UNIVERSIDADE FEDERAL DA PARAÍBA

CENTRO DE CIÊNCIAS APLICADAS A EDUCAÇÃO

DEPARTAMENTO DE CIÊNCIAS EXATAS

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

MELHORANDO O COMPONENTE DE CHECAGEM DE UM SISTEMA DE CRIAÇÃO DE ONTOLOGIAS QUE USA LINGUAGEM NATURAL

Elmano Neves Neto

Orientador: Prof. Me. Yuri Malheiros

RIO TINTO - PB

2014

ELMANO NEVES NETO

MELHORANDO O COMPONENTE DE CHECAGEM DE UM SISTEMA DE CRIAÇÃO DE ONTOLOGIAS QUE USA LINGUAGEM NATURAL

Monografia apresentada para obtenção do título de Bacharel à banca examinadora no Curso de Bacharelado em Sistemas de Informação do Centro de Ciências Aplicadas e Educação (CCAE), Campus IV da Universidade Federal da Paraíba.
Orientador: Prof. Me. Yuri Malheiros.

RIO TINTO - PB
2014

RIO TINTO - PB
2014

Ficha catalográfica preparada pela Seção de Catalogação e Classificação da Biblioteca da UFPB

N511m Neves Neto, Elmano.

Melhorando o componente de checagem de um sistema de criação de ontologias que usa linguagem natural. / Elmano Neves Neto. – Rio Tinto: [s.n.], 2014.

38 f. : il. –

Orientador: Prof. Ms. Yuri Malheiros.
Monografia (Graduação) – UFPB/CCAE.

1. Web semântica. 2.Ontologias – sistemas de informação. 3. Engenharia de ontologia.

ELMANO NEVES NETO

UM SISTEMA DE CHECAGEM AUTOMÁTICA DE ONTOLOGIAS USANDO LINGUAGEM NATURAL

Trabalho de Conclusão de Curso submetido ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal da Paraíba, Campus IV, como parte dos requisitos necessários para obtenção do grau de BACHAREL EM SISTEMAS DE INFORMAÇÃO.

Assinatura do autor:_____

APROVADO POR:

_____

Orientador: Prof. Me. Yuri Malheiros

Universidade Federal da Paraíba – Campus IV

_____

Prof. Me. Vanessa Farias Dantas

Universidade Federal da Paraíba

_____

Prof. Me. Joelson Nogueira de Carvalho

Universidade Federal da Paraíba

RIO TINTO - PB
2014

Aos amigos, colegas e professores, minha eterna gratidão por compartilhar comigo seus conhecimentos.

AGRADECIMENTOS

A Deus por ter dado esta oportunidade, sem ele nada disso teria acontecido.

Agradeço ao meu orientador Yuri Malheiros por toda paciência e conhecimento compartilhado ao longo do curso e no desenvolvimento deste trabalho.

Aos meus pais Glorijane e Elmano, por todo incentivo, confiança e carinho em toda minha vida, para alcançar meus objetivos.

A todos meus amigos, que nunca deixaram de me apoiar, em especial Diego Pessoa, pois ele sempre me ajudou nas dificuldades que tive durante toda a graduação.

E a todos aqueles que, direta ou indiretamente, colaboraram para que este trabalho consiga atingir os objetivos propostos.

RESUMO

Com o surgimento da Web 3.0 os computadores começaram a ter o poder de entender as informações escritas em um documento web. Isso foi possível devido a implementação de semântica nestes documentos. Uma das formas de criar semântica nestes documentos é através de ontologias; mas ainda não é uma tarefa fácil construir ontologias. Existe um sistema na literatura no qual facilita o processo de construção de ontologias, pois para criar ontologias neste sistema se utiliza linguagem natural escrita em Inglês. Este trabalho busca melhorar um dos componentes desse sistema, que é o componente de checagem.

Palavras chave: Web semântica, Ontologias, Engenharia de Ontologia, RDF, OWL.

ABSTRACT

With the emergence of Web 3.0 computers began to have the power to understand the information written into a web document. This was possible due to the implementation of these semantic documents. One way to create these documents is semantics through ontologies; but still not an easy task to build ontologies. There is a system in the literature in which facilitates the process of building ontologies, because ontologies to create this system using natural language in written English. This work seeks to improve one of the components of this system, which is the component check.

Keywords: Semantic web, Ontologies, Ontology Engineering, RDF, OWL.

LISTA DE FIGURAS

Figura 1 – Gráfico entre diferenças da ontologia e base de conhecimento 10
Figura 2 – Gráfico de triplas 24
Figura 3 – Arquitetura do sistema 29
Figura 4 – Código de desenvolvimento da regra *what is the* 32
Figura 5 – Código de desenvolvimento da regra *how many* 33

LISTA DE TABELAS

Tabela 1 – Testes de questões de competência 36

LISTA DE SIGLAS

RDF	*Resource Description Framework*
OWL	*Web Ontology Language*
QC	Questão de Competência
W3C	*World Wide Web Consortium*
XML	*Extensible Markup Language*
SPARQL	*Protocol and RDF Query Language*

SUMÁRIO

RESUMO **7**

ABSTRACT **8**

LISTA DE FIGURAS **9**

LISTA DE TABELAS **10**

LISTA DE SIGLAS **11**

1.INTRODUÇÃO **14**

1.1.MOTIVAÇÃO 14
1.2.OBJETIVOS 16
1.1.1.GERAL *16*

1.1.2.ESPECÍFICO *17*

1.3.ESTRUTURA DO TRABALHO 17

2.FUNDAMENTAÇÃO TEÓRICA **18**

2.1.ONTOLOGIAS 18
2.2.LINGUAGENS PARA CONSTRUÇÃO DE ONTOLOGIAS 20
2.2.1.OWL *20*

2.2.2.RDF *23*

2.3.ENGENHARIA DE ONTOLOGIA 24
2.4.QUESTÕES DE COMPETÊNCIA 25

3.DESENVOLVIMENTO **27**

3.1.IMPLEMENTAÇÃO 27
3.2.COMPONENTES 27
3.3.PRÉ-PROCESSAMENTO 29
3.4.REGRAS *29*
3.4.1.IS A *30*

3.4.2.VALOR DA PRORIEDADE *30*

3.4.3.EXISTÊNCIA *30*

3.4.4.WHAT IS THE 31

3*.4.5.HOW MANY* *31*

3.5.CONSULTANDO A ONTOLOGIA 33

4.RESULTADOS **34**

5.CONCLUSÃO **36**

REFERÊNCIAS BIBLIOGRÁFICAS **37**

1. INTRODUÇÃO

1.1. MOTIVAÇÃO

Nas últimas décadas o número de usuários da Internet vem crescendo de forma significativa a cada ano [1]. Como a internet é livre para a publicação de conteúdo, o número de informações aumenta cada vez mais. Por um lado isto é bom pois temos uma quantidade muito grande de conteúdo publicado, por outro lado os motores de busca como o Google, passam a ter um desafio ainda maior para encontrar o que é relevante no meio de tanta informação. A Web que conhecemos é baseada em documentos, e isto é apenas uma forma de fornecer informações para o usuário. Na web de documentos, as palavras não são estruturadas, então apenas as pessoas têm o entendimento do que está escrito nestes documentos. Na Web tradicional há a carência por uma semântica para que as informações possam ser interpretadas automaticamente por computadores. Caso o computador pudesse entender a semântica das informações, os resultados de pesquisas seriam bem mais precisos, pois eles poderiam entender sobre o que estamos interessados em pesquisar. Por isso, foi pensado em algo que pudesse resolver esse problema, o que deu origem à Web Semântica.

A Web Semântica[2] tem o objetivo de fazer com que os computadores possam ter um melhor entendimento das informações contidas na Web, para que elas não fiquem espalhadas em meio a tanto conteúdo. Uma boa explicação sobre web semântica seria, "A Web Semântica é uma extensão da Web atual, em que a informação é dada contendo um significado bem definido, permitindo que computadores e pessoas trabalhem em cooperação. " [Berners-Lee et al., 2001]. Vemos que a ideia não é nova, no entanto o conceito só veio ganhar forças nos últimos anos.

Existem outras vantagens do uso da Web Semântica, algumas dessas vantagens são:

- Linguagens de consulta: Como a Web Semântica se trata de dados, então seria preciso uma linguagem de consulta. SPARQL é a linguagem de consulta para a Web Semântica.
- Comunicação de Aplicações: A W3C está trabalhando com diferentes setores, por exemplo, nos cuidados da saúde, ciências, governo e energia; para melhorar a

colaboração, pesquisa, desenvolvimento e adoção de inovação através da Web Semântica. A Web Semântica seria uma forma de interligar informações entre as instituições.

Algumas tecnologias foram propostas para implementação da Web Semântica. Estas tecnologias vem sendo cada vez mais usadas por desenvolvedores e são tecnologias recomendadas pela W3C. O Consórcio *World Wide Web* (W3C) é um consórcio internacional no qual organizações filiadas, uma equipe em tempo integral e o público trabalham juntos para desenvolver padrões para a Web. Duas tecnologias mais importantes são:

- *Resource Description Framework* (RDF) [3]: É um dos modelos recomendados pela W3C, é um modelo padrão para intercâmbio de dados na Web, também conhecidos como metadados [4]. Usa uma semântica simples baseada em triplas, o que permite expressar relação semântica entre as coisas.

- *Web Ontology Language* (OWL) [5]: A OWL foi projetada para ser usada por aplicações que precisam processar o conteúdo da informação em vez de apresentar informações para seres humanos. O uso do OWL facilita a interpretação do conteúdo da web para as máquinas. Seu vocabulário é mais extenso que a RDF. Ela possui três sub linguagens, cada uma mais expressiva que a outra: OWL Lite, OWL DL e OWL Full.

Por mais que existam tais padrões e linguagens, ainda não é uma tarefa simples criar especificações de uma ontologia. Grupos de estudos vem desenvolvendo métodos para padronizar estas especificações, esta área vem ganhando cada vez mais força e já foi criada a disciplina de Engenharia de Ontologias para estes estudos. De acordo com Gómez-Perez [6], Engenharia de Ontologia se refere às atividades relacionadas ao processo, ciclo de vida, métodos, metodologias, ferramentas e linguagens para dar suporte ao desenvolvimento de ontologias [6]. Podemos fazer uma analogia com a Engenharia de Software, também possui definições similares, tais como fases, metodologias, ciclo de vida, ferramentas, etc. Com isto, assume-se que se uma ontologia for bem definida e especificada no começo do seu ciclo de vida, torna-se bem mais fácil o seu processo de evolução e a aplicação de correções, em caso de problemas futuros.

Duas atividades importantes dentro da Engenharia de Ontologias são a especificação de requisitos e a de checagem. Para saber se os requisitos foram satisfeitos, no desenvolvimento de ontologias tais requisitos podem ser chamados de QC (Questão de Competência). Um exemplo de questão de competência seria: “Quais características eu devo considerar quando escolher um vinho?”. Checar questões de competência ainda não é uma tarefa fácil de se realizar manualmente, então foram surgindo ferramentas para fazer isto automaticamente, mas ainda existe uma carência dessas ferramentas, principalmente se a ontologia está definida em OWL. No trabalho [17] foi proposto um sistema para construção de ontologias que possui um componente de checagem automática. Esse sistema usa questões de competência escritas em Inglês e checa ontologias OWL.

O objetivo do presente trabalho é melhorar o componente de checagem desse sistema através de regras, assim o sistema será capaz de entender mais tipos de perguntas. Esse sistema facilita muito o trabalho do engenheiro de ontologias, pois ele não precisa saber sintaxes complicadas, nem mesmo checar manualmente arquivos OWL contendo centenas ou milhares de linhas. O componente já existe na versão anterior do sistema e com as melhorias desse trabalho ele será capaz de entender novos tipos de perguntas.

1.2. OBJETIVOS

1.1.1. GERAL

O objetivo deste trabalho é melhorar o componente de checagem do sistema de construção de ontologias.

1.1.2. ESPECÍFICO

Como objetivos específicos, teremos:

- Criar novas regras para o componente de checagem;
- Implementar a consulta de conhecimento definidos na ontologia para os novos tipos de perguntas;
- Testar as perguntas suportadas pelo componente de checagem.

1.3. ESTRUTURA DO TRABALHO

O capítulo I apresenta a motivação e os objetivos do trabalho. No capítulo II abordaremos a estrutura básica para o entendimento do trabalho, explorando conceitos sobre ontologias e engenharia de ontologias e questões de competência. No capítulo III vamos abordar detalhadamente a solução do problema, mostrar como as regras foram implementadas. No capítulo IV serão mostrados os resultados obtidos e por fim, no capítulo V, será a conclusão.

2. FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão descritos os principais conceitos relacionados ao trabalho. São eles: ontologias, as linguagens usadas para descrever ontologias (OWL e RDF) e Engenharia de ontologias.

2.1. ONTOLOGIAS

O termo ontologia vem da filosofia e possui algumas definições. De acordo com Almeida et al. [23], ontologia é o ramo da metafísica que estuda os tipos de coisas que existem no mundo. Segundo Blackburn et al. [24], é também a parte da filosofia que trata da natureza do ser, ou seja, da realidade, da existência dos entes e das questões metafísicas em geral.

Na Ciência da Computação, segundo Tom Gruber [10], uma ontologia é uma especificação formal e explícita de uma conceitualização compartilhada. Em outras palavras, seriam os conceitos e outras entidades que devem existir em um domínio de interesse, bem como a relação entre eles.

Uma ontologia pode ser explícita de várias formas: em linguagem natural (extremamente informal), em linguagem natural mas de uma forma estruturada e restrita (semi-informal), em uma linguagem artificial e formalmente definida para a representação (semiformal) ou numa linguagem formal com completude (rigorosamente formal). Ushold e Gruninger [14].

As definições de Tom Gruber e de Ushold e Gruninger falam de conceitos, mas de poucas características de ontologia, não falam de como ela deve ser representada, fala que sua especificação é formal e explícita, mas não definem uma estrutura de ontologia.

Uma outra definição interessante de ontologia é a de Gómez-Pérez, originada de Swartout, presente no artigo "*Towards Distributed Use of Large-Scale Ontologies*" [26]: "Uma ontologia é um conjunto de termos ordenados hierarquicamente para descrever um domínio que pode ser usado como um esqueleto para uma base de conhecimentos."

A partir da definição de Swartout pode-se extrair algumas informações para uma estrutura de ontologia, pois ela deve possuir um conjunto de termos ordenados hierarquicamente. Também é importante notar a diferença entre ontologia e base de conhecimento, pois a partir de uma estrutura básica para criar a ontologia é que depois podemos usá-la como base de conhecimento.

Para exemplificar melhor, a Figura 1 diferencia ontologia e uma base de conhecimentos. Através da ontologia do componente eletrônico é possível gerar uma base de conhecimento.

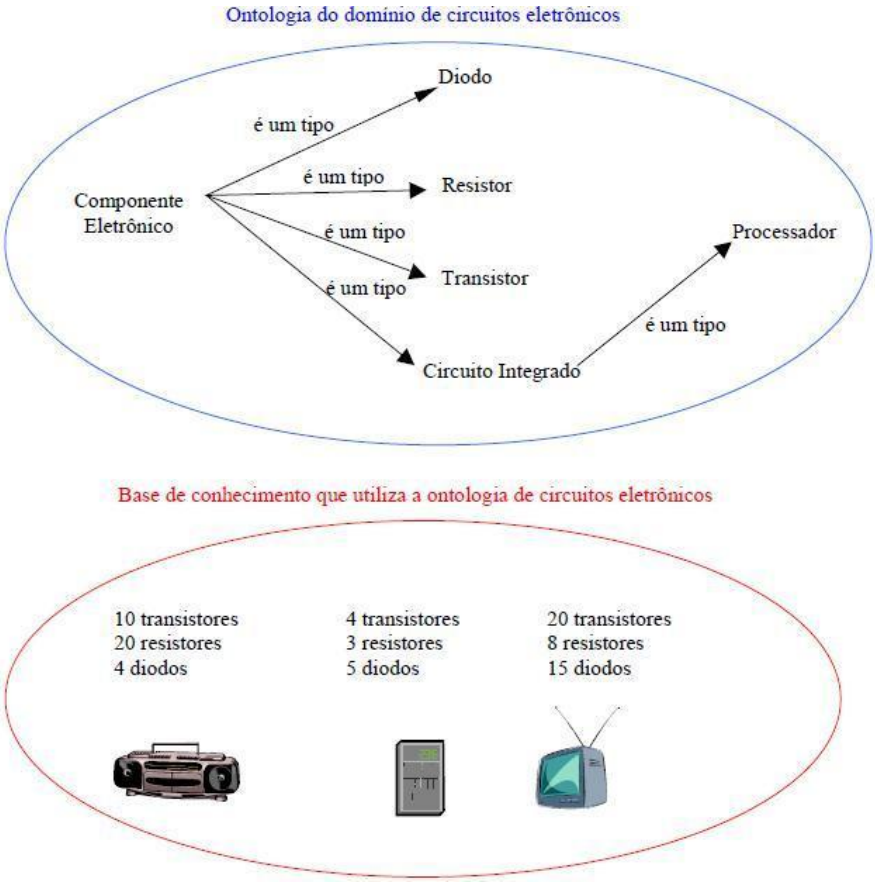

Figura 1 – Gráfico entre diferença de ontologia e base de conhecimento. Fonte: http://www2.dbd.puc-rio.br/pergamum/tesesabertas/0024134_02_cap_04.pdf

Criar ontologias não é uma tarefa simples. Para tal, existem algumas linguagens que surgiram para facilitar o seu desenvolvimento. A seguir serão mostradas algumas dessas linguagens, as quais são recomendadas pela W3C.

2.2. LINGUAGENS PARA CONSTRUÇÃO DE ONTOLOGIAS

Existem diversas linguagens para a construção de ontologias, dentre elas: Ontolingua/KIF (GRUBER, 1993), Loom (MacGREGOR, 1991), RDF (W3C, 2002), DAML + OIL (HAMERLEN e HORROCKS, 2002), OWL (W3C, 2002), etc.

A W3C recomenda a OWL para a construção de ontologias. A OWL é uma revisão da RDF, permitindo regras que não existem nessa linguagem.

2.2.1. OWL

OWL é uma linguagem para criar ontologias na web. Ela foi recomendada pela W3C em fevereiro de 2004. A OWL é uma extensão do vocabulário RDF. Sua sintaxe é baseada em XML, RDF e RDF-Schema. Um documento OWL pode conter descrições de classes, propriedades e suas instâncias. Uma ontologia pode estar ligada a outras ontologias do mesmo documento ou distribuídos em vários documentos.

OWL possui três sub-linguagens que aumentam a expressividade da semântica, que são:

- OWL Lite: para usuários que necessitam principalmente de uma hierarquia de classificação de características de restrição simples. Um exemplo seria: enquanto o OWL Lite suporta restrições de cardinalidade, ela só permite valores de cardinalidade de 0 ou 1. As ferramentas de suporte para OWL Lite devem ser mais simples de fornecer suporte do que seus parentes mais expressivos e fornecer um caminho de migração rápida para enciclopédias e outras taxonomias.

- OWL DL: para aqueles usuários que querem usar o máximo poder de processamento, sem perder completude (toda computação será garantida) e toda computação terminará num tempo finito. OWL DL contém toda linguagem OWL com algumas restrições, como a separação de tipo, uma classe não pode também ser um indivíduo ou propriedade, uma propriedade não pode ser também um indivíduo ou classe. OWL DL é assim chamado devido a sua relação com a lógica de descrição.

- OWL Full: é destinada para usuários que querem liberdade sintática do RDF sem garantias computacionais. O OWL Full garante que uma ontologia possa aumentar o significado do vocabulário pré-definido (RDF ou OWL). Um exemplo de uso seria: uma classe poderia ser simultaneamente uma coleção de indivíduos e um indivíduo no seu próprio direito. É improvável que qualquer software de raciocínio será capaz de suportar todas as funcionalidades do OWL Full.

Sendo assim, vemos que OWL possui um extenso vocabulário, fazendo com que simplifique escrever seus modelos de dados, além disso pode fazer relações entre diferentes ontologias. A seguir, iremos listar alguns casos de uso em que seria interessante usar OWL.

- Portais Web: os portais Web são sites que oferecem informações sobre uma determinada área de interesse. Para identificar tópicos do conteúdo são usadas as metatags, algo que ainda não é eficiente para um sistema de busca. Com o uso de ontologias, os conteúdos podem ter definições que afirmam coisas como "os autores de todas as publicações são pessoas", assim o sistema de busca seria mais eficiente pois faria relacionamento entre as ontologias.

- Coleções de multimídia: As ontologias podem ser usadas para fornecer anotações semânticas para coleções de imagens, áudio ou outros objetos não-textuais. Ontologias multimídia podem ser de dois tipos: meios específicos e de conteúdo específico. Ontologias de Mídia específicas, poderão ter taxonomias de diferentes tipos de mídia e descrever as propriedades dos diferentes meios de comunicação. Por exemplo, o vídeo pode incluir propriedades para identificar duração do clipe e cortes de cena. Ontologias específicas de conteúdo descrevem o assunto do recurso, como a definição ou participantes. Uma vez que tais ontologias não são específicas para a mídia, elas podem ser reutilizadas por outros documentos que tratam do mesmo domínio. Essa reutilização aumentaria a pesquisa que foi simplesmente à procura de informações sobre um determinado assunto, independentemente do formato do recurso. Pesquisas em que tipo de mídia for importante poderiam combinar as ontologias específicas de mídia e de conteúdo específico.

- Gestão de sites corporativos: Grandes empresas costumam ter várias páginas web referentes a coisas como comunicados de imprensa, ofertas de produtos, estudos de caso e etc. Ontologias podem ser usadas para indexar esses documentos e fornecer melhores meios de recuperação.

Para facilitar o entendimento, segue um trecho de Código1. Este código representa as diferentes cores para uma coleção de vinhos:

```
<owl:AllDifferent>
   <owl:distinctMembers rdf:parseType="Collection">
      <vin:WineColor rdf:about="#Red" />
      <vin:WineColor rdf:about="#White" />
      <vin:WineColor rdf:about="#Rose" />
   </owl:distinctMembers>
</owl:AllDifferent>
```

Código 1 – Especificação OWL

2.2.2. RDF

RDF é uma linguagem utilizada para escrever modelos de dados na Web Semântica. Sua estrutura é baseada em triplas, sendo assim: Sujeito – Predicado – Objeto. Estes modelos podem referenciar qualquer coisa, pessoas, objetos físicos, documentos, etc. Assim como o OWL, o RDF também é destinado para processamento de aplicações.

As triplas RDFs podem ser modeladas da seguinte forma:

```
<Bob> <é uma> <pessoa>
<Bob> <é amigo de> <Alice>
<Bob> <nasceu em> <4 de julho de 1990>
```

Podemos também ter uma ligação entre triplas, como mostra a Figura 2:

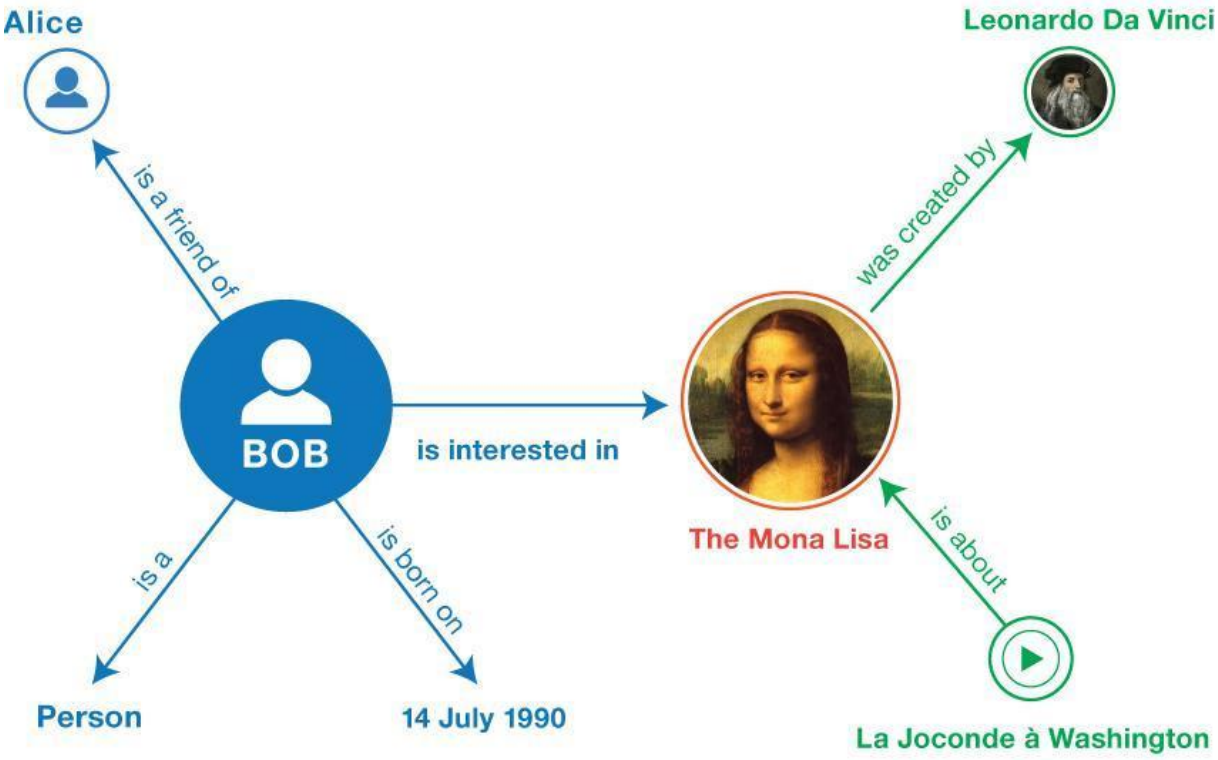

Figura 2 – Gráfico de triplas. Fonte: http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/

Vejamos o seguinte, <Bob> <tem interesse em> <Mona Lisa>, onde Mona Lisa já seria outra entidade que pode estar definida em um arquivo RDF diferente.

Alguns casos onde o RDF pode ser usado:

- Criação de uma rede social onde a descrição das pessoas seria pesquisada em vários sites da Web.
- Fornecer uma maneira compatível com os padrões para a troca de dados entre banco de dados.
- API de interligação de feeds, certificando-se de que os clientes podem facilmente descobrir como acessar mais informações.

O Código2 é um exemplo de código RDF, que define uma classe Fêmea, que é uma sub classe de Animal e tem uma disjunção com Macho:

```
<Class ID="Female">
   <subClassOf resource="#Animal"/>
   <disjointWith resource="#Male"/>
</Class>
```

Código 2 – Especificação RDF

No entanto, não é apenas por existirem tecnologias que facilitem a construção de ontologias que isto seja uma tarefa simples, a seguir é apresentada a disciplina de engenharia de ontologias, que veio para organizar este processo.

2.3. ENGENHARIA DE ONTOLOGIA

Devedzic [12] define que Engenharia de Ontologia abrange o conjunto de atividades realizadas durante a conceituação, concepção, implementação e implantação de uma ontologia. Mizoguchi e Ikeda [13] definem que o propósito da Engenharia de Ontologia é fornecer uma base de modelos de construção de todas as coisas em que a ciência da computação está interessada, e que deve abranger temas como filosofia, representação de conhecimento, design de ontologias, padronização e compartilhamento de conhecimentos, integração de mídia, etc.

Existem algumas metodologias na Engenharia de Ontologias e vamos listar as principais a seguir.

Uschold e Gruninger [14] propuseram uma metodologia cujo o propósito é documentar metodologicamente o processo de desenvolvimento e avaliação de uma ontologia com descrição informal de técnicas para a introdução de técnicas formais.

A metodologia de Fernández-López et al. [9] tem como objetivo melhorar atividades relacionadas ao processo de desenvolvimento de novas ontologias. Este processo de desenvolvimento é dividido em várias atividades. Para a fase de especificação de ontologias, recomenda-se fazer o uso de um documento escrito em linguagem natural e usando questões de competência.

Existe uma outra metodologia, que surgiu no Projeto NeON Haase et al. [15], que seria a evolução de metodologias já existentes, com a intenção de fazer um processo mais colaborativo e distribuído por várias pessoas. Ela também introduz a ideia de QC como uma alternativa para identificação de requisitos de ontologia.

Nas três metodologias, existem várias similaridades, e entre elas é o uso das QCs escritas em linguagem natural. Ainda não existem ferramentas para checar automaticamente numa ontologia definida em OWL, que é a ideia principal do nosso sistema.

2.4. QUESTÕES DE COMPETÊNCIA

Uma das maneiras de determinar o escopo da ontologia é esboçar uma lista de perguntas que uma base de conhecimento com base na ontologia deve ser capaz de responder.Gruninger e Fox [25]. Esta lista de perguntas é uma lista de questões de competência.

Noy e McGuinness [16] mostra como exemplo a seguinte lista de questões de competência para o domínio do vinho:

- QC1: Quais características eu devo considerar quando escolher um vinho?
- QC2: Bordeaux é um vinho tinto ou branco?
- QC2: Qual é a melhor escolha de vinho para carne grelhada?
- QC4: Quais foram boas safras para Napa Zinfandel?

A partir desta lista de perguntas, a ontologia precisa ter várias informações sobre as características dos vinhos, como tipo, classificações, alimentos que são indicados para acompanhar o vinho, etc, para que seja possível responder corretamente a lista de perguntas.

3. DESENVOLVIMENTO

Neste capítulo vamos abordar sobre o componente de checagem do sistema em que estamos melhorando, citado no Capítulo 1. Explicaremos passo a passo como ele funciona e como foram feitas as melhorias.

3.1. IMPLEMENTAÇÃO

Na versão atual do sistema, existem diversos componentes para a construção de ontologia, cada um com suas responsabilidades. Nesse trabalho o nosso foco é melhorar as capacidades do componente de checagem. Este componente é responsável por fazer a verificação de alguma pergunta e retornar alguma resposta.

A linguagem de representação de ontologias usada no sistema é a OWL, pois é uma das linguagens para especificar ontologias mais expressivas, ou seja, ela pode expressar conhecimentos mais complexos que em outras linguagens como a RDF. A OWL também tem por base a DL (lógica de descrição) o que garante que ela seja computável e decidível, ou seja, é possível processar os axiomas de forma viável usando um computador.

A seguir vamos explicar cada componente do sistema detalhadamente.

3.2. COMPONENTES

- Checagem em linguagem natural: O componente de checagem em linguagem natural é responsável por fazer o parser das perguntas que devem ser escritas em Inglês, depois é feita a verificação na base de conhecimentos e em seguida uma resposta é retornada. Em outras palavras, o parser é o processo que o sistema faz para separar cada palavra de uma pergunta e assim cada palavra terá uma verificação para retornar um resultado;

- Gerador de questões: quando o sistema não consegue responder uma pergunta corretamente usando o componente de verificador de linguagem natural, ele

gera perguntas para o usuário, para reunir mais conhecimentos e responder à pergunta inicial feita pelo usuário;

- Construtor de ontologia: todos os novos conhecimentos adquiridos através das questões geradas pelo componente anterior são adicionados à ontologia. Este componente é responsável por transformar as informações do componente anterior em uma linguagem de especificação de ontologias;

- *Tracker*: todas as mudanças que utilizam o sistema são rastreáveis. Este componente cria relações entre todos os axiomas adicionados a uma ontologia e sua correspondente questão de competência.

A Figura 3 exemplifica como os componentes estão separados.

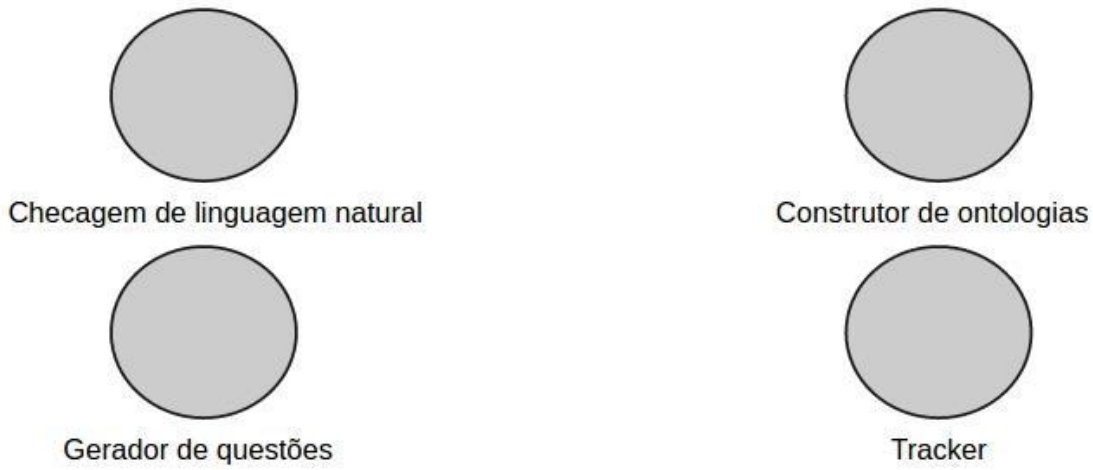

Figura 3. Arquitetura do sistema.

O componentes funcionam da seguinte maneira: O usuário escreve uma QC, caso o sistema saiba responder ele retorna o resultado. Caso o sistema não saiba responder, o componente de gerador de questões é chamado e gera questões para o usuário, o usuário vai respondendo as questões até o sistema conseguir responder a pergunta inicial feita pelo usuário. A partir dessas questões geradas pelo sistema e as respostas do usuário o componente de construção de ontologias é chamado para construir as novas ontologias. O *tracker* serve para rastrear todas ontologias geradas, mapeando os axiomas, assim o usuário vai saber de qual QC foi gerado o axioma.

O foco do trabalho é melhorar o componente de chacagem de linguagem natural, então vamos explicar a seguir como ele funciona.

3.3. PRÉ-PROCESSAMENTO

Para começar a criação ou evolução de uma ontologia, primeiramente temos que escrever uma QC em linguagem natural. Esta abordagem foi escolhida pois facilita para o engenheiro de ontologia, porque usar linguagem natural é mais fácil do que lógica de descrição. O sistema possui alguns tipos de perguntas pré-definidas que ele consegue responder. Estes tipos de perguntas são implementadas usando regras, rótulos gramaticais (substantivos, adjetivos, etc) e de operadores de expressão regular (*, +, ?, e |). Cada palavra da pergunta é analisada e recebe um rótulo usando o NLP Stanford POS Tagger [20]. Esses rótulos servem para categorizar as palavras em classes gramaticais. O próximo passo é verificar se as palavras e seus rótulos combinam com alguma regra em questão implementada no componente. Se não for possível casar com nenhuma regra o sistema retorna que não entende a pergunta do usuário.

3.4. REGRAS

Existem cinco tipo principais de perguntas: *is-a* (é um), o valor da propriedade , existência, *how many (*quantos/quantas) e *what is the*. As duas últimas foram implementadas para este trabalho. A seguir vamos exemplificar cada uso destas regras. As escolhas para o desenvolvimento destas regras foram retiradas do trabalho [11].

3.4.1. IS A

Exemplo de questão *is-a* (é um). Como essa é uma questão "*is-a*", a regra se inicia com *is* seguido de um pronome, adjetivo ou número (por extenso), etc, e logo após a preposição *a*, *an* (um, uns) e novamente um pronome, adjetivo, número (por extenso), etc.

Exemplo de questão *is-a* (é um):

Exemplo: Is red wine a wine? Regras: is (Noun|Adj|Num)+ (a|an) (Noun|Adj|Num)+ Respostas: Yes, no, true ou false

3.4.2. VALOR DA PROPRIEDADE

Primeiramente se inicia com os auxiliares de presente simples para se fazer perguntas *does* ou *do*, em seguida a expressão que pode ser pronome, adjetivo, número (por extenso), etc, em seguida vem o verbo e novamente o pronome, adjetivo, número (por extenso), etc.

Exemplo de questão propriedade valor:

Exemplo: Do bird eat animals? Regras: (does|do) (Noun|Adj|Num)+ Verb (Noun|Adj|Num)+ Respotas: Yes, no, true ou false

3.4.3. QUESTÃO DE EXISTÊNCIA

Este tipo de pergunta se inicia com *which* (qual/quais) e em seguida a expressão com pronome, adjetivo, número (por extenso), etc.

Exemplo de questões de existência:

Exemplo: Which wines exists? Regras: which (Noun|Adj|Num)+

Agora vamos explicar as regras que foram implementadas para o desenvolvimento deste trabalho.

3.4.4. WHAT IS THE

É uma pergunta de característica. Sua expressão se inicia com a sequência de regras *what is the,* seguido de pronome, adjetivo, número (por extenso), etc; em seguida mais uma regra *of*; e por fim mais uma regra de pronome, adjetivo, número (por extenso), etc.

Exemplo de questão:

Exemplo: What is the color of CabernetFranc? Regras: what is the (Noun|Adj|Num)+ of (Noun|Adj|Num)+ Resposta: Red

A Figura 4 mostra o código desenvolvido para a implementação da regra.

```
@Override
public boolean check() {
    Rule whatRule = new Keyword("what");
    Rule isRule = new Keyword("is");
    Rule theRule = new Keyword("the");
    Rule ofRule = new Keyword("of");

    Plus propertyRule = new Plus(new Or(new Noun(), new Adjective(), new CardinalNumber()));
    Plus individualRule = new Plus(new Or(new Noun(), new Adjective(), new CardinalNumber()));

    ArrayList<Token> validTokens = new ArrayList<Token>();
    validTokens.add(whatRule.parse(tokens));
    validTokens.add(isRule.parse(tokens));
    validTokens.add(theRule.parse(tokens));

    ArrayList<Token> propertyTokens = propertyRule.parse(tokens);
    validTokens.addAll(propertyTokens);

    validTokens.add(ofRule.parse(tokens));

    ArrayList<Token> classTokens = individualRule.parse(tokens);
    validTokens.addAll(classTokens);

    if (!validTokens.contains(null)) {
        this.properties = generatePossibleNames(propertyTokens);
        this.individuals = generatePossibleNames(classTokens);

        return true;
    }

    return false;
}
```

Figura 4. Desenvolvimento da regra *what is the*.

3.4.5. HOW MANY

É uma pergunta de quantidade. Sua expressão se inicia com as regras *How many*, seguido de pronome, adjetivo, número (por extenso), etc; mais um conjunto de regras definidas *have a*; e por fim mais uma expressão que pode ser pronome, adjetivo, número (por extenso), etc.

Exemplo:

Exemplo: How many Colors have a Wine? Regras: how many (Noun|Adj|Num)+ have a (Noun|Adj|Num)+ Resposta: 1

A Figura 5 mostra o código desenvolvido para a implementação da regra.

```
public boolean check() {
    Rule howRule = new Keyword("how");
    Rule manyRule = new Keyword("many");
    Rule haveRule = new Keyword("have");
    Rule aRule = new Keyword("a");

    Plus propertyRule = new Plus(new Or(new Noun(), new Adjective(), new CardinalNumber()));
    Plus classRule = new Plus(new Or(new Noun(), new Adjective(), new CardinalNumber()));

    ArrayList<Token> validTokens = new ArrayList<Token>();
    validTokens.add(howRule.parse(tokens));
    validTokens.add(manyRule.parse(tokens));

    ArrayList<Token> propertyTokens = propertyRule.parse(tokens);
    validTokens.addAll(propertyTokens);

    validTokens.add(haveRule.parse(tokens));

    validTokens.add(aRule.parse(tokens));

    ArrayList<Token> classTokens = classRule.parse(tokens);
    validTokens.addAll(classTokens);

    if (!validTokens.contains(null)) {
        this.properties = generatePossibleNames(propertyTokens);
        this.classes = generatePossibleNames(classTokens);

        return true;
    }

    return false;
}
```

Figura 5. Desenvolvimento da regra *how many*.

.

3.5. CONSULTANDO A ONTOLOGIA

O sistema possibilita que usuário escreva as palavras de diversas formas, pois mesmo escrevendo palavras no plural, separadas por espaço ou capitalizações diferentes o sistema é capaz de fazer a combinação da palavra na ontologia, sendo assim o usuário não precisa saber exatamente como um conceito foi definido na ontologia, ou seja, qual o nome dado para os conceitos. Para isso, o sistema testa variações das palavras escritas nas pergunta. Por exemplo, "*red wine*" pode ser combinado com a classe "*RedWine*" na ontologia, ou a palavra "*cows*" pode ser combinada com a classe "*Cow*". O componente utiliza o OWL API [18] e HermiT OWL reasoner [19] para fazer pesquisas por respostas.

O HermiT é um sistema de raciocínio baseado no cálculo de *hypertableus*. Assim como outros sistemas baseados no *tableus*, o HermiT reduz todas tarefas de raciocínio para satisfazer testes de ontologia, e prova como uma ontologia é insatisfatória para tentar construir a abstração de um modelo adequado. [22]

4. RESULTADOS

Para testar o sistema é preciso realizar algumas operações para checar requisitos, adicionar o código e usar o tracker na ontologia de vinhos. Esta ontologia esta disponível no site do Protégé [21].

O Código 3 mostra uma parte do arquivo de ontologias de vinho que está disponível no site do Protégé. Ele mostra características do vinho Bancroft Chardonnay.

```
Individual: vin:BancroftChardonnay

Types:
   vin:Chardonnay

Facts:
   vin:hasMaker vin:Bancroft,
   vin:hasSugar vin:Dry,
   vin:hasBody vin:Medium,
   vin:hasFlavor vin:Moderate,
   vin:locatedIn vin:NapaRegion
```

Código 3. Características de vinho de um arquivo OWL.

O primeiro passo é carregar a ontologia de vinhos e definir uma QC, por exemplo "*is red wine a wine?*" e a resposta esperada será "*true*". Porque esperávamos que a ontologia tenha os axiomas necessários para responder a esta pergunta com essa resposta. Para esse caso o sistema retorna "*ok*", em outras palavras os requisitos da ontologia foram satisfeitos.

Foram feitos cinco testes, três testes para as questões "*what is the*" e dois testes para as questões "*how many*". Uma ontologia de vinhos foi utilizada para estes testes, então as QCs elaboradas foram todas relacionadas a vinhos.

QC	Resposta
what is the color of bancroft chardonnay?	*white*
what is the flavour of bancroft chardonnay?	*moderate*
how many colors have a wine?	1
how many flavours have a wine?	1
what is the location of french wine?	*FrenchRegion*

Tabela 1 – Testes de questão de competência com as novas regras implementadas.

A Tabela 1 mostra que todas as novas regras implementadas para o sistema poder entender novos tipos de perguntas foram testadas e trouxeram a resposta esperada. Além disso, podemos observar a liberdade que o sistema deu de poder escrever as palavras em minúsculo e também de escrever as classes de cada frase separadas por espaço, por exemplo quando foi feita a pergunta *"what is the color of bancroft chardonnay?"* a palavra *bancroft chardonnay* não precisou ser escrita toda junta que é como está definida no arquivo OWL. Isto mostra o quanto o sistema é robusto o suficiente para entender as perguntas mesmo assim.

Com isso tiramos que o objetivo do trabalho foi atingido, fazendo com que o sistema respondesse corretamente as novas regras de perguntas implementadas.

5. CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho buscou melhorar o componente de checagem automática de ontologias de um sistema que apoia o processo de construção de ontologias. Este sistema facilita o trabalho do engenheiro de ontologia, pois, entre outras funcionalidades, ele pode escrever QCs em linguagem natural e não precisa fazer nenhuma iteração manual com a ontologia. O objetivo do trabalho foi alcançado, como vimos na sessão anterior foram feitos os devidos testes com as regras implementadas e todas foram satisfeitas.

Na implementação anterior o sistema já era capaz de responder alguns tipos de perguntas, e após as melhorias feitas neste trabalho ele é capaz de responder perguntas de quantidade "*how many*" e de características "*what is the*". Mas o componente ainda precisa evoluir, e para trabalhos futuros ele poderia ser melhorado nos seguintes pontos:

- O número de perguntas do componente é bem limitado, poderia ser feita uma melhoria para ele poder aceitar outros tipos de perguntas, além de perguntas mais complexas, como por exemplo duas perguntas numa mesma sentença;
- Suportar múltiplas ontologias;

Dar a liberdade para escrever perguntas em outros idiomas, como português, espanhol, etc.

[1] Disponível em: <http://www.internetworldstats.com/emarketing.htm>, acessado em: 07 de julho de 2014.

[2] Disponível em: <http://www.w3.org/standards/semanticweb/>, acessado em: 07 de julho de 2014.

[3] Disponível em: <http://www.w3.org/RDF/>, acessado em: 07 de julho de 2014.

[4] Disponível em: <http://www.w3.org/Metadata/>, acessado em: 07 de julho de 2014.

[5] Disponível em: <http://www.w3.org/TR/owl-features/>, acessado em: 07 de julho de 2014.

[6] Fernandez-Lopez, M., Gomez-Perez, A., Juristo, N.: Methontology: from ontolog-

ical art towards ontological engineering. In: Proceedings of the AAAI97 Spring

Symposium. pp. 33{40. Stanford, USA (Março 1997).

[7] Disponível em: <http://www.w3.org/TR/owl-guide/>, acessado em: 14 de julho de 2014.

[8] Disponível em: < http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/>, acessado em: 14 de julho de 2014.

[9] Fernández-López, M., Gomez-Perez, A., Juristo, N.: Methontology: from ontolog-

ical art towards ontological engineering. In: Proceedings of the AAAI97 Spring

Symposium. pp. 33{40. Stanford, USA (March 1997).

[10] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge

Acquisition, 5(2):199–220.

[11] BEZERRA, Camila; SANTANA, Filipe; FREITAS, Fred. CQChecker: A Tool to Check the Satisfaction of Description Logic Competency Questions on Ontologies, 2014.

[12] Devedzi, V.: Understanding ontological engineering. Commun. ACM 45(4), 136{

144 (Apr 2002), http://doi.acm.org/10.1145/505248.506002.

[13] Mizoguchi, R.: A step towards ontological engineering. In: 12th National Confer-

ence on AI of JSAI. pp. 24{31 (1998).

[14] Uschold, M. e Gruninger, M. (1996). Ontologies: principles, methods, and applications. Knowledge Engineering Review, 11(2):93–155.

[15] Haase, P., Lewen, H., Studer, R., Tran, D. T., Erdmann, M., d'Aquin, M., e Motta, E.

(2008). The neon ontology engineering toolkit. In WWW 2008 Developers Track.

[16] Noy, N. F. e Hafner, C. D. (1997). The state of the art in ontology design: A survey and comparative review. AI Magazine, 18:53–74

[17] MALHEIROS, Yuri; FREITAS, Fred. A Method to Develop Description Logic Ontologies Iteratively with Automatic Requirement Traceability, 2014.

[18]Horridge, M., Bechhofer, S.: The owl api: A java api for owl ontologies. Semant. web 2(1), 11{21 (Jan 2011), http://dl.acm.org/citation.cfm?id=2019470.2019471

[19] Royce, W.W.: Managing the development of large software systems: concepts and techniques. In: Proceedings of the 9th international conference on Software Engineering. pp. 328{338. ICSE '87, IEEE Computer Society Press, Los Alamitos, CA, USA (1987), http://dl.acm.org/citation.cfm?id=41765.41801

[20] Sure, Y., Erdmann, M., Angele, J., Staab, S., Studer, R., Wenke, D.: Ontoedit: Collaborative ontology development for the semantic web. In: Proceedings of the First International Semantic Web Conference on The Semantic Web. pp. 221{235. ISWC '02, Springer-Verlag, London, UK, UK (2002), http://dl.acm.org/citation.cfm?id=646996.711413

[21] Disponível em: http://protege.stanford.edu/download/ontologies.html

[22] HORROCKS, Ian; MOTIK, Boris; WANG, Zhe. The HermiT OWL Reasoner.

[23] ALMEIDA, M; BAX, M. Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. Revista Ciência da Informação, 32(3), 2003.

[24] BLACKBURN, S; MARCONDES, D. Dicionário Oxford de Filosofia. Tradução de Murcho et al., Rio de Janeiro, Jorge Zahar, 1997.

[25] Fox, M. S. and Gr¨uninger, M. (1994). Ontologies for enterprise integration. In CoopIS, pages 82–89

[26] William Swartout, Ramesh Patil, Kevin Knight, and Tom Russ; Toward Distributed Use of Large-Scale Ontologies, Proceedings of the 10th Banff Knowledge Acquisition Workshop; Banff, Alberta, Canada; November 9-14, 1996.