultados	SPARQL e dados da Web Semântica	Query e ResultSet
Motor de Inferência	Motor de Inferência	Reasoner
Regras	SWRL[6]	Reasoner
Notificações de Eventos	Não Aplicado	ObjectListener

Tabela 5.1: *Comparação da Web Semântica e Jena*

Populando o Model com Dados da Web Semântica

Como apresentado na tabela 5.1, um Model corresponde a abstração dos dados da Web Semântica em uma classe Java. Note que é possível carregar uma descrição através de um arquivo local ou de um recurso externo, como uma ontologia de um endereço na Web. Logo abaixo, pode-se observar um exemplo de como é carregada em memória a descrição da ontologia e suas instâncias.

```
Model ontologia = ModelFactory.createDefaultModel();
// carregamento em arquivo
FileManager.get().readModel(ontologia, "ontologia.owl");
...
// carregamento em URL
FileManager.get().readModel(ontologia, URL);
```

Definindo Relações de Equivalência

Com a ontologia carregada podemos adicionar mais informações à ela dinamicamente, como por exemplo, explicitar conceitos que são equivalentes. Como mencionamos neste trabalho, estamos lidando com o domínio do Lattes e do Qualis e ambos apresentam o conceito de Periodico e possuem uma propriedade de dados ISSN. Podemos considerá-los como equivalentes no nosso domínio.

```
Resource resource = schema.getResource(QUALIS + "Periodico");
Resource obj = schema.getResource(LATTES + "Periodico");
schema.add(resource, OWL.equivalentClass, obj);

resource = schema.getResource(QUALIS + "ISSN");
obj = schema.getResource(LATTES + "ISSN");
schema.add(resource, OWL.equivalentProperty, obj);

resource = schema.getResource(QUALIS + "titulo_periodico");
obj = schema.getResource(LATTES + "titulo");
schema.add(resource, RDFS.subPropertyOf, obj);

resource = schema.createResource(LATTES + "ehClassificadoPor");
obj = schema.getResource(QUALIS + "ehClassificadoPor");
schema.add(resource, OWL.equivalentProperty, obj);
```

Veja que estamos considerando apenas que os conceitos são equivalentes, e não quais instâncias são as mesmas. Para esta tarefa teríamos de explicitar instância por instância utilizando a propriedade `owl:sameAs` que permite considerar que dois recursos com diferentes URIs correspondem a mesma entidade. Este tipo de inferência não pode ser feita apenas com ontologias, para isso utilizamos regras que definem certas relações.

Definindo Regras na Ontologia

Regras são formas de representar conhecimento que vão além OWL ou que são mais fáceis de se compreender. Na Web Semântica, elas tendem a ser declarações condicionais. Abaixo iremos determinar como os Periódicos do Lattes e Qualis se tornam a mesma entidade, se ambas tiverem o mesmo ISSN.

Descrevendo a regra num arquivo `lattes.rules`, temos algo como:

```
@prefix lattes: http://www.semanticlattes.com.br/curriculo#
@prefix qualis: http://qualis.capes.gov.br/qualis-capes.owl#
[rule1:
  (?p1 lattes:ISSN ?issn)
  (?p2 qualis:ISSN ?issn)
  ->
  (?p1 owl:sameAs ?p2)]
```

É possível definir inúmeras regras num mesmo arquivo. Agora é possível aplicar estas regras sobre a base de conhecimento.

```
Parser parser = Rule.rulesParserFromReader(
                  new BufferedReader(
```

```
                    new FileReader("lattes.rules")));
Reasoner ruleReasoner = new GenericRuleReasoner(Rule.parseRules(parser));
ruleReasoner.bindSchema(schema);
InfModel ruled = ModelFactory.createInfModel(ruleReasoner, model);
```

No código acima, é criado um parser para o documento contendo as regras. Em seguida, cria-se um `Reasoner` para regras, neste caso do tipo `GenericRuleReasoner` (do Jena). Assim, obtemos uma model do tipo InfModel contendo as inferências obtidas das regras. Portanto, neste exemplo temos que todas as instâncias com a propriedade ISSN e de mesmo valor, são consideradas como a mesma entidade.

Realizando Inferência Sobre os Dados

Com a representação dos dados, agora é possível aplicar o motor de inferência sobre este conjunto e obter novas triplas concluídas a partir das descrições da ontologia.

```
Reasoner reasoner = PelletReasonerFactory.theInstance().create();
InfModel inferencia = ModelFactory.createInfModel(reasoner, ontologia);
```

Inicialmente cria-se uma classe Reasoner que corresponde a uma interface e então cria-se um instância do motor de inferência Pellet. Ao aplicar o `reasoner` na `ontologia`, gera-se um novo objeto `inferencia` que corresponde a representação em memória das conclusões lógicas geradas pelo Pellet. Note que os objetos `ontologia` e `reasoner` são disjuntos, o objeto inferido contém apenas os resultados da inferência e o objeto da ontologia apenas o conjunto de dados originais.

É recomendável manter separado o Model das instâncias com o da ontologia. Assim, é possível remover facilmente os instância do Model inferencia, para evitar a criação de um novo `InfModel`.

Assim para adicionar ou remover as instâncias:

```
inferencia.add(instancias);    // adiciona
inferencia.remove(instancias); // remove
```

No caso se queira unificar as informações, basta adicionar o resultado da inferência ao objeto `ontologia`.

```
instancias.read(inferencia);
```

Consultando os Dados da Web Semântica

A busca dos dados é realizada através da linguagem de consulta SPARQL. Para se obter o resultado, cria-se um objeto Query com a representação da consulta SPARQL e executa-se sobre os dados semânticos criados anteriormente.

```
Query query = QueryFactory.create(sparql);
QueryExecution qexec = SparqlDLExecutionFactory.create(query, inferencia);
ResultSet resultado = qexec.execSelect();
ResultSetFormatter.out(resultado);
qexec.close();
```

Os resultados são representados pelo objeto `resultado` que é uma instância da classe ResultSet. Assim, basta iterar sobre os resultados e apresentá-los ao usuário da forma desejada.

5.1.4 TDB

Os arquivos de instâncias em OWL gerados a partir dos currículo no formato XML da base Lattes, na seção 5.2, necessitam ser carregados em memória sempre que o motor de busca semântico for utilizado, como visto na seção 5.1.3.1. Entretanto, com o crescimento dos conjuntos de dados de novos currículos, torna-se impossível carregar todas as informações em memória, devido às limitações de espaço de memória dos computadores atuais.

Na aplicação de inferência sobre um conjunto de dados, novos grafos são gerados e, como consequência, existe um acréscimo no volume de informações a serem mantidas em memória. Diante desta limitação, é necessário que exista algum mecanismo de armazenamento, além da memória, para que se torne possível o tratamento de grandes volumes de dados. Uma primeira opção de armazenamento é o uso do sistema de arquivos do sistema operacional que é facilmente extendido, porém mais lento. Esta limitação de desempenho, pode ser tratada utilizando banco de dados que possuem mecanismos para otimizar estas tarefas.

Contudo, o uso de banco de dados relacional, que é a forma de armazenamento de dados tradicional, não é própria para sistemas baseados em ontologias, como apresentado na seção 2.3.2, pois os dados da Web Semântica são constituintes de grafos. Existem diversas áreas de pesquisas e empresas desenvolvendo bancos de dados orientados a grafos, como por exemplo Neo4j[7] e o AllegroGraph[8].

Em função da utilização do Jena nesse projeto, optou-se por utilizar o projeto TDB que é uma implementação interna no projeto da ferramenta Jena. O TDB consitui-se em um sistema de armazenamento de dados especializado para os tipos de dados tratados na Web

[7] http://neo4j.org/.
[8] http://www.franz.com/agraph/allegrograph/.

Semântica, incluindo mecanismos de banco de dados como índices e *cache* de dados em disco, para melhorar o desempenho nas buscas. Existe um outro projeto chamada SDB, também do Jena, que permite o uso de banco de dados relacional (MySQL, SQL Server, PostgreSQL e entre outros). Entretanto, como mencionado anteriormente, banco de dados relacionais não são especializados para este tipo de estrutura de armazenamento, portanto são recomendados apenas em situações em que a utilização de banco de dados é, de certa forma, imposta.

5.1.4.1 Armazenamento de Dados em TDB com Jena

O armazenamento de dados se faz da mesma forma que o processo de carregamento dos dados da Web Semântica em mémoria. Entretanto, no lugar de uma representação em memória, utiliza-se uma outra implementação da classe Java Model do Jena, no caso o TDB. A diferença esta no uso dos métodos `commit` e `close`, que são necessários para que a persistência em disco seja feita.

```
Model modelDB = TDBFactory.assembleModel(assemblerFile);
InputStream in = FileManager.get().open("ontologia.owl");
modelDB.read(in);
in.close();
modelDB.commit();
modelDB.close();
```

É possível persistir dados de inferência no TDB, contudo isto não é muito recomendado pois as informações tendem a mudar e isso deve refletir nos dados armazenados no TDB. Outra consequência é que o volume de formações gerados e processados são extremamente grandes dependendo do volume de dados existentes. O que é possível, no entanto, é o armazenamento de um subconjunto destes dados gerados.

5.1.4.2 Atualização de Dados no TDB

A atualização dos dados do TDB, é feito de forma transparente, bastando carregar os novos dados atualizados e submeter as mudanças. Entretanto, segundo recomendação dos desenvolvedores do projeto Jena, obtidas no grupo de discussão do projeto, é preferível que, em certos casos, essa atualização seja precedida pela remoção completa dos dados e posterior persistência dos dados atualizados, a fim de aumentar a confiabilidade dos dados no sistema. Por exemplo, poderia haver a ocorrência de inconscistência nos dados devido a alterações na ontologia.

Em casos em que existem muitos arquivos a serem importados, é preferível que as informações do TDB sejam exportadas para um único documento OWL, contendo a completa representação dos dados, reduzindo o trabalho de carga na base de conhecimento.

5.1.4.3 Configuração do TDB

A configuração do TDB é feita por um arquivo externo contendo o local em que o TDB é armazenado. A configuração mais simples está apresentada logo abaixo:

```
@prefix tdb:     <http://jena.hpl.hp.com/2008/tdb#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix ja:      <http://jena.hpl.hp.com/2005/11/Assembler#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .

[] ja:loadClass "com.hp.hpl.jena.tdb.TDB" .
tdb:DatasetTDB  rdfs:subClassOf  ja:RDFDataset .
tdb:GraphTDB    rdfs:subClassOf  ja:Model .

<#dataset> rdf:type      ja:RDFDataset ;
     ja:defaultGraph <#graph> ;
.

<#graph> rdf:type tdb:GraphTDB ;
    tdb:location "DB" ;
    .
```

Basicamente, o documento determina o local (`tdb:location`) do banco de dados, que nesse caso é determinada como a pasta DB.

5.1.5 Ruby, JRuby e Sinatra

Nos últimos anos o Ruby tem ganhado bastante visibilidade no desenvolvimento de aplicações Web, principalmente como consequência do framework de desenvolvimento Rails, comumente chamado de Ruby on Rails. O Ruby é uma linguagem interpretada, semelhante ao Python, mas com um alto poder de expressividade e flexibilidade, aliado a características de linguagem imperativa e funcional, incluindo metaprogramação. Esta flexibilidade, torna o desenvolvimento de soluções para problemas muito mais claras que linguagens como C# e Java. Devido a necessidade do uso de Java pelas ferramentas apresentadas nas seções anteriores, optamos por utilizar o JRuby.

O JRuby é uma implementação completamente em Java para a linguagem de programação

Ruby[9]. É um projeto com o intuito de trazer os benefícios existentes no Ruby e a robustez do Java para o desenvolvimento de aplicações.

O Ruby on Rails[10] é um framework completo baseado na arquitetura MVC (*Model-View-Controller*), possuindo inúmeras funcionalidades para uma solução completa de um sistema Web. Entretanto, para projetos menores ele acaba se tornando complexo demais. Para isto, existe um outro framework minimalista chamado Sinatra[11].

O Sinatra é uma ferramenta para desenvolvimento de aplicações Web que não necessitam de uma infraestrutura completa de uma aplicação Web, sendo excelentes para o desenvolvimento de pequenas aplicações e serviços com arquitetura RESTful. A sua notação é bastante expressiva e por este motivo foi utilizado como ferramenta para o desenvolvimento da interface deste projeto.

5.2 Geração das Instâncias

Instância é denominação atribuída ao resultado da tradução das informações existentes nos currículos do sistema Lattes para um conjunto de descrições em OWL com base na ontologia construída.

Esta etapa foi a mais trabalhosa do projeto, pois correspondeu ao mapeamento dos dados. A dificuldade está no fato de que os currículos são disconexos e extensos e, como consequência, certas informações não possuem vínculos com outros currículos. Os currículos são escritos livrementes, dentro de algumas restrições, por cada autor de seu currículo, ou seja, poderíamos ter um *Jon Doe* que corresponde ao *John Doe*, mas com erro de escrita.

Alguns conceitos, ainda, precisam ser inferidos durante a carga do currículo no sistema, pois, em alguns casos, não há uma declaração explícita sobre as classificações utilizadas na ontologia. Por exemplo, para poder definir um autor do currículo como Professor é necessário analisar os elementos do currículo que declaram que os seus vínculos profissionais não estejam marcados com uma data de finalização.

[9] http://jruby.org.
[10] http://rubyonrails.org/.
[11] http://www.sinatrarb.com/.

5.2.1 Abordagem Adotada

A abordagem consistiu apenas na importação dos currículos e conversão para instâncias OWL. As instâncias dentro do currículo foram guardadas a partir de alguma informação que pudesse identificá-la, como por exemplo, o nome de uma pessoa ou o título de um artigo.

Cada instância possui um único URI que serve como chave, para isso foram usados UUID (Universally Unique IDentifier), e para cada relacionamento que referenciava uma instância já existente um UUID era atribuído ou criado, quando necessário.

Numa importação inicial, um autor que não seja o dono do currículo não passa de uma pessoa. No caso de importar o currículo deste autor, que anteriormente era apenas uma pessoa, as informações como departamento em que trabalha e vínculo institucional era adicionadas quando possível.

Abaixo temos exemplos de instâncias de uma pessoa.

```
<Pessoa rdf:about="#6F2FDABC-E8FD-489E-B647-2A1454F4FDA5">
  <nome>Jaime Sim&#227;o Sichman</nome>
  <nomeParaCitacao>SICHMAN, J. S.</nomeParaCitacao>
</Pessoa>
```

Acima temos a versão inicial importada de uma instância, deste modo criamos um registro para o nome com a chave "6F2FDABC-E8FD-489E-B647-2A1454F4FDA5". Na carga de um outro currículo que possue mais informações sobre essa pessoa apontando para a mesma chave criada, os dados do registro anterior são atualizados como mostrado abaixo, sem a necessidade de desenvolver algum código que gerenciasse estas alterações.

```
<Pessoa rdf:about="#6F2FDABC-E8FD-489E-B647-2A1454F4FDA5">
  <rdf:type rdf:resource="&owl;Thing"/>
  <rdf:type rdf:resource="#Professor"/>
  <nome>Jaime Sim&#227;o Sichman</nome>
  <nomeParaCitacao>SICHMAN, J. S.</nomeParaCitacao>
  <temTitulo rdf:resource="#TituloGraduado"/>
  <temTitulo rdf:resource="#TituloMestre"/>
  <temTitulo rdf:resource="#TituloDoutor"/>
  <temVinculoCom rdf:resource="#9755DC67-3192-471D-96D5-FFEEFCFCFB45"/>
  <trabalhaNoDepartamento rdf:resource="#D2515900-4EE5-49CA-8C81-6F69BBDEACE8"/>
</Pessoa>
```

Essa característica se deve a proposição de Mundo Aberto adotada na descrição das ontologias da Web Semântica, conforme explicado no seção 2.1.2.3. Esse recurso extremamente prático possibilita focar na geração de arquivos OWL corretamente.

5.2.2 Implementação dos Tradutores

Para implementar os tradutores, foi utilizada a linguagem Ruby por ser bastante expressiva e versátil no desenvolvimento.

Com as informações dos currículos organizadas de forma apropriada, segundo as abordagens apresentadas acima, utilizou-se algumas bibliotecas, denominadas *Gems* no Ruby, que tornaram o desenvolvimento mais eficiente. Utilizando a *Gem builder*, foi possível construir o arquivo de instâncias das classes a partir dos elementos dos currículos da seguinte forma:

```
@e.Pessoa "rdf:about" => "##{generate_uuid}" do
  @e.rdf :type, "rdf:resource" => "&owl;Thing"
  @e.nome pessoa.name
  @e.nomeParaCitacao dados['NOME-COMPLETO']
  @e.temTitulo "rdf:resource" => "#TituloGraduado" if graduaodo?(curriculo)
  @e.temTitulo "rdf:resource" => "#TituloMestre" if mestre?(curriculo)
  @e.temTitulo "rdf:resource" => "#TituloDoutor" if doutor?(curriculo)
  instituicoes(curriculo).each do |i|
    @e.temVinculoCom "rdf:resource" => "##{generate_uuid(i)}"
  end
  departamentos(curriculo).each do |d|
    @e.trabalhaNoDepartamento "rdf:resource" => "##{generate_uuid(d)}"
  end
end
```

Pode-se ver no código acima que a construção XML é bem expressiva, diferentemente das alternativas equivalentes nas linguagens Java e C#.

5.3 Interface do Usuário

A interface do sistema utiliza um sistema Web para consulta e visualização dos resultados. A aplicação que realiza as inferências e consulta na base de conhecimento é implementada em Java, devido às restrições das ferramentas disponíveis para Web Semântica. Entretanto, em função da adoção da linguagem Ruby no desenvolvimento dos módulos de apoio a interface Web, foi utilizada o framework Sinatra, para elaboração da Interface do usuário.

A interface compreende duas funcionalidades: carga de novos currículos e consulta aos currículos.

5.3.1 Carga de Currículos

A carga de currículos permite importar um currículo extraído do Lattes em XML para o sistema, incorporando novas instâncias na base de conhecimento.

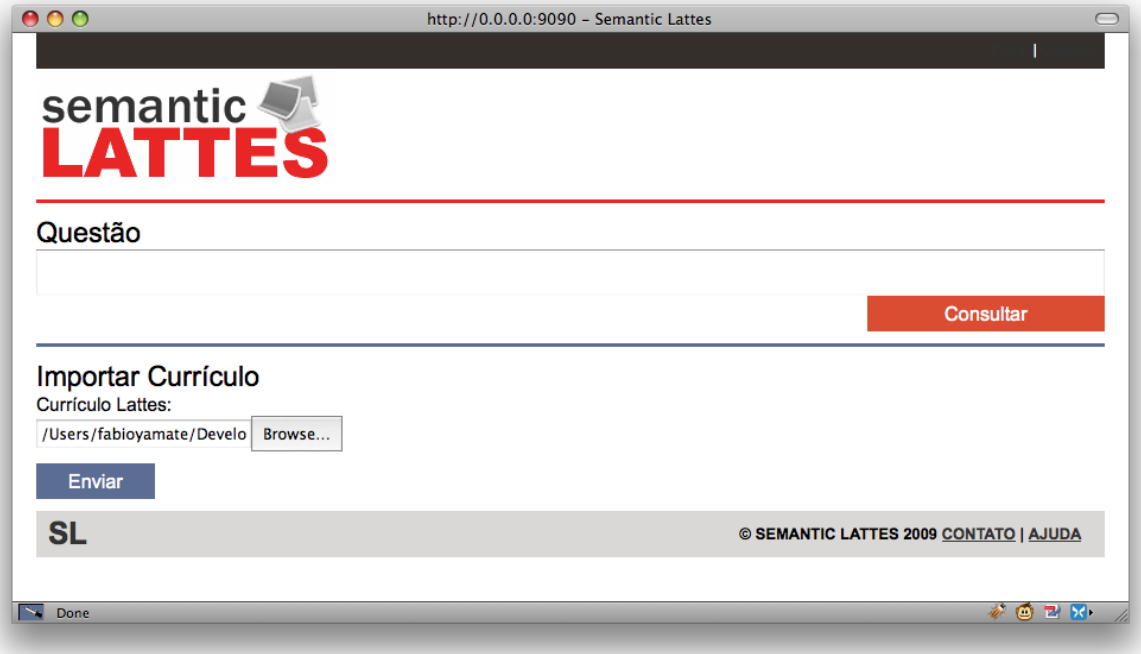

Figura 5.2: *Interface para carga de currículos no sistema.*

Ao inserir o currículo no sistema, os dados lidos são aplicados na atualização ou criação das instâncias, que depois são convertides para OWL e então agregada à base de conhecimento do sistema.

5.3.2 Consulta aos Currículos

As consultas são feitas pela entrada da questão de interesse no campo de busca da interface, como ilustrada na figura 5.3. Como apresentado na seção 3.2, as questões são perguntas e não palavra-chaves, por exemplo:

> Quais são os livros publicados pelo professor `<nome do professor>`?

Este tipo de abordagem torna mais próximo o sentido semântico da busca em ontologias, que visa dar sentido aos elementos do domínio. Isto torna mais preciso os resultados daquilo que o usuário tem interesse, ao invés de palavra-chaves como: "livro" "professor" "<nome>".

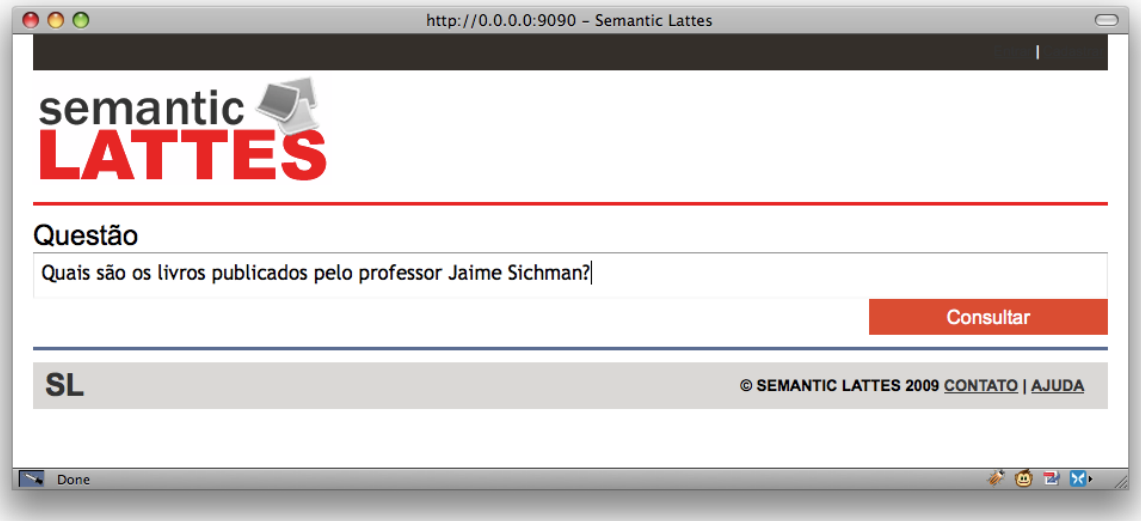

Figura 5.3: *Interface de consulta a base de conhecimento semântico do Lattes.*

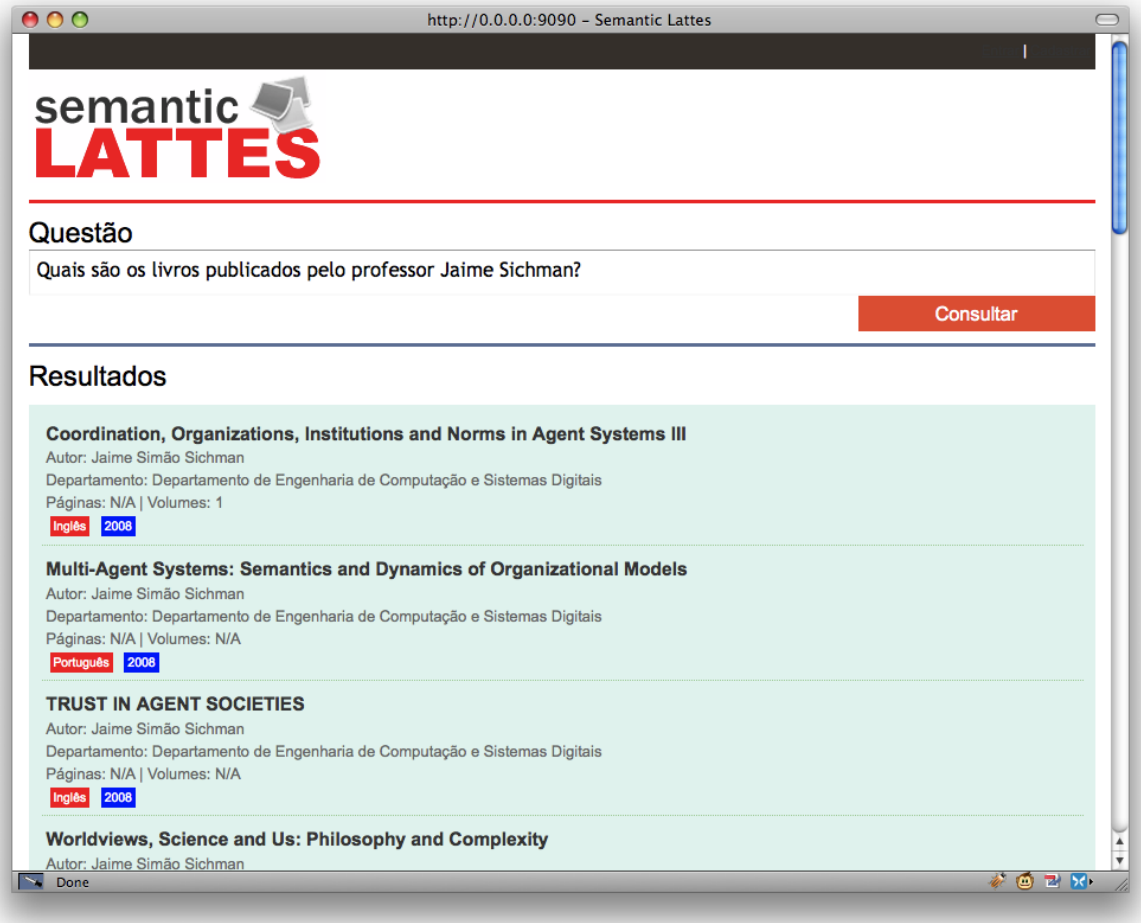

Figura 5.4: *Interface com resultados obtidos da consulta.*

5.4 Testes

Durante o desenvolvimento das ontologias foram utilizadas ferramentas para a construção de testes unitários que validassem os requisitos da ontologia. O processo de construção da ontologia depende muito de suas aplicações e, portanto, são constantemente alteradas. A não familiarização com o domínio tratado exigiu a necessidade constante de mudança na ontologia.

Conforme a ontologia se desenvolve, começa a crescer em tamanho e complexidade. Sem a existência de um processo de desenvolvimento, é possível corromper facilmente uma situação anterior válida, como ocorre em todo projeto. Continuar validando a ontologia manualmente é um processo custoso e sujeito às falhas do desenvolvedor.

Para contornar este tipo de problema foram utilizados o JUnit juntamente com o Jena para a validação da ontologia. Para cada relacionamento construído foi definido um cenário que validasse a construção, o que permitiu a configuração de um ambiente automatizado para testes.

Como não existe um padrão ou ferramenta para testes de ontologias, tentou-se adaptar modelos de testes em software para a construção de ontologias.

5.4.1 Testes da Ontologia

No desenvolvimento dos testes, foi criada uma classe que realiza a leitura da ontologia em OWL e obtém as entidades e propriedades definidas nela. Caso uma entidade fosse renomeada, isso quebraria o leitura da ontologia e causaria uma inconsistência nos testes, o que seria esperado.

O Jena provê uma API para criação em memória de instâncias. Com isso, é possível criar um cenário contendo as entidades para testes e realização de inferências e consultas sobre este cenário, validando o resultado esperado. Abaixo segue um exemplo de teste realizado:

```
public class PublicacoesDeDepartamentoTest extends CurriculoTestCase {
  Resource umaPessoa, umDepartamento, umaPublicacao;

  @Override
  protected void setUp() throws Exception {
    super.setUp();
    umaPessoa = createResource(Pessoa, "umaPessoa");
    umDepartamento = createResource(Departamento, "umDepartamento");
    umaPublicacao = createResource(Publicacao, "umaPublicacao");
  }
```

```
  public void test_umaPessoaPublicouUmaPublicacao() {
    umaPessoa.addProperty(publicou, umaPublicacao);

    String query =
      "SELECT * " +
      "WHERE {" +
      "?pub rdf:type lattes:Publicacao;" +
      "  lattes:publicadoPor ?pes." +
      "}";

    ResultSet rs = executeQuery(mountQuery(query), WITH_VERBOSE);
    QuerySolution qs = rs.next();
    Resource pes = qs.getResource("pes");
    Resource pub = qs.getResource("pub");

    assertEquals(umaPessoa, pes);
    assertEquals(umaPublicacao, pub);

    assertEquals(rs.hasNext(), false);
  }
}
```

Durante o processo de desenvolvimento, esta prática foi muito útil, considerando que o ambiente de execução do processo era compartilhado, e a alteração de um desenvolvedor poderia corromper o que outro havia desenvolvido. Utilizando esta abordagem, foi possível minimizar a ocorrência de erros e ter um processo de desenvolvimento menos caótico.

5.4.2 Testes Integrados

Os testes integrados correspondem aos testes utilizando todos os componentes desenvolvidos para o sistema. Neste caso, seriam:

1. Importação do currículo;
2. Entrada da consulta/questão;
3. Reconhecedor da questão;
4. Busca na Base de Conhecimento;
5. Formatação do resultado.

5.4.2.1 Importação de Currículo

A importação dos currículos é feita pelo envio através da interface do sistema como apresentado na figura 5.5.

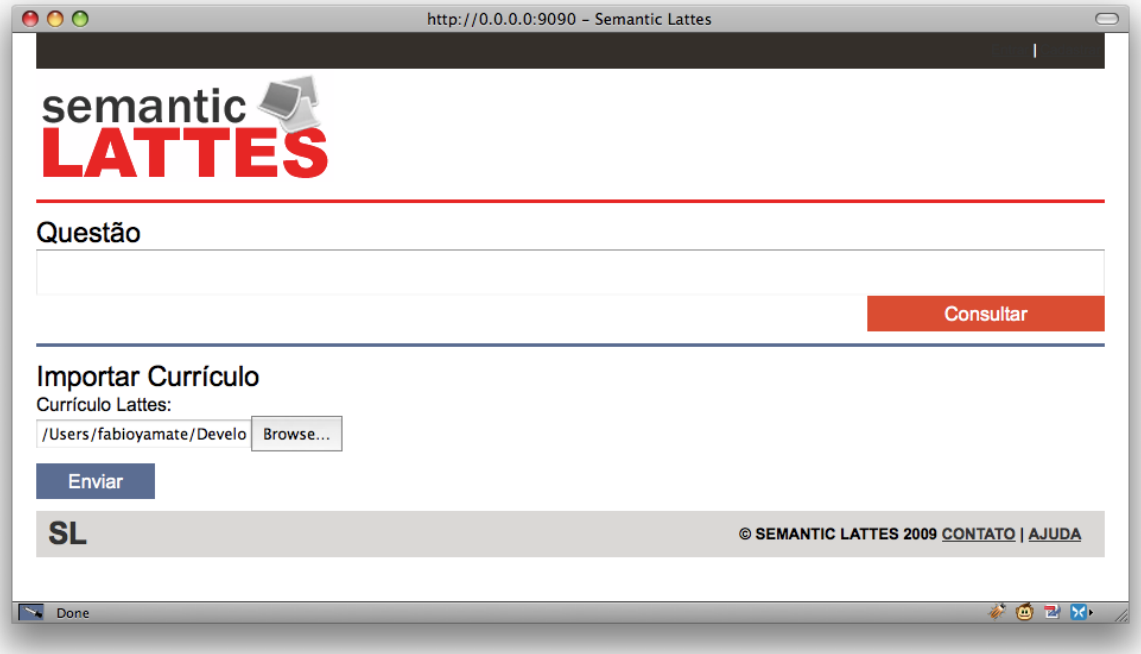

Figura 5.5: *Teste: Entrada do currículo Lattes para o sistema.*

O currículo, então, é convertido para instâncias da ontologia e carregadas na base de conhecimento.

5.4.2.2 Busca na Base de Conhecimento Semântico do Lattes

A entrada da consulta corresponde à uma pergunta em "linguagem natural" com certos argumentos pré-definidos no sistema.

Esta pergunta é então enviada à um reconhecedor que tentará extrair os respectivos argumentos, de acordo com os padrões escritos em expressões regulares. Cada questão está associada a uma query em SPARQL com os respectivas entradas para os argumentos extraídos da pergunta.

Se a consulta é encontrada, ela é executada na base de conhecimento, gerando as entradas pertinentes à consulta e então exibidas ao usuário de acordo com a formatação de cada resultado. No caso de uma questão inválida, ou seja, não reconhecida, uma mensagem é notificada ao usuário.

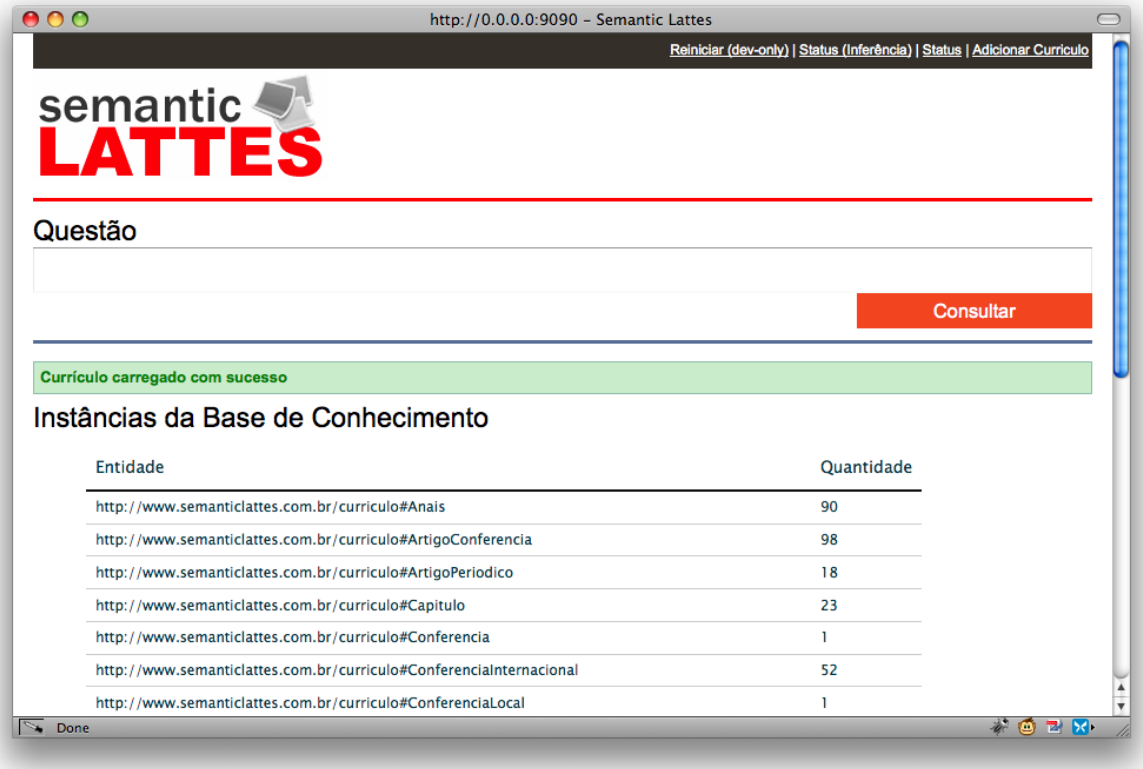

Figura 5.6: *Teste: Notificação de importação.*

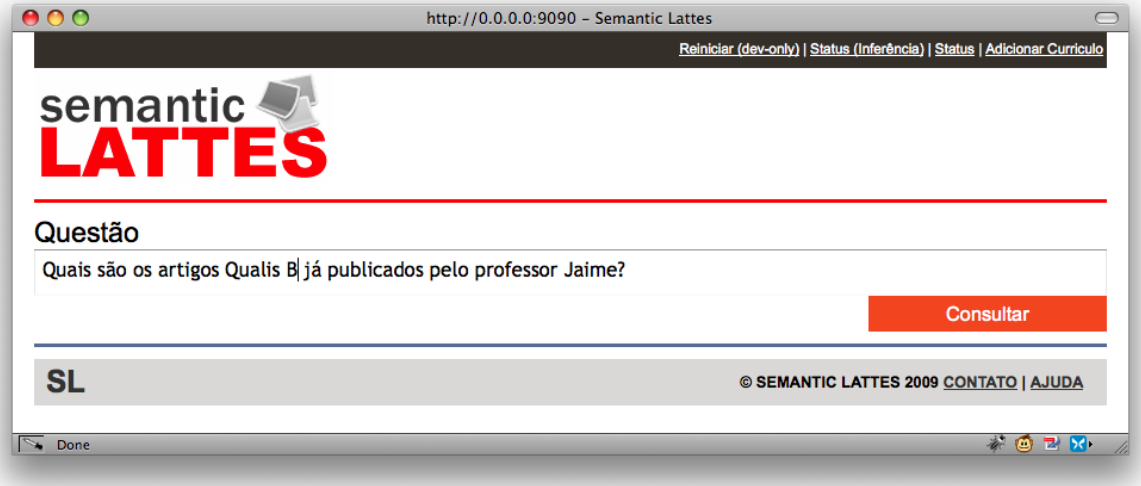

Figura 5.7: *Teste: Entrada da consulta pelo usuário.*

Os testes de integração dos diferentes módulos da aplicação não foram automatizados, sendo que, a validação é feita manualmente através da inserção das questões e verificação dos resultados com base nos dados carregados no sistema.

Figura 5.8: *Teste: Consulta reconhecida e resultado é exibido.*

6 CONCLUSÕES

O projeto obteve como resultado uma ferramenta de consulta à base Lattes com uma abordagem baseada em ontologias no lugar dos tradicionais bancos de dados relacionais.

A mudança neste paradigma permite agregar valor semântico aos dados de um domínio, no caso Lattes e Qualis. O sistema final realiza a consulta dos principais tipos de publicações de autores e atividades acadêmicas como a atuação e orientações. É possível também realizar a importação de novos currículos, incorporando novos dados ou agregando mais conhecimento aos já existentes, aumentando mais a base de conhecimento.

Com o uso de ontologias, pode-se relacionar elementos de diferentes domínios entre si, aumentando o conhecimento e possibilitando consultas mais complexas. Neste cenário, foi utilizado o domínio de classificação do Qualis para atribuir diferentes classificações às publicações da base Lattes dentre diferentes áreas, cada qual com sua relevância. Isso possibilita que num mesmo sistema utilizemos diferentes conhecimentos, ao invés de subdividir a tarefa em obter as publicações e, então, ir em outro sistema para obter seus respectivos estratos.

Os domínios não necessariamente tem de estar no mesmo sistema, pois há a possibilidade de um conteúdo ser publicado por uma entidade externa, e então ser incorporado em outro sistema, com o intuito de agregar mais conhecimento em sua base de conhecimento.

Neste projeto, diversos aspectos não foram tratados, mas que podem vir a melhorar a qualidade dos sistemas de busca. Nesse projeto, foram utilizadas ontologias para descrever o conhecimento de forma a agregar valor semântico aos dados tratados e foram utilizadas questões com "linguagem natural" com o objetivo de expor exatamente aquilo que a pessoa deseja saber. O uso de palavra-chaves independe do contexto, logo acaba gerando resultados indesejados.

Com isso, foram levantadas algumas perspectivas futuras para este projeto que podem tornar melhor a ferramenta:

Melhorias na Ontologia do Lattes e Qualis e uso de mais Domínios de Conhecimento
Esse projeto apenas validou a possibilidade de dois domínios distintos, agregarem mais conhecimento. Assim, seria possível utilizar outros domínios como geolocalização, para uso das informações geográficas como cidade, país de publicação e entre outras, encontrados no currículos; e, CAPES, para avaliação dos curos de pós-graduação.

Compilador para Linguagem Natural Simplificado Específico para o Domínio Neste caso, não há pretensão de implementar um interpretador de linguagem natural do português, mas sim gerar um compilador capaz de entender certas gramáticas da linguagem específica do domínio da base Lattes e Qualis para possibilitar uma geração automática de queries SPARQL de consulta à base de conhecimento.

Correção de Informações utilizando Inteligência Hoje os currículos apresentam diversos problemas, o principal deles é a falta de coerência em nomes, por exemplo. Diferentes autores de currículos Lattes podem escrever o nome de um artigo de formas diferentes, ou mesmo o nome de uma pessoa. Isso gera instâncias inconsistentes, pois a mesma entidade é representada em duas instâncias com nomes diferentes.

Através do uso de estatística, seria possível tentar encontrar elementos distintos com certas semelhanças e então apresentar ao usuário as possibilidades, e então esperar por uma decisão humana que confirme aquela asserção. Por exemplo, John Doe e Jon Doe, apesar de escritos de forma diferentes, poderiam corresponder à mesma pessoa. Porém, a análise não é tão simples, seria necessário utilizar um maior número de parâmetros para se determinar a relação correta entre essas instâncias geradas.

Essa implementação possibilitaria que inconsistências nos diferentes currículos pudessem ser corrigidos de forma mais ágil. Como o iPhoto'09, da Apple, e o Picasa, da Google, que hoje fornecem reconhecimento facial e exibem ao usuários os diferentes rostos encontrados e solicitam à decisão humana para melhorar seus resultados.

O grupo estima que sistemas como este tendem a ter interesse futuro, apesar da pouca visibilidade atual. Existem diversas áreas de conhecimento em que a demanda por melhor qualidade na busca por uma informação é necessária. Dentre elas, podemos citar áreas médicas e biológicas, com diversos campos de conhecimento que podem ser correlacionados entre si afim de se obter diagnósticos mais precisos. Entretanto, o processo de construção em si ainda é manual por demandar conhecimento especializado do domínio. Com o surgimento da Internet e o desenvolvimento de conhecimento compartilhado e colaborativo, existe uma grande demanda

por tornar este volume de informação compreensível para a máquina, de forma a tornar seus resultados cada vez mais precisos. Muitos têm dito que esta será a próxima geração da Web.

Anexo A – ONTOLOGIAS

As ontologias em anexo estão representadas no formato N3 por ser de mais fácil leitura.

A.1 Lattes

```
@prefix :        <http://www.semanticlattes.com.br/curriculo#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl2xml:  <http://www.w3.org/2006/12/owl2-xml#> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix curriculo:  <http://www.semanticlattes.com.br/curriculo#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

curriculo:idioma
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:paginaFinal
      a       owl:DatatypeProperty .

curriculo:TrabalhoTecnico
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:Publicacao
      a       owl:Class .

curriculo:Orientacao
      a       owl:Class .

curriculo:meioDeDivulgacao
      a       owl:DatatypeProperty .

curriculo:TituloDoutor
      a       owl:Thing , curriculo:Doutor .
```

```
curriculo:OrientacaoMestrado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Orientacao ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onClass curriculo:DissertacaoMestrado ;
                owl:onProperty curriculo:temDissertacaoMestrado ;
                owl:qualifiedCardinality
                        "1"^^xsd:nonNegativeInteger
              ] .

curriculo:temAutor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Publicacao ;
      rdfs:range curriculo:Pessoa ;
      owl:inverseOf curriculo:publicou .

curriculo:OrientacaoPosDoutorado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Orientacao .

curriculo:Anais
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao ;
      owl:equivalentClass
          [ a       owl:Restriction ;
            owl:onClass curriculo:EdicaoConferencia ;
            owl:onProperty curriculo:ehPublicadoEmEdicaoConferencia ;
            owl:qualifiedCardinality
                    "1"^^xsd:nonNegativeInteger
          ] .

curriculo:editora
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:cidade
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:instituicao
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:temVinculoCom
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Instituicao .
```

```
curriculo:EdicaoConferencia
      a       owl:Class ;
      rdfs:subClassOf owl:Thing .

curriculo:Livro
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:Titulo
      a       owl:Class ;
      rdfs:subClassOf owl:Thing .

curriculo:emAndamentoSobOrientacaoDe
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:sobOrientacaoDe ;
      owl:propertyChainAxiom
              (curriculo:sendoElaboradoPor curriculo:ehOrientadoPor) .

curriculo:ehOrientadorDeDoutorando
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:Doutorando ;
      rdfs:subPropertyOf curriculo:ehOrientadorDe .

curriculo:pais
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:TeseDoutorado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Trabalho .

curriculo:fasciculo
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

[]    a       owl:AllDisjointClasses ;
      owl:members (curriculo:ArtigoAceito
          curriculo:ArtigoConferencia
          curriculo:ArtigoPeriodico) .

curriculo:temAtividade
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa .

curriculo:statusOrientacao
      a       owl:DatatypeProperty .
```

```
curriculo:estahElaborando
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Estudante ;
      rdfs:range curriculo:Trabalho ;
      owl:inverseOf curriculo:sendoElaboradoPor .

curriculo:ano
      a       owl:DatatypeProperty ;
      rdfs:range xsd:gYear .

curriculo:Professor
      a       owl:Class ;
      rdfs:subClassOf curriculo:Pessoa ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onProperty curriculo:trabalhaNoDepartamento ;
                owl:someValuesFrom curriculo:Departamento
              ] .

curriculo:TituloGraduado
      a       owl:Thing , curriculo:Graduado .

curriculo:Doutorando
      a       owl:Class ;
      rdfs:subClassOf curriculo:Estudante ;
      owl:equivalentClass
              [ a       owl:Class ;
                owl:intersectionOf ([ a       owl:Restriction ;
                      owl:onProperty curriculo:ehOrientadoPor ;
                      owl:someValuesFrom curriculo:Pesquisador
                    ] [ a       owl:Restriction ;
                      owl:onProperty curriculo:estahElaborando ;
                      owl:someValuesFrom curriculo:TeseDoutorado
                    ])
              ] .

curriculo:localDeApresentacao
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:local .

curriculo:serie
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:ehTituloDe
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Titulo ;
```

```
      rdfs:range curriculo:Pessoa ;
      owl:inverseOf curriculo:temTitulo .

curriculo:Instituicao
      a       owl:Class .

curriculo:ehPublicadoEmEdicaoConferencia
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Anais ;
      rdfs:range curriculo:EdicaoConferencia ;
      rdfs:subPropertyOf curriculo:ehPublicadoEm ;
      owl:inverseOf curriculo:possuiAnais .

curriculo:ehEdicaoDoPeriodico
      a       owl:FunctionalProperty , owl:ObjectProperty ;
      rdfs:domain curriculo:EdicaoPeriodico ;
      rdfs:range curriculo:Periodico .

curriculo:ehCoAutorCom
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Pessoa ;
      owl:propertyChainAxiom
              (curriculo:ehPrincipalAutorDe
              curriculo:temCoAutor) ,
              (curriculo:ehCoAutorDe
              curriculo:temPrincipalAutor) .

curriculo:cidadeEditora
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:cidade .

curriculo:paisDePublicacao
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:pais .

curriculo:temOrientado
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Orientacao ;
      rdfs:range curriculo:Estudante .

curriculo:localDePublicacao
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:local .

curriculo:instituicaoPromotora
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:instituicao .
```

```
curriculo:trabalhaNoDepartamento
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Professor ;
      rdfs:range curriculo:Departamento ;
      rdfs:subPropertyOf curriculo:temVinculoCom .

curriculo:edicao
      a       owl:DatatypeProperty ;
      rdfs:range xsd:unsignedInt .

curriculo:numeroDePaginas
      a       owl:DatatypeProperty ;
      rdfs:range xsd:unsignedInt .

curriculo:finalidade
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:possuiOrientacao
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:Orientacao ;
      owl:inverseOf curriculo:temOrientador .

curriculo:disponibilidade
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:ehVinculadoCom
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Instituicao ;
      rdfs:range curriculo:Pessoa ;
      owl:propertyDisjointWith
              curriculo:temVinculoCom .

curriculo:temTrabalho
      a       owl:FunctionalProperty , owl:ObjectProperty ;
      rdfs:domain curriculo:Orientacao ;
      rdfs:range curriculo:Trabalho .

curriculo:temFuncionario
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Departamento ;
      rdfs:range curriculo:Professor ;
      rdfs:subPropertyOf curriculo:ehVinculadoCom ;
      owl:inverseOf curriculo:trabalhaNoDepartamento .
```

```
curriculo:possuiCapitulo
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Livro ;
      rdfs:range curriculo:Capitulo ;
      owl:inverseOf curriculo:ehCapituloDoLivro .

curriculo:possuiArtigoPublicadoEmConferencia
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Conferencia ;
      rdfs:range curriculo:ArtigoConferencia ;
      rdfs:subPropertyOf curriculo:possuiArtigoPublicado ;
      owl:inverseOf curriculo:ehPublicadoEmConferencia ;
      owl:propertyChainAxiom
              (curriculo:possuiEdicaoConferencia
               curriculo:possuiAnais
               curriculo:possuiArtigoPublicadoEmAnais) .

curriculo:ConferenciaInternacional
      a       owl:Class ;
      rdfs:subClassOf curriculo:Conferencia ;
      owl:disjointWith curriculo:ConferenciaNacional .

curriculo:EdicaoPeriodico
      a       owl:Class ;
      rdfs:subClassOf owl:Thing .

curriculo:anoDePublicacao
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:ano .

curriculo:RelatorioTecnico
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:OrientacaoDoutorado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Orientacao ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onClass curriculo:TeseDoutorado ;
                owl:onProperty curriculo:temTeseDoutorado ;
                owl:qualifiedCardinality
                        "1"^^xsd:nonNegativeInteger
              ] .

curriculo:dataDePublicacao
      a       owl:DatatypeProperty .
```

```
curriculo:duracao
      a       owl:DatatypeProperty ;
      rdfs:range xsd:unsignedInt .

curriculo:DissertacaoMestrado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Trabalho ;
      owl:disjointWith curriculo:TeseDoutorado .

<http://www.semanticlattes.com.br/curriculo>
      a       owl:Ontology .

curriculo:tituloIngles
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:titulo .

curriculo:Graduado
      a       owl:Class ;
      rdfs:subClassOf curriculo:Titulo .

curriculo:local
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:ehCoAutorDe
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:ehAutorDe ;
      owl:inverseOf curriculo:temCoAutor .

curriculo:ehOrientadorDeIC
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:EstudanteIC ;
      rdfs:subPropertyOf curriculo:ehOrientadorDe .

curriculo:Texto
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:ehAutorDe
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Publicacao ;
      owl:equivalentProperty
              curriculo:publicou ;
      owl:inverseOf curriculo:publicadoPor , curriculo:temAutor .

curriculo:paginaInicial
```

```
      a       owl:DatatypeProperty .

curriculo:temPrincipalAutor
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:temAutor ;
      owl:inverseOf curriculo:ehPrincipalAutorDe .

curriculo:Doutor
      a       owl:Class ;
      rdfs:subClassOf curriculo:Titulo .

curriculo:ehOrientadorDe
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:Estudante .

[]    a       owl:AllDisjointClasses ;
      owl:members (curriculo:Anais
                   curriculo:Artigo
                   curriculo:Capitulo
                   curriculo:Livro
                   curriculo:OutrasPublicacoes
                   curriculo:Patente
                   curriculo:Texto) .

curriculo:publicadoPor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Publicacao ;
      rdfs:range curriculo:Pessoa ;
      owl:equivalentProperty
              curriculo:temAutor .

curriculo:OutrasPublicacoes
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:ISSN
      a       owl:FunctionalProperty , owl:DatatypeProperty ;
      rdfs:range xsd:string .

[]    a       owl:AllDisjointClasses ;
      owl:members (curriculo:Conferencia
                   curriculo:EdicaoConferencia
                   curriculo:EdicaoPeriodico
                   curriculo:Instituicao
                   curriculo:Periodico
                   curriculo:Pessoa
                   curriculo:Publicacao
```

```
                curriculo:Titulo) .

curriculo:ConferenciaNacional
      a       owl:Class ;
      rdfs:subClassOf curriculo:Conferencia .

[]    a       owl:AllDisjointProperties ;
      owl:members (curriculo:ehOrientadorDeDoutorando
                curriculo:ehOrientadorDeIC
                curriculo:ehOrientadorDeMestrando) .

curriculo:Trabalho
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:TextoRevista
      a       owl:Class ;
      rdfs:subClassOf curriculo:Texto .

curriculo:ArtigoPeriodico
      a       owl:Class ;
      rdfs:subClassOf curriculo:Artigo ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onProperty curriculo:ehPublicadoEmEdicaoPeriodico ;
                owl:someValuesFrom curriculo:EdicaoPeriodico
              ] .

curriculo:sobOrientacaoDe
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Publicacao .

curriculo:CursoMinistrado
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:ehCapituloDoLivro
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Capitulo ;
      rdfs:range curriculo:Livro .

curriculo:Conferencia
      a       owl:Class .

curriculo:volume
      a       owl:DatatypeProperty .

curriculo:OrganizacaoEvento
```

```
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:numeroDeVolumes
      a       owl:DatatypeProperty ;
      rdfs:range xsd:unsignedInt .

curriculo:MaterialDidatico
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:ehPublicadoEm
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Publicacao .

curriculo:foiOrientadoPor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Estudante ;
      rdfs:range curriculo:Pesquisador .

curriculo:ehPublicadoEmAnais
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:ArtigoConferencia ;
      rdfs:range curriculo:Anais ;
      rdfs:subPropertyOf curriculo:ehPublicadoEm .

curriculo:TituloMestre
      a       owl:Thing , curriculo:Mestre .

[]    a       owl:AllDisjointClasses ;
      owl:members (curriculo:Doutorando
                   curriculo:EstudanteIC
                   curriculo:Mestrando) .

curriculo:Software
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:possuiArtigoPublicado
      a       owl:ObjectProperty ;
      rdfs:domain
              [ a       owl:Class ;
                owl:unionOf (curriculo:Anais
                    curriculo:Conferencia
                    curriculo:EdicaoPeriodico)
              ] ;
      rdfs:range curriculo:Artigo .
```

```
curriculo:periodoDePublicacao
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string ;
      rdfs:subPropertyOf curriculo:dataDePublicacao .

curriculo:ConferenciaLocal
      a       owl:Class ;
      rdfs:subClassOf curriculo:ConferenciaRegional .

curriculo:ehPublicadoEmConferencia
      a       owl:ObjectProperty ;
      rdfs:domain
              [ a       owl:Class ;
                owl:unionOf (curriculo:Anais
                             curriculo:ArtigoConferencia)
              ] ;
      rdfs:range curriculo:Conferencia ;
      rdfs:subPropertyOf curriculo:ehPublicadoEm ;
      owl:propertyChainAxiom
              (curriculo:ehPublicadoEmAnais
               curriculo:ehPublicadoEmEdicaoConferencia
               curriculo:ehEdicaoDaConferencia) .

curriculo:possuiArtigoPublicadoEmAnais
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Anais ;
      rdfs:range curriculo:ArtigoConferencia ;
      rdfs:subPropertyOf curriculo:possuiArtigoPublicado ;
      owl:inverseOf curriculo:ehPublicadoEmAnais .

curriculo:Artigo
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:ArtigoConferencia
      a       owl:Class ;
      rdfs:subClassOf curriculo:Artigo ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onProperty curriculo:ehPublicadoEmAnais ;
                owl:someValuesFrom curriculo:Anais
              ] .

curriculo:instituicaoFinanciadora
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:instituicao .

curriculo:nome
```

```
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:Patente
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:temOrientador
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Orientacao ;
      rdfs:range curriculo:Pesquisador .

curriculo:ehPublicadoEmEdicaoPeriodico
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:ArtigoPeriodico ;
      rdfs:range curriculo:EdicaoPeriodico ;
      rdfs:subPropertyOf curriculo:ehPublicadoEm .

curriculo:natureza
      a       owl:DatatypeProperty .

curriculo:ambiente
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:publicou
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Publicacao ;
      owl:inverseOf curriculo:publicadoPor .

curriculo:cidadeDoTrabalho
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:cidade .

curriculo:Pessoa
      a       owl:Class .

curriculo:ehPrincipalAutorDe
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:ehAutorDe .

curriculo:Capitulo
      a       owl:Class ;
      rdfs:subClassOf curriculo:Publicacao .

curriculo:titulo
      a       owl:DatatypeProperty ;
```

```
      rdfs:range xsd:string .

curriculo:Mestrando
      a       owl:Class ;
      rdfs:subClassOf curriculo:Estudante ;
      owl:equivalentClass
              [ a       owl:Class ;
                owl:intersectionOf ([ a       owl:Restriction ;
                    owl:onProperty curriculo:ehOrientadoPor ;
                    owl:someValuesFrom curriculo:Pesquisador
                  ] [ a       owl:Restriction ;
                    owl:onProperty curriculo:estahElaborando ;
                    owl:someValuesFrom curriculo:DissertacaoMestrado
                  ])
              ] .

curriculo:ehOrientadoPor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Estudante ;
      rdfs:range curriculo:Pesquisador ;
      owl:inverseOf curriculo:ehOrientadorDe .

curriculo:temTeseDoutorado
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:OrientacaoDoutorado ;
      rdfs:range curriculo:TeseDoutorado ;
      rdfs:subPropertyOf curriculo:temTrabalho .

curriculo:possuiArtigoPublicadoEmEdicaoPeriodico
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:EdicaoPeriodico ;
      rdfs:range curriculo:ArtigoPeriodico ;
      rdfs:subPropertyOf curriculo:possuiArtigoPublicado ;
      owl:inverseOf curriculo:ehPublicadoEmEdicaoPeriodico .

curriculo:temCoAutor
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:temAutor ;
      owl:propertyDisjointWith
              curriculo:temPrincipalAutor .

curriculo:ISBN
      a       owl:FunctionalProperty , owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:concluidoSobOrientacaoDe
      a       owl:ObjectProperty ;
      rdfs:subPropertyOf curriculo:sobOrientacaoDe ;
```

```
      owl:propertyChainAxiom
              (curriculo:publicadoPor curriculo:foiOrientadoPor) .

curriculo:sendoElaboradoPor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Trabalho ;
      rdfs:range curriculo:Estudante .

curriculo:TextoJornal
      a       owl:Class ;
      rdfs:subClassOf curriculo:Texto ;
      owl:disjointWith curriculo:TextoRevista .

curriculo:Periodico
      a       owl:Class .

owl:Thing
      a       owl:Class .

curriculo:possuiEdicaoPeriodico
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Periodico ;
      rdfs:range curriculo:EdicaoPeriodico ;
      owl:inverseOf curriculo:ehEdicaoDoPeriodico .

curriculo:possuiEdicaoConferencia
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Conferencia ;
      rdfs:range curriculo:EdicaoConferencia .

curriculo:Pesquisador
      a       owl:Class ;
      rdfs:subClassOf curriculo:Pessoa ;
      owl:equivalentClass
              [ a       owl:Restriction ;
                owl:onProperty curriculo:publicou ;
                owl:someValuesFrom curriculo:Publicacao
              ] .

curriculo:plataforma
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:temDissertacaoMestrado
      a       owl:FunctionalProperty , owl:ObjectProperty ;
      rdfs:domain curriculo:OrientacaoMestrado ;
      rdfs:range curriculo:DissertacaoMestrado ;
      rdfs:subPropertyOf curriculo:temTrabalho .
```

```
curriculo:ehOrientadorDeMestrando
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:Mestrando ;
      rdfs:subPropertyOf curriculo:ehOrientadorDe .

curriculo:anoDeRealizacao
      a       owl:DatatypeProperty ;
      rdfs:subPropertyOf curriculo:ano .

curriculo:ApresentacaoTrabalho
      a       owl:Class ;
      rdfs:subClassOf curriculo:OutrasPublicacoes .

curriculo:Mestre
      a       owl:Class ;
      rdfs:subClassOf curriculo:Titulo .

curriculo:ConferenciaRegional
      a       owl:Class ;
      rdfs:subClassOf curriculo:ConferenciaNacional .

curriculo:registrou
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Patente ;
      owl:inverseOf curriculo:registradoPor .

curriculo:temTitulo
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Titulo .

[]    a       owl:AllDisjointClasses ;
      owl:members (curriculo:ApresentacaoTrabalho
                   curriculo:CursoMinistrado
                   curriculo:MaterialDidatico
                   curriculo:OrganizacaoEvento
                   curriculo:RelatorioTecnico
                   curriculo:Software
                   curriculo:TrabalhoTecnico) .

curriculo:publicouCom
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pessoa ;
      rdfs:range curriculo:Pessoa ;
      owl:propertyChainAxiom
```

```
              (curriculo:ehAutorDe curriculo:temAutor) .

curriculo:EstudanteIC
      a       owl:Class ;
      rdfs:subClassOf curriculo:Estudante .

curriculo:Departamento
      a       owl:Class ;
      rdfs:subClassOf curriculo:Instituicao .

curriculo:registradoPor
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Patente ;
      rdfs:range curriculo:Pessoa .

curriculo:ehEdicaoDaConferencia
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:EdicaoConferencia ;
      rdfs:range curriculo:Conferencia ;
      owl:inverseOf curriculo:possuiEdicaoConferencia .

curriculo:nivelDoCurso
      a       owl:DatatypeProperty ;
      rdfs:range xsd:string .

curriculo:orientou
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:Pesquisador ;
      rdfs:range curriculo:Estudante ;
      owl:inverseOf curriculo:foiOrientadoPor .

curriculo:Estudante
      a       owl:Class ;
      rdfs:subClassOf curriculo:Pessoa .

curriculo:possuiAnais
      a       owl:ObjectProperty ;
      rdfs:domain curriculo:EdicaoConferencia ;
      rdfs:range curriculo:Anais .

curriculo:ArtigoAceito
      a       owl:Class ;
      rdfs:subClassOf curriculo:Artigo .
```

A.2 Qualis

```
@prefix :        <http://qualis.capes.gov.br/qualis-capes.owl#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl2xml:  <http://www.w3.org/2006/12/owl2-xml#> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix qualis-capes:  <http://qualis.capes.gov.br/qualis-capes.owl#> .

qualis-capes:nome_area
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:Area .

qualis-capes:titulo_conferencia
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:Conferencia .

qualis-capes:ehClassificadoPor
      a       owl:ObjectProperty ;
      rdfs:domain
              [ a       owl:Class ;
                owl:unionOf (qualis-capes:Conferencia qualis-capes:Periodico)
              ] ;
      rdfs:range qualis-capes:Classificacao ;
      owl:inverseOf qualis-capes:classifica .

qualis-capes:Periodico
      a       owl:Class ;
      rdfs:subClassOf owl:Thing .

qualis-capes:ISSN
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:Periodico .

qualis-capes:classifica
      a       owl:ObjectProperty ;
      rdfs:range
              [ a       owl:Class ;
                owl:unionOf (qualis-capes:Conferencia qualis-capes:Periodico)
              ] .

qualis-capes:titulo_periodico
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:Periodico .

qualis-capes:Classificacao
```

```
      a       owl:Class .

<http://qualis.capes.gov.br/qualis-capes.owl>
      a       owl:Ontology .

qualis-capes:Area
      a       owl:Class .

owl:Thing
      a       owl:Class .

qualis-capes:classificadoNaArea
      a       owl:FunctionalProperty , owl:ObjectProperty ;
      rdfs:domain qualis-capes:Classificacao ;
      rdfs:range qualis-capes:Area .

qualis-capes:validoParaAnoBase
      a       owl:FunctionalProperty , owl:ObjectProperty ;
      rdfs:domain qualis-capes:Classificacao ;
      rdfs:range qualis-capes:AnoBase .

qualis-capes:ano_base
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:AnoBase ;
      rdfs:range xsd:gYear .

qualis-capes:Conferencia
      a       owl:Class ;
      owl:disjointWith qualis-capes:Periodico .

qualis-capes:estrato
      a       owl:DatatypeProperty ;
      rdfs:domain qualis-capes:Classificacao .

qualis-capes:AnoBase
      a       owl:Class .
```

REFERÊNCIAS BIBLIOGRÁFICAS

ANTONIOU, G.; HARMELEN, F. van. *A Semantic Web Primer*. 2nd. ed. [S.l.]: The MIT Press, 2008.

BERNERS-LEE, T.; HENDLER, J. A.; LASSILA, O. The semantic web. *Scientific American*, v. 284, n. 5, p. 34–43, maio 2001. ISSN 0036-8733. Disponível em: <http://www.scientificamerican.com/article.cfm?id=the-semantic-web>.

BRANDãO, A. A. F.; LUCENA, C. J. P. de. Uma introdução à engenharia de ontologias no contexto da web semântica. 2002.

BREITMAN, K. *Web Semantica: A Internet do futuro*. [S.l.]: LTC, 2005.

GRUBER, T. *What is an Ontology?* Acesso em: 03 de novembro de 2009. Disponível em: <http://www.ksl.stanford.edu/kst/what-is-an-ontology.html>.

GRUBER, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition*, v. 5, p. 199–220, 1993.

GUARINO, N.; GIARETTA, P. Ontologies and knowledge bases: Towards a terminological clarification. In: MARS, N. (Ed.). *Towards very large knowledge bases*. [S.l.]: Amsterdam: IOS Press, 1995.

HEBELER, J.; FISHER, M.; BLACE, R.; PEREZ-LOPEZ, A. *Semantic Web Programming*. [S.l.]: Wiley Publishing, 2009.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence a modern approach*. 2nd international edition. ed. Upper Saddle River, N.J.: Prentice Hall, 2003.

SEGARAN, T.; TAYLOR, J.; EVANS, C. *Programming the Semantic Web*. Cambridge, MA: O'Reilly, 2009. ISBN 9780596802066.

USCHOLD, M.; BENJAMINS, V. R.; CH, B.; GOMEZ-PEREZ, A.; GUARINO, N.; JASPER, R. A framework for understanding and classifying ontology applications. [S.l.: s.n.], 1999. p. 16–21.