Joel Augusto de Oliveira

MÉTODO DE AVALIAÇÃO DE ONTOLOGIAS ATRAVÉS DE ALINHAMENTO SEMIAUTOMÁTICO BASEADO EM VISUALIZAÇÃO DE INFORMAÇÃO

Belo Horizonte

2012

Joel Augusto de Oliveira

MÉTODO DE AVALIAÇÃO DE ONTOLOGIAS ATRAVÉS DE ALINHAMENTO SEMIAUTOMÁTICO BASEADO EM VISUALIZAÇÃO DE INFORMAÇÃO

Dissertação apresentada ao programa de Pós-Graduação em Ciência da Informação da Escola de Ciência da Informação da UFMG, como requisito parcial à obtenção do título de Mestre em Ciência da Informação.

Linha de pesquisa: Organização e Uso da Informação.

Orientador: Prof. Dr. Maurício Barcellos Almeida

Belo Horizonte

Escola de Ciência da Informação da UFMG

2012

Oliveira, Joel Augusto de.

O48m Método de avaliação de ontologias através de alinhamento semiautomático baseado em vizualização de informação [manuscrito] / Joel Augusto de Oliveira. – 2012.
206 f. : il., enc.

Orientador: Maurício Barcellos Almeida.
Dissertação (mestrado) – Universidade Federal de Minas Gerais, Escola de Ciência da Informação.
Bibliografia: f. 173-183
Anexos: f. 184-206

1. Ciência da informação – Teses. 2. Ontologias (Recuperação da informação) – Avaliação – Teses. I. Título. II. Almeida, Maurício Barcellos. III. Universidade Federal de Minas Gerais, Escola de Ciência da Informação.

CDU: 025.4.03

Ficha catalográfica: Biblioteca Profª Etelvina Lima, Escola de Ciência da Informação da UFMG

UFMG

Universidade Federal de Minas Gerais
Escola de Ciência da Informação
Programa de Pós-Graduação em Ciência da Informação

FOLHA DE APROVAÇÃO

"MÉTODO DE AVALIAÇÃO DE ONTOLOGIAS ATRAVÉS DE ALINHAMENTO SEMIAUTOMÁTICO BASEADO EM VISUALIZAÇÃO DE INFORMAÇÃO"

Joel Augusto de Oliveira

Dissertação submetida à Banca Examinadora, designada pelo Colegiado do Programa de Pós-Graduação em Ciência da Informação da Universidade Federal de Minas Gerais, como parte dos requisitos à obtenção do título de **"Mestre em Ciência da Informação"**, Linha de Pesquisa: **"Organização e Uso da Informação - OUI"**.

Dissertação aprovada em: 08 de novembro de 2012.

Por:

Prof. Dr. Maurício Barcellos Almeida - ECI/UFMG (Orientador)

Prof. Dr. Renato Rocha Souza - FGV/RJ

Profa. Dra. Gercina Ângela Borém de Oliveira Lima - ECI/UFMG

Aprovada pelo Colegiado do PPGCI

Profa. Renata Maria Abrantes Baracho Porto
Coordenadora Pró-Tempore

Versão final Aprovada por

Prof. Maurício Barcellos Almeida
Orientador

UFMG

Universidade Federal de Minas Gerais
Escola de Ciência da Informação
Programa de Pós-Graduação em Ciência da Informação

ATA DA DEFESA DE DISSERTAÇÃO DE **JOEL AUGUSTO DE OLIVEIRA**, matrícula: 2010661405

Às 09:00 horas do dia 08 de novembro de 2012, reuniu-se na Escola de Ciência da Informação da UFMG a Comissão Examinadora aprovada pelo Colegiado do Programa de Pós-Graduação em Ciência da Informação em 04/10/2012, para julgar, em exame final, o trabalho intitulado ***Método de Avaliação de Ontologias através de alinhamento semiautomático baseado em visualização de informação***, requisito final para obtenção do Grau de MESTRE em CIÊNCIA DA INFORMAÇÃO, Área de Concentração: Produção, Organização e Utilização da Informação, Linha de Pesquisa: Organização e Uso da Informação - OUI. Abrindo a sessão, o Presidente da Comissão, Prof. Dr. Maurício Barcellos Almeida, após dar conhecimento aos presentes do teor das Normas Regulamentares do Trabalho Final, passou a palavra ao candidato para apresentação de seu trabalho. Seguiu-se a argüição pelos examinadores com a respectiva defesa do candidato. Logo após, a Comissão se reuniu sem a presença do candidato e do público, para julgamento e expedição do resultado final. Foram atribuídas as seguintes indicações:

Prof. Dr. Maurício Barcellos Almeida - Orientador	APROVADO
Prof. Dr. Renato Rocha Souza	APROVADO
Profa. Dra. Gercina Ângela Borém de Oliveira Lima	APROVADO

Pelas indicações, o candidato foi considerado APROVADO.

O resultado final foi comunicado publicamente ao candidato pelo Presidente da Comissão. Nada mais havendo a tratar, o Presidente encerrou a sessão, da qual foi lavrada a presente ATA que será assinada por todos os membros participantes da Comissão Examinadora.

Belo Horizonte, 08 de novembro de 2012

Prof. Dr. Maurício Barcellos Almeida
ECI/UFMG

Prof. Dr. Renato Rocha Souza
FGV/RJ

Profa. Dra. Gercina Ângela Borém de Oliveira Lima
ECI/UFMG

Obs: Este documento não terá validade sem a assinatura e carimbo da Coordenadora.

Profa. Renata Maria Abrantes Baracho Porto
Coordenadora do Programa de
Pós-Graduação em Ciência da
Informação - ECI / UFMG

AGRADECIMENTOS

Agradeço minha mãe pela paciência e suporte em tempo integral. Sem ela não somente este trabalho, mas qualquer coisa que tivesse tentado não poderia ter sido feita da mesma forma. Ao meu pai e irmãos por todo suporte e noites a fora me vendo falar do que estava fazendo.

Ao meu orientador Maurício, que não somente me ensinou a produzir este trabalho, mas também me apoiou e trouxe lições de vida que levarei comigo após esta etapa. Que aguentou o desespero e meu medo dentro da dessa aventura no meio acadêmico. Me mostrou obstinação e dedicação a ciência. Muito obrigado!

Aos meus amigos Gabi, Guilherme, Denise e Camila pela enorme paciência, pelas noites a fio me suportando falando da mesma coisa por quase três anos. Por continuar me chamando pra sair mesmo sabendo que não ia ou que, se fosse, ia falar da mesma coisa o tempo todo. Por suportar minha chatura sem fim. Muito obrigado!

Às pessoas importantes que conheci durante esse processo: Luciana, Flavinha e Raisa. Pessoas que quero sempre perto. Que pretendo manter no coração e no convívio.

Ao restante da minha família e amigos que fazem parte do que sou e do que tenho de mais importante na vida.

Enfim, agradeço a todos eu me fizeram companhia.

Muito obrigado!

“A Ciência não estuda ferramentas. Ela estuda como nós as utilizamos, e o que descobrimos com elas”

(Edsger Dijkstra)

RESUMO

O presente trabalho teve como objetivo criar um método de avaliação de ontologias baseado em alinhamento entre ontologias e visualização de informação. Esse método visa auxiliar a correta construção de uma ontologia a partir da avaliação de sua interoperabilidade com ontologias já estabelecidas somado à análise visual de suas estruturas e descrições. Para alcançar o objetivo, alguns processos de avaliação e alinhamento foram desenvolvidos, bem como um suporte tecnológico. Os processos basearam-se na alta probabilidade de que duas ontologias de mesma base e domínio serem intercomunicantes, ou seja, deve existir interoperabilidade entre elas. Tais processos foram divididos em três: tratamento, sistema e avaliação, em que cada um representou um aspecto do método de avaliação. O primeiro representa os processos de tratamento e avaliação; o segundo representa o suporte tecnológico; e o terceiro representa a junção dos dois primeiros. Um sistema de alinhamento e visualização de Ontologias Modeladas na Linguagem OWL (*Ontology Web Language*), que retrata a segunda parte do método de avaliação. Através dele as análises de estruturas, descrições e alinhamento foram feitas. Isso permitiu a consolidação dos processos através da aplicação do que foi desenvolvido como a terceira parte do processo. Ao final, trouxe-se uma apresentação dos processos e sistemas criados, além de um estudo de caso para averiguação da aplicabilidade, vantagens e desvantagens do método.

Palavras Chave:
Ontologia, OWL, Avaliação, Alinhamento, Processos, Sistemas

ABSTRACT

The present work has as a major goal to develop an evaluation method based on ontology alignment and information visualization. This method aims to assist the correct construction of an ontology, based on the evaluation of its interoperability with a well established ontology in the field added to visual analysis of its structures and descriptions. In order to achieve this goal, some assessment and alignment processes and their technological support were developed. The processes are based on the high probability in which two ontologies possessing the same base and domain are inter-connecting, ie, there must be interoperability between them. These processes weredivided into three pillars: treatment, system, and evaluation, each representing an aspect of the evaluation method. The first is the process of treatment and evaluation, the second represents the technological support and the third represents the union of the first two pillars. An alignment and visualization system of ontologies modeled by OWL (Web Ontology Language), which is the second part of the evaluation method, was developed to support the evaluation method and ontology visualization. It is through the system that the analysis of structures, descriptions and alignment were done. This allows the consolidation of processes to occur by applying the third developed pillar. In conclusion, a presentation with the created processes and systems are shown, as well as a case study to investigate the method's applicability, and advantages.

Key words:
Ontology, OWL, Evaluation, Alignment, Processes, System

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 - Introdução

O campo do estudo sobre ontologias tem-se mostrado cada vez mais expressivo no escopo da Ciência da Informação. Ontologias têm recebido grande atenção em pesquisas sobre modelagem de dados, organização da informação, integração de fontes heterogêneas, dentre outras. A necessidade de ontologias pode ser vista como consequência do crescente volume de informação, praticamente exigindo a consideração de ferramentas da tecnologia da informação. Aliada às constantes inovações tecnológicas dos últimos vinte anos para acesso e disseminação de informação, a produção de conhecimento se tornou uma atividade dinâmica capaz de gerar progresso. Historicamente, a Ciência da Informação se propõe a lidar com a produção de conhecimento em diversas áreas, atuando como uma metaciência (BATES, 1999). As complexas atividades relacionadas à produção e disseminação de conhecimento geram a necessidade do tratamento da informação produzida, especialmente àquela referente aos processos internos de uma instituição. Também surge a necessidade do controle do que é produzido pela organização. Por exemplo, em uma empresa de grande porte de geração de energia, além do controle das demandas de geração e transmissão de energia existe ainda a necessidade de controle dos processos internos para manutenção da competitividade. Isso inclui a reflexão sobre como tratar e disseminar informação.

De fato, tanto as atividades e as relações de produção, quanto o contexto e as relações sociais, carecem de informações acessíveis, pressupondo a necessidade de organização das informações. Ferramentas de classificação e organização da informação, em última instância, atendem a necessidades dos usuários independentemente do quão complexa pode ser essa ferramenta em termos computacionais. O que importa na verdade é o sucesso ou insucesso do usuário em sua busca. Um aumento no volume de dados sem contextualização e tratamento leva à sobrecarga de informação, quando o usuário não consegue mais dar sentido ao conjunto de dados recuperado.

Uma importante motivação do presente trabalho reside no uso de potencialidades tecnológicas no tratamento da informação e do conhecimento, uma vez que as aplicações da tecnologia da informação estão presentes em grande parte das atividades humanas. O uso de ferramentas para classificação e organização da informação levou à criação de linguagens para representação em sistemas. Dessa forma, tornou-se possível o uso do aparato tecnológico pelos usuários de esquemas de classificação da informação. As ontologias são uma alternativa a essa demanda, possibilitando a criação de modelos abstratos da realidade com fins de

organização e de inferência computacional. Proporcionam uma maneira de classificar as entidades do mundo e representá-las para a máquina.

Nesse contexto cabe citar a título de exemplo, a *Ontology Web Language* (OWL), uma linguagem de representação semiestruturada criada para desenvolvimento de ontologias na *Web* Semântica (W3C, 2004). Uma das principais vantagens no uso da OWL é a padronização que a linguagem proporciona, auxiliando na tarefa de organizar conhecimento. Outra vantagem é que a OWL é um padrão *web*, ou seja, foi criada para o contexto de disseminação de informações que a Internet proporciona, podendo ser utilizada dentro e fora das organizações via intranets ou extranets.

As ontologias permitem modelar conhecimento, pois geram conjuntos organizados de dados significativos em um contexto, auxiliando na recuperação da informação em um domínio específico. Por exemplo, uma ontologia sobre medicina é utilizada em ferramentas manipuladas por profissionais de saúde, apresentando informações consolidadas sobre doenças, anatomia, vírus, dentre outras. Ainda assim, existem problemas em aberto, por exemplo, para grande volume de dados disponível permanece complexa a interpretação e a representação do conhecimento para os usuários finais, uma vez que os métodos de análise comumente aplicados não são eficientes (HEALEY, 2000).

Ontologias estão presentes ainda, como ferramentas, em campos de pesquisa como Ciência da Informação, Sistemas de Informação e Inteligência Artificial (GRUBER,1993; GRUNINGER; FOX, 1995) e como forma de organizar informação, em domínios variados como Medicina, Geografia, Meio Ambiente, Arquitetura e Engenharia.

No sentido de facilitar a interpretação dos dados gerados continuamente, vêm sendo desenvolvidos trabalhos na área de visualização de informação utilizando ontologias, os quais abrangem vários domínios do conhecimento. A visualização de informação não diz respeito, necessariamente, às visualizações computacionais (CARVALHO; DIAS, 2007). A maioria das atividades humanas geram dados, os quais devem ser interpretados. Existe uma grande variação no volume de dados gerados em cada tipo de atividade, o que leva a diferentes necessidades de ferramentas para interpretação destes dados.

Segundo Kiner, Calonego e Buk (2004), tais áreas do conhecimento, as quais geram grande quantidade de dados, podem tirar proveito das técnicas de visualização de informação. Em cada área de conhecimento, a visualização recebe um nome específico como, por exemplo: visualização científica, visualização geográfica, visualização de negócios, visualização de software, dentre outras. Cada um dos tipos de visualização acima citados tem aplicação específica em sua própria área de conhecimento ou de atividades.

A visualização científica é utilizada na interpretação de dados de computação cientifica, dados coletados de experimentos e dados da natureza. Por sua vez, a visualização cartográfica é utilizada para a análise das diferentes características dos dados representados nos mapas, além de auxiliar os usuários produtores de mapas a construí-los de acordo com os princípios de projeto cartográfico (ROBBI, 2000). Em negócios, a visualização de informação é utilizada no suporte à tomada de decisão (KINER; CALONEGO; BUK, 2004). Em visualização de *software*, o objetivo de ferramentas de visualização é facilitar a compreensão do processo de desenvolvimento, pois as técnicas de visualização são utilizadas para construir mecanismos geradores de representações visuais sobre programas e sistemas (SASSO; CHUBACHI; LUZZARDI, 2001).

Existem ainda técnicas de visualização da informação utilizadas para apuração de informação dita genérica, ou seja, aquela que não objetiva necessariamente atender aos usuários especializados. Neste campo, além dos sistemas que podem utilizar comumente as técnicas de visualização – como sistemas de recuperação de informação (SRIs), bibliotecas digitais, sistemas de catalogação – existem também técnicas de transformação de dados abstratos em imagens. Esse processo auxilia na construção e entendimento dos processos organizacionais.

Junto a essas técnicas de visualização, processos de alinhamento entre ontologias também podem contribuir em processos de avaliação, à medida que sua aplicação nos revela a qualidade de uma série de característica esperadas de uma ontologia escrita em OWL. Para aplicação dos processos de alinhamento foram pesquisadas algumas técnicas em voga. Entre elas foram pontuadas com maior ênfase, neste trabalho, as técnicas baseadas em estrutura e técnicas baseadas em *string*. Pois estas permitem, dentro do contexto deste trabalho, melhor conexão entre os passos automáticos e os semiautomáticos do método desenvolvido.

Dentre as diversas técnicas existentes, o presente trabalho aborda a utilização de técnicas para visualização, unidas às técnicas de alinhamento de ontologias, e o contexto de avaliação de ontologias. Busca-se entender como o processo de avaliação de uma ontologia desenvolvida em OWL pode se beneficiar das técnicas de visualização da informação alinhamento de ontologias. Isso foi baseado na idéia de que duas ontologias de mesma base ontológica e descritiva sobre um mesmo assunto devem se comunicar em alguma instância. A busca por tal apoio à avaliação de ontologias reside no fato de que, apesar da existência de inúmeros métodos para avaliação de ontologias descritos na literatura, poucos são realmente sistematizados (ALMEIDA, 2008). A importância dos métodos de avaliação reside na

necessidade de saber se a ontologia utilizada realmente atende aos objetivos para os quais foi criada.

Justifica-se a abordagem adotada no presente trabalho, por meio da constatação de que uma falha na construção de uma ontologia gera um produto final não adequado à aplicação que se destina. Um dos problemas encontrados na avaliação de ontologias diz respeito à corretude do domínio modelado. Nem sempre os especialistas em ontologias têm conhecimento sobre a melhor forma de conduzir o processo de construção e avaliação dessas, uma vez que não possuem conhecimento técnico no domínio em questão.

Nesse contexto, o presente trabalho busca a criação de um método de avaliação de ontologias que mescla alinhamento de ontologias, visualização e avaliação por um especialista. Para criação de tal método, o presente trabalho persegue o seguinte objetivo geral: **desenvolver e testar um roteiro para auxílio à avaliação de ontologias de domínio baseado no suporte de visualização gráfica e alinhamento semiautomático de ontologias**. De forma a atingir o objetivo geral, planejou-se os seguintes objetivos específicos:

- Desenvolver uma ferramenta de visualização e alinhamento de ontologias, preferencialmente criadas em OWL, a qual utiliza a linguagem de programação *Python*;
- Aplicar o conjunto ferramenta e metodologia em ontologias já testadas, da área biológica, e obter dados sobre a concordância do método desenvolvido em relação ao teste anterior;
- Realizar estudo de caso em duas ontologias consolidadas, utilizando o método e ferramentas criadas para verificação dos benefícios da visualização aliada ao alinhamento para a avaliação de ontologias.

O restante deste trabalho está organizado como segue. A seção 2 apresenta um breve levantamento dos métodos utilizados na avaliação de ontologias. Cabe citar que o assunto avaliação de ontologias é considerado como um cenário central para teste do método e ferramentas aqui desenvolvidas, além das principais abordagens da literatura a atividade de *matching* de ontologias. A seção 3 descreve as técnicas de visualização da informação. A seção 4 descreve as bases técnicas de desenvolvimento. Na seção 5 temos metodologia de pesquisa, utilizada na busca pelos resultados. Já na seção 6, apresentamos a aplicação do método desenvolvido e seção 7 temos nossas considerações finais.

2 - Ontologias

Os primeiros passos para o pensamento ontológico foram dados há mais de dois mil anos, com a busca do estudo do ser enquanto “ser”, ou seja, do conhecimento primordial sobre “aquilo que é”. Aristóteles criou sua “filosofia primeira”, a qual recebeu posteriormente a denominação de *metafísica*. O termo *ontologia* começou a ser utilizado para designar o estudo do ser a partir do século XVII, principalmente na filosofia moderna. Filósofos da tradição escolástica cultivaram a ontologia como objeto de estudo (STANFORD..., 2012).

Com Christian Wolff, o termo foi popularizado e ganhou caráter dedutivo, abstrato e estruturado, como o estudo de conceitos do ser. Outros autores, ao se oporem ao que, até então, era considerada ontologia, contrapuseram-na à filosofia primordial e reescreveram seus conceitos. Para Hartmann, a "ontologia analítica e crítica" é distinta da ontologia dos escolásticos e racionalistas, que pretendiam chegar à existência (ou seja, a uma "lógica dos entes") a partir da construção de um saber sobre “essências”. Segundo o autor, a ontologia não é a tentativa de resolver todos os problemas, mas de reconhecer aquilo que é metafisicamente insolúvel. Sendo assim, a ontologia consistiria no estudo da existência ou do essencial ser do ser: no primeiro caso, referindo-se a uma característica singular, ao “todo”; no segundo caso, ao que tem de essencial, de especifico em cada ser (STANFORD..., 2012).

A partir da década de 1970, a comunidade de Inteligência Artificial (IA) adota ontologias na modelagem de bases de conhecimento, utilizando-as para descrever conceitos e as relações entre eles. Além da IA, a pesquisa em gestão da informação e do conhecimento encontrou na ontologia uma ferramenta de modelagem para descrição de domínios para sistemas de informação. A ontologia é utilizada para se referir a termos e conceitos que descrevem uma área de conhecimento ou a representam.

As ontologias permitem ver a realidade através de conceitos necessários para descrição de um domínio. Dentro desse domínio, utilizam-se os termos organizados na ontologia para a descrição dos relacionamentos entre os conceitos. Isso permite modelar uma abstração em uma base de conhecimento. Mesmo com atualizações, a ontologia não se altera desde que o domínio seja o mesmo.

Segundo Guarino (1998, p.04):

> [...] ontologia se refere a um artefato constituído por um vocabulário usado para descrever certa realidade, mais um conjunto de fatos explícitos e aceitos que dizem respeito ao sentido pretendido para as palavras do vocabulário. Este conjunto de fatos tem a forma da teoria da lógica de primeira ordem, onde as palavras do vocabulário aparecem como predicados unários ou binários.

Descreve-se assim a capacidade da ontologia em modelar a realidade, na qual o "vocabulário" serve para caracterizar "coisas" de certo domínio e a interação entre essas "coisas". Qualquer domínio, abstrato ou concreto, pode ser representado.

Já Smith (2003) nos traz que ontologias podem ser consideradas artefatos de *software*, quando se considera o campo de sistemas de informação, que foi projetado para execução de tarefas pré-determinadas. Isto significa que elas estão inseridas dentro de um contexto computacional que é delimitado em escopo de ação, contexto e recursos disponíveis.

Sowa (2006) também nos traz uma definição de ontologias na qual ele propõe que ontologia é o estudo das categorias das coisas que existem ou deveriam existir em um domínio. Ontologia seria um catalogo de tipo de coisas que existem ou deveriam existir dentro de um domínio D considerando a perspectiva de uma pessoa que usa a linguagem L com o propósito de falar sobre D.

Gruber (1993, p.03) trouxe uma proposta sobre o que é uma ontologia. Para ele, uma ontologia é uma especificação explícita de uma conceitualização. Ele nos diz que o conceito tem raízes na filosofia, na qual é utilizada como uma "explicação sistemática da Existência". Como um sistema baseado em conhecimento a ontologia leva em conta o que existe, pois o que existe é o que pode ser representado.

As ontologias podem ser de vários tipos. Apesar de não existir uma classificação consensual sobre os tipos de ontologias, Guarino (1998) coloca quatro tipos de ontologias:

- Ontologias de alto nível ou genéricas descrevem conceitos mais genéricos.
- Ontologias de domínio descrevem um vocabulário relacionado com um domínio genérico.
- Ontologias de tarefas especificam as conceitualizações necessárias para execução de uma tarefa específica.
- Ontologias de aplicação descrevem conceitos que dependem do domínio e de uma tarea, geralmente são uma especialização dos dois.

Já Van Heist et al (1997) nos diz que as ontologias podem ser classificadas ou divididas nas seguintes categorias:

- Ontologias terminológicas que especificam os termos que são usados para representar conhecimento no universo do discurso. Podem ser usadas para unificar o vocabulário em um campo determinado.
- Ontologias de informação especificam a estrutura de armazenamento de bases de dados. Oferecem um marco para armazenamento padronizado de informações.
- Ontologias de modelo de conhecimento especificam conceitualizações do conhecimento. Contem uma rica estrutura interna e podem estar ajustadas para o uso do conhecimento que descrevem.

As ontologias podem ser escritas em várias linguagens, mas mantêm semelhanças em sua estrutura. Todas elas descrevem classes, atributos e relacionamentos, podendo também descrever objetos específicos de cada classe. Por exemplo, a classe “Animal”, com os atributos que a descrevem e os “relacionamentos” entre esses atributos. No caso dessa classe, um exemplo de objeto seria “Leão”.

Os componentes básicos de uma ontologia segundo Gruber (1996) e Noy e Guinness (2001) são objetos (instâncias), classes, atributos, relações e axiomas (restrições). Explica-se abaixo cada um dos componentes:

- Objeto: um objeto é um exemplo, uma instância de uma classe. Apesar de o objetivo de uma ontologia não ser a determinação de um objeto específico, é possível também representá-lo. Objetos atendem às restrições, relações e têm alguns ou todos os atributos de uma classe. Em todos os casos, os objetos atendem às restrições e aos relacionamentos entre características de cada classe, mas não necessariamente atendem a todos os atributos. Um “Leão” não tem todas as características que podem definir todos os tipos de seres vivos, mas tem características e restrições básicas definidoras de um “ser vivo genérico”. “Capacidade de produzir descendentes” seria um exemplo de restrição.
- Classes: de forma semelhante à programação orientada a objetos, uma classe na ontologia é uma abstração de dados que reúne as definições de um conjunto de objetos similares em suas características, relações e restrições, ou seja, é um conjunto que contém objetos e atributos, além de outras classes. Como na orientação a objetos, as classes podem conter e estar contidas em outras classes. Por exemplo, “Mamífero” pode ter a

classe “Felino”, que contém o objeto “Leão”, uma vez que a classe “Felino” tem todas as características para pertencer a “Mamífero” (ter glândulas mamárias, por exemplo). Já a classe “Mamífero” está contida na classe “Ser Vivo”, que é definida com todos os atributos, relações e restrições necessários para designar mamíferos. Alguns problemas são comuns nas definições de como as classes podem comportar ou pertencer a outras classes: uma classe pode conter ela mesma? uma única abstração pode definir tudo? Para contornar tais problemas, são usadas restrições nas relações, de forma a não comprometer as possibilidades de inferência computacional pretendidas com o uso da ontologia.

- Relação: determina classes ou objetos que se relacionam. As declarações são descritas por atributos. Em sua maioria, as relações são definidas por um valor que, na realidade, é um outro objeto dentro do modelo. As relações geram significados dentro do domínio, ou seja, o conjunto de relações gera a semântica de uma ontologia.
- Restrições: determinam como são as relações entre Classes e Objetos, e, consequentemente, determinam limites para os domínios. Para ser “Mamífero” é preciso ter “Glândulas Mamárias”. Essa restrição fundamenta um axioma e os axiomas são normalmente usados para representação das restrições. Restrições têm aplicações em uma ontologia, podendo auxiliar tanto na descrição da relação entre classes distintas, quanto na melhoria da definição semântica.

2.1 - Avaliação de Ontologias

A presente subseção é destinada à apresentação de pesquisas envolvendo avaliação de ontologias. Para tal são apresentadas metodologias que guiam o processo de avaliação de uma ontologia. Não se pretende um levantamento exaustivo, uma vez que, conforme citados anteriormente, a avaliação de ontologias é apenas o “cenário” onde se pretende testar as técnicas de visualização.

Conforme Almeida (2009) é possível encontrar na literatura vários métodos de avaliação de ontologias, mas ainda não existe um método unificado e formal. Uma dificuldade é o fato do processo de produção de ontologias ser ainda mais artesanal do que científico. Segundo Gómèz-Pérez (2001) a construção de ontologias realmente não é sistematizada. Isso ocorre porque cada equipe de desenvolvimento utiliza seu próprio método para a criação de ontologias. Mesmo que diferentes tais metodologias apresentam como fator comum um conjunto básico de atividades: a identificação de propósito da ontologia, a aquisição de conhecimento e a necessidade de avaliação de ontologia.

Brank, Grobelnik e Mladeni`c (2006) citam quatro categorias de avaliação de ontologias: aquelas baseadas na comparação da ontologia a um *Gold Standard* (MAEDCHE; STAAB, 2002); aquelas baseadas no uso da ontologia em uma aplicação e a avaliação dos resultados (PORZEL; MALAKA, 2004); aquelas envolvendo comparações com uma fonte de dados, uma coleção de documentos, por exemplo, sobre o domínio a ser coberto pela ontologia (BREWSTER et al, 2004); aquelas onde avaliação é feita por pessoas que tentam averiguar quão bem a ontologia atinge um conjunto pré-definido de critérios, padrões, requisitos (LOZANO-TELIO; GÓMEZ-PÉREZ, 2004).

2.1.1 - Avaliações baseadas em “padrão de ouro” e tarefas

Maedche e Staab (2002) apresentam uma abordagem baseada em medidas de similaridade para comparar ontologias. As medidas de similaridade são propostas nos níveis lexical e conceitual. No nível lexical, comparam-se duas entradas lexicais, de acordo com a distância de *Levenshtein*. Funciona da seguinte maneira: usando um algoritmo de programação dinâmico mede-se o número mínimo de inserções, substituições e eliminações necessárias para transformar um conjunto de caracteres em outro. No nível conceitual, para encontrar medidas de similaridade, é necessário investigar as relações conceituais existentes entre os termos. Isso é feito baseado na comparação de taxonomias e na verificação da precisão com que duas relações ou dois conceitos se sobrepõem.

Brank, Grobelnik e Mladeni´c (2006) apresentam uma abordagem baseada em um padrão de avaliação, a qual é voltada para avaliação automática de uma ontologia. Seu foco principal é comparação da extensão da semelhança entre a ontologia e o padrão. Isso é feito considerando a arquitetura das instâncias dentro dos conceitos e baseado na organização

hierárquica dos conceitos propriamente ditos. O metodo descrito pelos autores se assemelha a outros métodos também baseados em comparação com padrões, com a diferença de que a avaliação enfatiza os atributos dos conceitos da ontologia e não na descrição em linguagem natural dos conceitos e instâncias. Não são feitas suposições sobre a representação das instâncias, somente é considerada a distinção presente entre elas . Outro pré-requisito é que a ontologia deve ser baseada no mesmo conjunto de instâncias que o *Gold Standard*.

Elhadad e Netzer (2010) apresentam uma metodologia de avaliação fundamentada no seguinte processo: dada uma instância da ontologia, obtem-se automaticamente grande quantidade de documentos textuais, associada à instância. Com base nestes dados textuais, uma análise linguística automática pode ser realizada para determinar se a ontologia reflete as informações que foram retiradas na primeira parte do processo.

Porzel e Malaka (2004) propõem uma forma de avaliação de ontologias baseada na medida de desempenho. Analisa-se dentro de um conjunto de diversas ontologias seu desempenho na execução de uma mesma tarefa. Para os autores existem poucas formas sistematizadas de avaliação de ontologias, além do fato de que a maioria dos métodos existentes não cobre tudo que é necessário para uma avaliação rápida. Existem alguns problemas na utilização de ontologias, um deles é a comunicação entre sistemas e outro seria o gargalo na aquisição de conhecimento. É proposto uma forma quantitativa para avaliação baseada em tarefas. Para os autores, a principal pergunta a ser feita no processo de avaliação é centradas na tarefa proposta, pois se a ontologia foi produzida para uma tarefa é possível avaliar se esta foi bem executada.

Ainda nesse método, Porzel e Malaka (2004) colocam que as ontologias podem ser averiguadas em três diferentes níveis: o taxonômico, o vocabular e das relações não taxonômicas. Os seguintes aspectos são recomendados para a avaliação: erros de inserção que são conceitos supérfluos na ontologia, e erros de dileção que são conceitos ausentes. Com as tarefas apropriadas e algoritmos independentes trabalhando na ontologia para resolver essas tarefas e prover os parâmetros de avaliação, pode-se calcular a quantidade de erros relativos aos aspectos citados. A avaliação baseada em tarefas tem vários elementos: a tarefa precisa ser complexa o suficiente para a avaliação da ontologia; existe a necessidade de existir pelo menos uma ontologia para ser avaliada e é necessário utilizá-la; o padrão ouro fornece as respostas que utilizadas como base de comparação.

2.1.2 - Avaliações baseadas em comparação de corpus de dados e aplicações

Corcho et al. (2004) afirmam que, como qualquer outro recurso de aplicação de *software*, o conteúdo de uma ontologia deve ser avaliado previamente, (re)utilizando em outros aplicativos de ontologia. Dessa forma, enfatizam que não é adequado publicar um software, ou programar um software que se baseie em uma ontologia escrita por terceiros sem antes avaliar seu conteúdo. A avaliação de ontologia é uma atividade importante que deve ser feita fora do ciclo de vida da ontologia.

Outra proposta para aplicação de uma avaliação de ontologias baseada em tarefas é fornecida por Yu e Tam (2007) que adotam as categorias de navegação da *WikiPédia*[1]. Para realizar a abordagem baseada em tarefas aplicada à avaliação de ontologias, modela-se a tarefa sobre a apresentação de um espaço de informação estruturado em uma determnada categoria dada. O processo é similiar a maneira que os usuários navegam através das categorias da *WikiPédia*.

Sabou et al. (2006) explicam como realizar a seleção de ontologias ou de componentes de uma ontologia. Deve-se atender a certos requisitos especificados, definidos quando existe a necessidade de uma ontologia mais abrangente e bem determinada. Para tal, é necessário a avaliar a ontologia, cujos requisitos são identificados. Poucos requisitos identificados para a *Web* Semântica são encontrados nos atuais métodos de avaliação, já que estes foram feitos para lidar com uma ontologia, não um conjunto delas, e a maioria desses processos não é automático como, por exemplo, o *Ontoclean*.

Gangemi et al. (2006) propõem dois modelos para avaliação de ontologias: a criação de uma meta-ontologia e uma ontologia para seleção de ontologias. No primeiro caso, a meta-ontologia cria a possibilidade de se identificar as três medidas principais para a avaliação de ontologias. São elas: i) métricas estruturais: enfatiza a sintaxe e a semântica formal, os quais são representados como grafos; ii) métricas funcionais: são baseadas no uso pretendido para a ontologia; iii) métricas de usabilidade: são relacionadas ao nível de marcação da ontologia considerada, isto é, apresentam-se metadados sobre a ontologia e seus elementos. A meta-ontologia é complementada pela *oQual* que permite diagnóstico de elementos, processos e atributos da ontologia em avaliação.

[1] *WikPédia* é uma enciclopédia livre *online,* cujos verbetes e artigos podem ser elaborados e modificados pelos usuários de forma colaborativa Cf. WIKPÉDIA. Disponível em: <http://pt.wikipedia.org/wiki/Wikip%C3%A9dia>. Acesso em: 10 set. 2012.

Brewster et al. (2004) expõem a existência de problema na avaliação de ontologias: não se sabe exatamente o que está sendo avaliado. Avaliação de ontologias não pode ser comparada às tarefas básicas de recuperação de informação e avaliação de linguagem natural, pois a noção de precisão e revocação não podem ser facilmente utilizadas. Não existe um conjunto claro de "conhecimento a ser levantado" porque o mesmo conjunto de informações pode gerar diferentes interpretações. Uma abordagem é dividir a avaliação em partes: as relações, as classes e os axiomas. A ontologia é assim vista como uma abstração de um conjunto de textos (em linguagem natural) que descrevem um domínio específico e têm como base reverter esse processo de abstração.

Brewster et al. (2004) ainda propõem encontrar relações entre um texto em linguagem natural e os conceitos da ontologia. Uma solução é avaliar ontologias sob a perspectiva da aplicação para a qual foi projetada. Para isso é necessário estabelecer um conjunto claro de aplicações que permitirá avaliar várias ontologias com o propósito de saber qual está em um bom grau. A comparação ocorre entre uma ou mais ontologias com um corpus de texto, ao invés de com outras ontologias. Para isso pode ser feita uma extração automática de termos do corpus e uma contagem do número de aparições desses mesmos termos no corpus e na ontologia. Três passos da avaliação são a identificação de termos, a expansão das questões e o mapeamento de ontologia. Ao comparar o corpus com uma ontologia, contam-se quantos termos na ontologia estão entre os itens lexicais marcados.

Gómez-Pérez (2001) ainda sugere que a avaliação de ontologias se refere à correta construção do conteúdo de uma ontologia, o que significa garantir que suas definições (linguagem formal e linguagem natural) implementam corretamente os requisitos e questões de competência de uma ontologia. Questões de competência (GRUNINGER; FOX, 1995) exploram o domínio em questão para avaliar a capacidade da ontologia em resolver problemas. Para que isso seja possível o método de avaliação leva em conta cada axioma e definição, o conjunto de definições e de axiomas que estão explicitamente definidos, as definições importadas e as definições que podem ser inferidas de outras definições da ontologia.

A metodologia *OntoClean* (GUARINO; WELTY, 2006) fornece diretrizes para a avaliação baseadas em princípios ontológicos formais da ontologia metafísica. A metodologia é usada para avaliar taxonomias, detectando modelagem inconsistente em relações *is-a*. São definidas meta-propriedades, as quais impõem restrições à estrutura taxonômica de uma ontologia.

Fernández, Cantador e Castells (2006) apresentam um sistema para avaliar qual ontologia de um repositório é a mais adequada para representar um domínio, a partir de uma descrição informal. O ambiente é dividido em três componentes: o primeiro recebe a descrição do problema baseada em um conjunto de termos; o segundo aplica critérios automáticos de avaliação às ontologias de forma a determinar qual se adequa melhor ao problema; o terceiro faz uso de avaliações manuais em ontologias, as quais incorporam a avaliação colaborativa de usuários. Esses componentes correspondem, respectivamente, às três fases de utilização do sistema: definição do problema, recomendações do sistema e avaliação colaborativa.

Nessa mesma linha, Tartir et al. (2006) apresentam uma ferramenta para avaliação de ontologias chamada *OntoQA*, que utiliza métricas para avaliação divididas em duas dimensões: métricas do esquema e métricas das instâncias. A primeira dimensão avalia o projeto da ontologia e seu potencial de representação do conhecimento; a segunda dimensão avalia o posicionamento das instâncias dentro da ontologia de acordo com o conhecimento levantado. A primeira dimensão, por sua vez, é dividida em duas métricas: diversidade de relacionamentos, que mostra a quantidade de diferentes relacionamentos dentro de uma ontologia e profundidade de esquema, que descreve a distribuição das classes dentro da árvore. A segunda métrica é dividida em tres subdivisões: métricas que avaliam o posicionamento geral das instancias em relação ao esquema; métricas de classe-específica que avaliam as instâncias de uma classe específica e compara-a com instâncias de outras classes; e métricas de relacionamentos específicos que avaliam as instâncias de um relacionamento específico e compara-o a instancias de outros relacionamentos.

2.2 - Interoperabilidade de ontologias e sistemas

As ontologias permitem a construção de informações estruturadas e livres de ambiguidades, sendo moldadas de tal forma que permitem o processamento automático (FELICISSIMO, 2004). Além do processamento, o formalismo alcançado com uma ontologia tem sido considerado uma saída para necessidade de comunicação entre sistemas (EUZENAT; SHVAIKO, 2005). O volume de informações e dados diferentes, para o tratamento dos vários domínios do conhecimento, gerou uma grande quantidade de sistemas

para o tratamento desses mesmos dados (VALIATI, 2008), no entanto, algumas situações surgiram em decorrência desse elevado número de sistemas.

Os sistemas são criados para efetuar tarefas específicas dentro de um campo de conhecimento, por exemplo, pode existir um sistema para cadastro de usuários do serviço de saúde de um país e outro sistema para controle das fichas médicas em si. Nesse caso, temos dois sistemas que poderiam compartilhar a sua base de dados além de sua modelagem, mas utilizam diferentes conceitos para problemas específicos de um mesmo domínio e que, segundo Falbo et al. (1998), pode ser considerado um entrave na comunicação entre eles. Assim, quando o sistema é criado independente de outro sistema e existe a necessidade de trocarem informações, deve ser criado um novo sistema, que pode ser um tradutor de dados, para permitir que dois sistemas se comuniquem. Um exemplo deste tipo de sistema é o *Process Integration* do sistema SAP, que consiste na interface de importação de dados de sistemas externos para as bases de dados do SAP (SAP..., 2010, *online*).

O problema da intercomunicabilidade de dados e informações vai além de sistemas de cadastro. Grande esforço de comunicação e tradução de dados entre sistemas poderiam ser evitados utilizando-se uma modelagem conceitual compartilhada. Vários sistemas de mesma base organizacional compartilhariam os primórdios de modelagem somente diferindo nas questões específicas. Isso aplicado ao exemplo anterior significaria que os dois sistemas poderiam ter o cadastro de pessoas modelado de forma unificada. Assim, não existiria a necessidade de dois registros iguais e suas consultas intrasistemas teriam a mesma configuração de dados como repositório.

O uso de ontologia é uma saída dentro do contexto de intercomunicabilidade de sistemas, pois permite a criação de um vocabulário controlado sem ambigüidades, o que auxiliaria na tradução de dados entre vários sistemas. As ontologias podem ser consumidas em *webservices* (SANTANNA, 2008), utilizadas como dicionário de dados, como base de dados propriamente ditas, entre várias outras formas de acesso a dados utilizados dentro de sistemas computacionais. Também podem ser utilizadas na modelagem dos sistemas, trabalhando desta forma nas trocas de informação na raiz, mantendo a possibilidade de intercomunicação desde o início do ciclo de vida de um sistema computacional (ALBUQUERQUE, 2009; GUIZZARDI, 2005; 2006). Dessa maneira, dois sistemas que necessitariam se comunicar teriam seus modelos baseados no mesmo estilo de dados. Salientando que é necessário considerar as diferenças entres várias designações do que se quer modelar, dentro do domínio em questão, para criação de conceitualizações únicas, um dos

esforços dentro da área. (BUSSLER; FENSEL; MAEDCHE, 2002; HOLSAPPLE; JOSHI, 2002).

2.2.1 - Interoperabilidade de Ontologias

Engenheiros de ontologias têm que lidar com vários tipos de ontologias que, como sistemas de informação, também podem ser atingidas pelos problemas que os sistemas de informação sofrem, pois ontologias construídas para o mesmo propósito podem não necessariamente se comunicar. Além disso, existem vários processos dentro do que pode ser chamado de engenharia de ontologias que levam em conta a interoperabilidade de ontologias (EUZENAT; SHVAIKO, 1998, 2005, 2008).

Esses processos são citados por GAL et al, 2003; AUMUELLER et al, 2005; EUZENAT; SHVAIKO, 1998, 2005, 2008, 2012; FAROOQ et al, 2010. Alguns deles são:

- Versionamento de ontologias: versionamento é utilizado para controlar as diferentes versões de um documento ou código específico. Permite manter um registro das diversas alterações feitas no documento ou código, além da possibilidade de recuperação de versões anteriores e de se manter um histórico de desenvolvimento (NOY; MUSEN, 2004; RODDICK, 1996). Ontologias, como os *softwares*, podem ser desenvolvidas de forma colaborativa e distribuída (SHVAIKO; EUZENAT, 2012). Assim, podem-se ter diferentes versões que devem ser controladas e ter o registro do seu desenvolvimento mantido de forma organizada.
- Importação: desenvolvedores de ontologias encontram, diversas vezes, a necessidade de concatenar ontologias distintas de um mesmo domínio de conhecimento (NOY; MUSEN, 2004). Isso pode ocorrer por motivos distintos como necessidade de integração de mais de uma fonte de conhecimento sobre um mesmo assunto, união de um conjunto de ontologias externas, que trazem dados relevantes para o sistema baseado em ontologias sendo construído, entre outros. Para isso, o processo de importação pode ser utilizado. Esse processo consiste em, literalmente, importar uma ontologia dentro da outra, ou seja, tem-se duas ontologias do mesmo tópico e necessita-se de uma ontologia com as informações

dessas duas ontologias. Utilizando um processo de importação pode-se obter uma nova ontologia com as informações necessárias.

- Edição: como em processos de desenvolvimento de softwares o processo de utilização de ontologias pode levar à necessidade de se modificar ontologias já existentes, edição pode ser considerada um processo suporte para importação e casamento de ontologias;
- Atualização (NOY, 2004; RODDICK, 1996): dentro do cenário dos vários domínios em que as ontologias evoluam junto com as outras necessidades. Atualização é o processo pelo qual as ontologias vão sendo melhoradas e evoluem com o tempo para atenderem novas necessidades de modelagem e de interoperabilidade. Como nos sistemas, isso ocorre para atender às demandas dos domínios ou tarefas para que fossem escritas. O processo de atualização se vale do processo de versionamento para o registro e controle das diversas atualizações aplicadas ao longo do tempo. Como exemplo, pode-se colocar uma ontologia médica que tem seus termos redefinidos com novas pesquisas dentro do campo modelado.

2.3 - Matching

As diferentes formas de descrição de entidades no que consiste aos diferentes tipos de representação de dentro dos sistemas de informação (computacionais) são chamadas de heterogeneidade (GAL et al, 2003). Essas diferenças podem trazer dificuldades não só para a interoperabilidade de sistemas ou ontologias, mas também para a recuperação de dados relevantes na web. Entretanto, as ontologias que dariam suporte a esses processos podem ser atingidas por problemas resultantes de dados heterogêneos (FAROOQ et al, 2010).

Para os autores Euzenat e Shvaiko (2005, 1998), Shvaiko e Euzenat (2012) há quatro tipos mais relevantes de heterogeneidade, descritos a seguir:

- Heterogeneidade Sintática: quando as ontologias não são escritas na mesma linguagem ou quando as ontologias são escritas com diferentes formalismos de representação de uma ontologia.

- Heterogeneidade Terminológica: ocorre pela variação dos nomes de entidades similares em ontologias diferentes.
- Heterogeneidade Conceitual: ocorre quando existe diferença na forma de modelar ontologias de um mesmo domínio, modelando de formas diferentes entidades, relações ou axiomas.
- Heterogeneidade Semiótica: também chamada de heterogeneidade pragmática, acontece na interpretação das pessoas sobre como se deve modelar as entidades reais. Pessoas diferentes têm visões diferentes sobre a mesma coisa e, desta forma, geram duas ontologias heterogêneas sobre o mesmo domínio.

A busca para a identificação das diferenças e semelhanças das representações do conhecimento é o foco das técnicas de *Matching* (SHVAIKO; EUZENAT, 2012; FAROOQ et al, 2010; GAL et al, 2003; AUMUELLER et al, 2005; EUZENAT; SHVAIKO, 1998, 2005, 2008), pois a Correspondência entre termos ou *Matching* tem como objetivo evidenciar as similaridades e divergências entre os termos de fontes de dados heterogêneas (GAL et al, 2003). As diversas técnicas para *Matching* também visam ajudar na solução de problemas de interoperabilidade, não só de ontologias, mas de sistemas e outras estruturas de representação de dados como XML, banco de dados, formatos de mensagem; pois têm como objetivo reconhecer as correspondências semânticas entre entidades descritas dentro dessas linguagens ou sistemas (AUMUELLER et al, 2003).

Euzenat e Shvaiko (1998, 2005, 2012) descrevem de maneira muito específica o *Matching* entre ontologias quando definem que esse processo busca identificar as correspondências entre entidades semanticamente semelhantes de diferentes ontologias. Eles também apontam as diversas semelhanças que podem ser distinguidas pelo processo de correspondência entre ontologias. O resultado do processo de *Matching* é o grau de alinhamento entre ontologias, que expressa os diferentes níveis de proximidade entre as entidades modeladas na ontologia.

A literatura aborda várias técnicas para a solução do problema de correspondência entre termos dessas estruturas de dados heterogêneas (EUZENAT; SHVAIKO, 1998), além de diversos algoritmos que servem de auxílio no trabalho de *matching* de ontologias. As técnicas e algoritmos serão descritas nas seções 2.3.1 e 2.3.2 deste trabalho.

2.3.1 - Tipos de *Matching*

Os tipos de *Matching* são divididos em foco nas entidades e foco na estrutura. O primeiro leva em conta as entidades de uma ontologia de forma isolada, fazendo as análises e verificações para cada uma das entidades das ontologias a serem alinhadas. O segundo foco leva em consideração as estruturas das ontologias, ou seja, leva em conta as entidades para a comparação das relações entre elas (EUZENAT; SHVAIKO, 1998, 2005, 2012).

Euzenat e Shvaiko (2005, 2012) fazem uma descrição ampla das técnicas que estão contidas dentro das duas classificações de *Matching* entre ontologias resumidas, conforme é demonstrado a seguir:

Técnicas com foco na entidade (RAHM e BERNSTEIN, 2001):

- Baseadas em *String*s : baseadas na comparação de conjuntos de caracteres que formam os nomes ou as descrições das entidades modeladas dentro das ontologias. Quanto mais similares são os conjuntos de caracteres formadores das descrições ou os nomes das entidades dentro das ontologias, maior seria seu alinhamento (STOILOS et al., 2005; COHEN et al., 2003).
- Baseados em linguagem: ao invés de considerar os caracteres individualmente, é considerada uma formação inteira de letras, que são separadas em palavras. Dessa forma é feita uma comparação entre nomes dos conceitos modelados. Quanto maior a quantidade de coincidências entre os nomes, maior a probabilidade de alinhamento.
- Baseadas nas características dos relacionamentos: foca na análise das relações entre as entidades tais como: cardinalidade, tipos e chaves.
- Recursos lingüísticos: também baseada na comparação de palavras, ou seja, comparação e casamento entre os nomes das entidades modeladas. Este método leva em conta sinônimos e antônimos, além do casamento puro entre os nomes dos termos que estão descritos nas ontologias.
- Reuso de alinhamento: utiliza resultados de alinhamentos anteriores como base para nova comparação. Por exemplo, para fazer comparação entre as ontologias A e B utilizam-se os resultados já obtidos entre uma comparação prévia entre uma ontologia X – A e X – B como base para fazer o alinhamento A – B (DO; RAHM, 2002; AUMULLER et al., 2005; RAHM et al., 2004);

- Técnicas com foco na estrutura apresentadas por (KANG E NAUGHTON, 2003):
- Baseadas em grafo: as estruturas são analisadas como grafos que modelam os termos e suas relações. Normalmente, a comparação é feita entre um par de nós que representam as entidades e suas interrelações. Essa técnica leva em conta a posição dos nós dentro do grafo e também considera que duas ontologias do mesmo domínio têm a mesma representação posicional dos nós e folhas.
- Baseados em taxonomia: também é uma técnica baseada em grafos, mas só leva em conta as especializações entre as relações.
- Repositório de estruturas: guarda as ontologias e pedaços destas junto com as similaridades entre os pares de entidades/estruturas das ontologias. Essa técnica difere do reuso de alinhamento, pois não mantém os alinhamentos propriamente ditos, somente similaridades (RAHM et al., 2004).
- Baseado em modelo: técnica baseada na interpretação semântica das entidades das ontologias a serem alinhadas.

2.3.2 - Técnicas de *Matching*

Nos tipos considerados para alinhamento de ontologia, existem subdivisões de técnicas que podem ser encaixadas dentro de cada um dos subtipos citados dentro de técnicas baseadas em elementos e técnicas baseadas em estrutura. De acordo com Euzenat e Shvaiko (1998) e Ehrig (2004), são subdivididas em: similaridade; distâncias e outras medidas; técnicas baseadas em nome (EHRIG, 2004); baseadas em estruturas; técnicas extencionais e baseadas em semântica. Essas técnicas são descritas a seguir de acordo com que Euzenat e Shvaiko (2005) descrevem em seu livro.

- Distância e outras medidas: para cálculo de distâncias de similaridade de ontologias é necessário ter formas de mensurar essas distâncias e similaridades. Essas formas de mensuração são dividas em formas distintas em: similaridade, dissimilaridade, distancia ultrametrica e dissimilaridade normalizada (EUZENAT; SHVAIKO, 1998, 2005; EHRIG; STAAB, 2004).

- Função normalizada de similaridade: permite a comparação de entidades diferentes de duas ou mais ontologias. Esse tipo de mensuração coloca as dimensões das diferentes entidades em uma mesma escala.
- Dissimilaridade normalizada: coloca dentro dos cálculos de similaridade e dissimilaridade um intervalo real pré-definido como [0..1], ou seja, a transformação da diferença ou similaridade entre duas entidades em números respeita o intervalo real de 0 a 1.

2.3.2.1 Técnicas baseadas em nome

As técnicas de comparação de *string*s são aplicadas aos nomes das entidades e podem ser utilizadas para comparação de nomes ou identificadores das classes. Para comparação de nomes, utiliza-se a comparação direta entre *string*s. Por exemplo, temos uma *string S* e uma *string C*, elas são levadas em conta somente se *S* está contida em *C* e vice-versa. Essa técnica é usada para comparar diretamente nomes de entidades de ontologias distintas (RAHM; BERNSTEIN, 2001; MASCARDI; LOCORO; ROSSO,2010)

A técnica baseada em nome esbarra em dois problemas: sinônimos e homônimos (RAHM; BERNSTEIN, 2001).

- Sinônimos: quando temos duas palavras diferentes com o mesmo significado e são utilizadas para dar nome a classes que representam a mesma coisa, como exemplo carro e automóvel, palavras utilizadas para designar o mesmo objeto real, porém são dois nomes que não se encaixam na definição de comparação, portando não seriam classificados como similares.
- Homônimos: a mesma palavra definindo duas coisas distintas do mundo real, como exemplo, a palavra manga que pode designar a manga da camisa e a fruta manga. Neste caso, haveria a confirmação de similaridade de duas entidades distintas.

2.3.2.2 - Técnicas baseadas em *string*

Entre as técnicas de comparação de nomes, existem as mais avançadas de comparação de *string*s, que permitem melhor averiguação de similaridade. Nestes casos, não é necessário que duas *strings* sejam exatamente iguais. Por exemplo, “livro” e “livraria”, dependendo do algoritmo de casamento de caracteres utilizado, seriam *strings* iguais. , Contudo, permanecem os problemas decorrentes de sinônimos e homônimos citados anteriormente (ZIVIANE, 2004, p.56).

Ziviane (2004) aponta varias técnicas de comparação de caracteres utilizadas na comparação de *string*s, cada uma delas leva em conta uma quantidade de erro aceitável na comparação.

- Algoritmos de comparação exata: formas de comparação de *string*s que retornam positivamente somente se as *strings* forem exatamente iguais.
- Algoritmos que ignoram maiúsculas e minúsculas: formas de comparação que ignoram se existem letras maiúsculas ou minúsculas na *string*, exemplo as *strings* “Paralelepípedo” e “paralelepípedo” seriam consideradas iguais, o que difere dos algoritmos de comparação exata, que considerariam essas duas *string*s diferentes.
- Algoritmos que substituem caracteres: formas de comparação, cuja base de comparação auxilia na substituição de caracteres como letras acentuadas por suas formas básicas, por exemplo, “remédio” e “remedio” seriam consideradas *string*s iguais.
- Algoritmos que aceitam caracteres diferentes à direita, à esquerda ou no meio da *string*: algoritmos que desconsideram pequenas diferenças entre as *string*s. Esses algoritmos permitem algumas trocas de letras dentro de palavras ou sentenças, por exemplo, “livraria” e “livaria” poderiam ser consideradas como a mesma coisa na aplicação desta forma.

Essas técnicas de comparação de *string* permitem melhoria da técnica de comparação de nomes, pois permitem reconhecimento de maior quantidade de formas e diferenciações entre os nomes das entidades.

2.3.2.3 - Comparação de caminho

Além de utilizar a comparação de nomes, a técnica de comparação de caminho utiliza também as conexões entre os nomes de uma ontologia, ou seja, o “caminho” seguido da raiz até a entidade em questão. Essa técnica agrega os nomes desses caminhos e realiza o casamento de *string*s entre essas agregações, utilizando as técnicas de comparação de *string*s já descritas (VALTCHEV, 1999).

2.3.2.4 - Baseadas em linguagem

Essas técnicas se baseiam em processamento de linguagem natural para extração de termos do texto. Esses termos e suas relações são utilizados para medir o grau de similaridade entre duas ontologias pelos nomes e descrições de suas entidades. As técnicas baseadas em linguagem são divididas em duas categorias: métodos intrínsecos (MAYNARD; ANANIADOU, 2001) e métodos extrínsecos. O primeiro consiste em reduzir cada forma de um termo para uma forma padronizada, que pode ser reconhecida com facilidade. O segundo método consiste em utilizar fontes externas para as comparações de termos, possibilitando uma melhor comparação de termos dentro das ontologias no processo de alinhamento. Utilizam-se as seguintes fontes externas:: dicionários, tesauros, dicionários de idiomas, entre outras fontes (MILLER, 1995; FELLBAUM, 1998).

2.3.2.5 - Técnicas baseadas em estruturas

As técnicas de *matching* baseadas em estruturas utilizam as estruturas das entidades para fazer comparação entre ontologias. Essas técnicas se dividem em duas categorias: comparação de estruturas internas e comparação de estruturas relacionais (KALFOGLOU, Y. ; SCHORLEMMER, 2003).

A comparação de estruturas internas leva em conta a modelagem interna das entidades, utilizando suas propriedades internas relações, cardinalidade, atributos, multiplicidade/cardinalidade, transitividade ou simetria das propriedades das entidades para fazer comparação entre entidades de ontologias distintas (CHEN.; TAN; LAMBRIX, 2006). Este método é mais utilizado para diminuição dos candidatos ao alinhamento, do que a análise das correspondências entre duas ontologias em si, pois a quantidade de entidades com estruturas internas comparáveis é grande (EUZENAT; SHVAIKO, 1998, 2005).

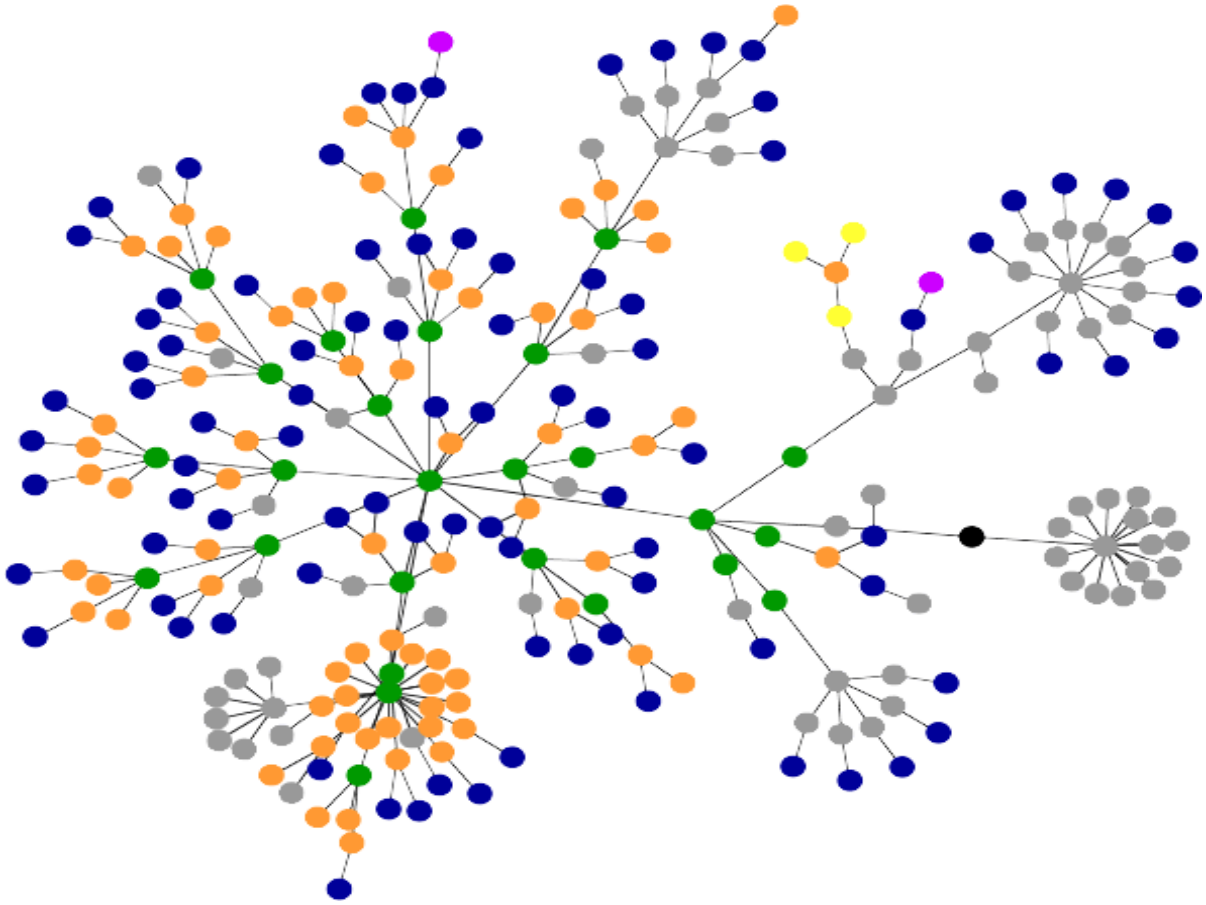
A seguir, apresenta-se pequeno sumário da comparação de cada propriedade levada em conta no alinhamento baseado em estruturas internas.

- Comparação de propriedades e chaves de identificação: o conjunto de propriedade identifica de forma única uma entidade de uma ontologia. Comparando as propriedades de entidades de classes distintas, pode-se identificar se elas modelam a mesma "coisa" do mundo real. A comparação das chaves de identificação também pode ser utilizada para identificação de classes semelhantes (VALTCHEV; EUZENAT, 1997; EUZENAT; SHVAIKO, 1998, 2005), por exemplo, duas ontologias que utilizam RG para identificar pessoas. A constatação dessa chave em duas entidades pode indicar que elas representam a mesma coisa.
- Comparação de tipos de dados: tipos de dados são as formas que um dado pode ser armazenado no computador (inteiro, caractere, numero real, etc.). A comparação entre tipos de dados consiste em analisar os tipos de dados modelados dentro das entidades. Eles podem ser a variação das relações como, por exemplo, entre zero e dez ou a restrição aplicada à certa entidade. A análise objetiva dos tipos de dados pode mostrar a similaridade entre entidades de ontologias distintas. As comparações entre tipos de dados são feitas em relação à proximidade dos tipos: se forem iguais, similaridade alta; próximos, similaridade média e contrários, similaridade mínima (VALTCHEV, 1999; VALTCHEV; EUZENAT, 1997; CHEN: TAN: LAMBRIX., 2006);
- Comparação de domínio: comparação do domínio de uma propriedade como o de uma função, ou seja, utiliza o conjunto de valores possíveis de um atributo para comparação deste entre classes de diferentes ontologias (VALTCHEV, 1999; CHEN; TAN; LAMBRIX, 2006).

- Multiplicidade/cardinalidade: duas multiplicidades são compatíveis se a intercessão de seu intervalo não for vazia, ou seja, compara-se o intervalo de cardinalidade de uma propriedade especifica para analisar a proximidade entre duas entidades de ontologias distintas (LEE et al., 2002; EUZENAT; SHVAIKO, 2005).

Outra forma é a estrutura relacional. Ao invés de se comparar as estruturas internas das entidades, uma das formas de se fazer o alinhamento de ontologias é através da comparação de suas estruturas relacionais. Compara-se as ligações entre as entidades, suas cardinalidade e restrições (NOY; MUSEN, 2003). Nesse caso, a estrutura das ontologias, nas quais quer se aplicar o processo de alinhamento, é transformada em grafos (FIG. 1) e esses são comparados para que se encontrem as semelhanças entre suas arestas e vértices. Compara-se os subgrafos, como pode se observar na FIG. 2, para entrar medidas de semelhanças entre esses e, quanto mais semelhantes, melhor o alinhamento (EUZENAT; SHVAIKO, 2005, 2007).

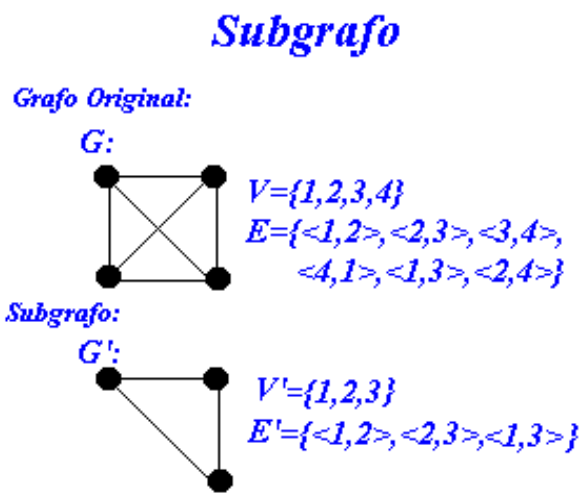
FIGURA 1 - Grafo



Linhas representam ligações
Pontos representam classes
Fonte: MARTINS (2010, *online*)

Para essa técnica quanto maior a similaridade das relações de entidades de ontologias distintas, mais próximas essas entidades são. Contudo, a relação não é direta, pois para a comparação dessa proximidade deve-se levar em conta o tipo de relação sendo comparada. Existem três tipos principais de relação que são levados em conta no alinhamento baseado em estrutura relacional: relações taxonômicas, relações meriológicas e as outras relações envolvidas (EUZENAT et al., 2004; KALFOGLOU; SCHORLEMMER, 2003; CHEN; TAN; LAMBRIX., 2006).

FIGURA 2 - Representação subgrafo



Fonte: ASTUDILLO (1998, *online*)

Relações taxonômicas

Considera-se a estrutura criada a partir das relações subgrafo para a transformação das estruturas das ontologias em grafos. Aqui essas relações são consideradas o eixo central de uma ontologia. O método mais comum para comparação das estruturas baseadas na relação subclassof é a comparação do numero de interligações entre as entidades da ontologia (WANG; LIU; BEL , 2010)

Algumas medidas são levadas em conta dentro das relações taxonômicas, conforme os autores Euzenat et al. (2004); Kalfoglou e Schorlemmer (2003); Chen, Tan e Lambrix (2006).

- Dissimilaridade baseada em distância: considera a distância medida no grafo, da raiz à classe, para a comparação entre duas classes de ontologias distintas. Essa abordagem não necessariamente tem relevância semântica, já que nem sempre duas ontologias têm a mesma estrutura taxonômica.
- Similaridade baseada em hierarquia: leva em conta a comparação das superclasses ou subclasses para comparação de classes/entidades de ontologias distintas dentro do processo de alinhamento. essa abordagem nos diz que se duas classes são similares, suas superclasses e subclasses, provavelmente, são similares também. Essa abordagem deve ser utilizada com cuidado e com outras abordagens de suporte, pois classes não similares podem ser mapeadas

dentro das mesmas superclasses por falta de uma análise mais cuidadosa das entidades. A similaridade entre as superclasses e subclasses também devem ser avaliadas, desta forma também se tem uma necessidade de alinhamento entre essas classes;

- Caminho delimitado de correspondências: consiste em analisar o caminho percorrido até as classes dentro da hierarquia da taxonomia de uma ontologia modelada em forma de grafo. Nesse caminho são analisadas correspondências, relações, termos e as posições dentro desse caminho para verificar os termos e classes semelhantes. Com isso busca-se um melhor alinho levando em conta não somente as classes em si, mas também toda estrutura ate ela.

As técnicas estruturais taxonômicas podem ser usadas com técnicas baseadas em nome e *string*s para melhor avaliação das correspondências entre ontologias em um processo de alinhamento, desta forma obtêm-se um processo e produto com maior qualidade (WANG, LIU; BELL , 2010).

Estrutura Mereológica

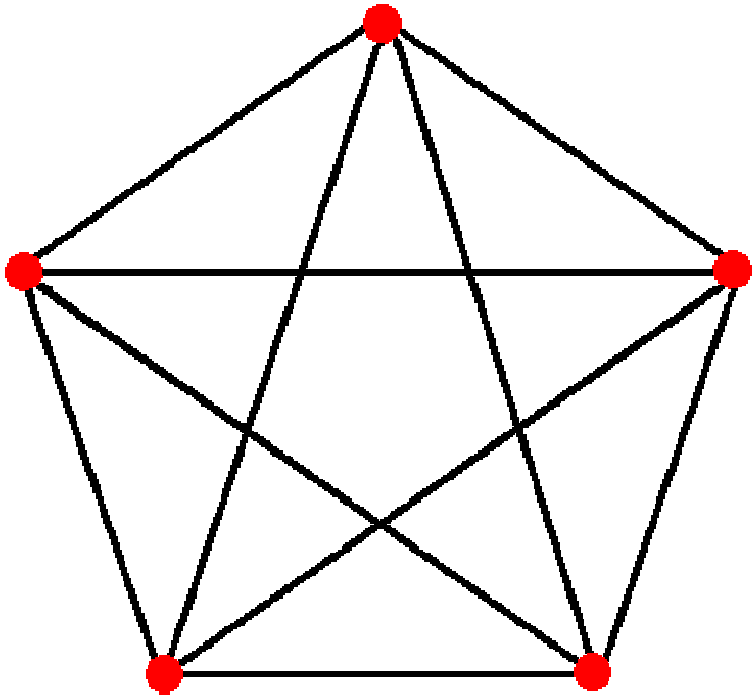
Para se utilizar esse processo de alinhamento é necessário encontrar as propriedades das classes que contem as restrições part-of dentro da ontologia, por exemplo, quando se tem a classe mamífero pode-se encontrar a regra eum. Este tipo de relação pode ser considerada uma relação mereológica, ou seja, uma relação que leva em conta a função entre a parte e o todo (DUVAL, 1988).

Comparando-se as estruturas mereológicas dos modelos de ontologias distintas envolvidas em um processo de alinhamento pode-se averiguar a similaridade das classes sendo avaliadas. Caso as relações sejam similares às classes são consideradas similares.

Relações

Pode-se considerar para o alinhamento as relações entre classe de ontologias, não só as relações *part-of*, mas todas as relações que podem estar contidas dentro das classes de uma ontologia (MAEDCHE; STAAB, 2002). Para esses casos temos a representação das ontologias em grafos, mas diferente de quando se considera a taxonomia, os grafos gerados podem conter circuitos, ou seja, caminhos que começam e terminam no mesmo vértice (que representa a classe) como na FIG. 3.

FIGURA 2 - Circuito em um grafo



Fonte: FIGUEIREDO (2000, *online*)

Esse mesmo raciocínio pode ser generalizado para um conjunto maior de relações, esse conjunto deve ser comparado entre duas ontologias que estão sofrendo o processo de alinhamento, se eles são similares então as ontologias são similares e o casamento das ontologias é feito através dessas comparações (KALFOGLOU; SCHORLEMMER, 2003).

Esse processo leva em conta o conhecimento sobre as relações do engenheiro de ontologias que esta guiando o processo de alinhamento. Para o processamento automático dessa técnica pura problemas de circularidade seriam encontrados, para se resolver isso pode-se casar essa técnicas com as técnicas baseadas em nome e *string* já citadas. E caso as relações estejam organizadas de forma taxonômica podemos utilizar as técnicas estruturais já citadas.

Extensionais

Esta parte foi incluída para efeito de conhecimento, não sendo explorada por não ser aplicada diretamente na pesquisa.

Segundo Palopoli et al. (2000) Essa técnica leva em conta as instâncias produzidas por uma ontologia quando essas estão disponíveis. Se duas ou mais ontologias

têm as mesmas instancias produzidas o processo de alinhamento é facilitado, pois quando estes são iguais a chances de alta similaridade são grandes.

Mesmo que as instancias não tenham exatamente as mesmas características, que são herdadas de suas classes, mas elas pretendem representar a mesma “coisa” do mundo real podem-se utilizar as técnicas para o casamento das ontologias.

2.3.4 - Trabalhos Relacionados e Ferramentas de *Matching*

Após a descrição das técnicas de *matching* e mapeamento, descreveremos algumas aplicações dessas técnicas encontradas em trabalhos relacionados à pesquisa.

Bicer et. al. (2005) traz a descrição do Artemis (*Artemis message Exchange Framework*) que consiste em um projeto internacional de interligação de informações de repositórios biomédicos desenvolveu um produto para o mapeamento e alinhamento de instancias/mensagens de ontologias com objetivo de comunicação entre duas ou mais ontologias de diferentes institutos. O nome dado a esse *framework* é AMEF. Ele permite trabalhar da seguinte forma:

- Mapeamento de uma ontologia base em uma ontologia destino utilizando uma ferramenta de mapeamento, isso produz uma definição de mapeamento. O resultado dessa etapa é utilizado em um processo automático para transformar mensagens em instancias de uma ontologia e vice versa;
- Um mapeamento estrutural é feito entre a estrutura da ontologia base e a ontologia destino, isso gera um mapa de normalização;
- As mensagens entre as bases de dados são transformadas em instâncias das ontologias envolvidas no processo, para isso é utilizado uma normalização de dados. Após isso a definição de mapeamento é utilizada para agregar a nova instancia à ontologia alvo.

Esse processo é utilizado para comunicação entre as bases de dados de duas ou mais instituições de saúde. Utiliza conceitos de mapeamento e alinhamento de ontologias como: *string based*, caminho e mereológicas.

Hélios que é um *framework* desenvolvido para ajudar no compartilhamento de conhecimento em redes P2P baseadas em ontologias, é descrito por Castano et. al (2003) ele é

baseado no projeto Artemis já descrito. O processo de compartilhamento de informação no sistema é baseado em *Peer-ontology*, ontologias que representam os pontos de uma rede P2P cada descrição ontológica tem as informações da base e do alvo, os *peers* (que podem ser considerados os nós de uma representação em grafo) contem a informação necessária para a comunicação entre eles. A comparação é feita a partir de informações contidas na *peer-ontology*, compara-se o conteúdo de um nó com o conteúdo de uma descrição da ontologia para definir onde este deve se encaixar na ontologia alvo.

No que concerne ao alinhamento de ontologias, o framework Helios utiliza como base o sistema já desenvolvido no projeto Artemis, esse sistema se baseia em técnicas de matching já descritas (estrutural, esquemática, etc). O sistema também leva em conta técnicas baseadas em nomes, atributos e estrutura, pois dessa forma consegue alcançar vários níveis de alinhamento entre duas ou mais ontologias que dão suporte a comunicação P2P. Para cada um das técnicas levadas em conta são atribuídas medidas de similaridade que, no caso, são chamadas de coeficientes de afinidade.

IF-Map utiliza técnicas de alinhamento baseado em modelo para criar uma forma automática de alinhamento e mapeamento de ontologias (KALFOGLOU; SCHORLEMMER ,2003). Também utilizando uma série de heurísticas baseadas nas técnicas baseadas em *string* e taxonomia para diminuir os candidatos a mapeamento de ontologias, desta forma busca conseguir melhor performance e menor complexidade.

Ehrig e Sure (2004) Apresentam um método de alinhamento de ontologias que combinam diferentes métodos de alinhamento e cálculos de similaridade. O trabalho visa a melhores resultados dessa forma em relação a outros métodos “inteligentes”. Nesse método é utilizada a combinação das regras de similaridade das seguintes categorias: entidades; semântica; descrição; restrições; regras; vocabulário para aplicações direcionadas; similaridade de caminhos.

O NOM também traz uma série de pequenos suportes para melhorar o resultado da aplicação das regas de similaridade de cada um dos tipos citados: comparação de *labels* através de *string based methods*; utilização dos melhores resultados da comparação de labels; apagar mapeamentos repetidos.

Esses passos são aplicados na comparação de duas entidades para avaliar o quão são próximas e são aplicados da seguinte forma:

- Mapeamento das ontologias a serem alinhadas;
- Tipos básicos de similaridade são levados em conta, sem contar outras características. No trabalho dos autores esses tipos básicos são: label, igualdade de

URIs e a relação é o mesmo Q. Com base nisso a matriz de similaridade é montada;

- Todas as técnicas de similaridades citadas no NOM são utilizadas nos pares de entiade;
- Os dois passos anteriores são repetidos múltiplas vezes para um refinamento no mapeamento;
- Avaliação do alinhamento.

Ehrig e Staab (2004**)** trazem uma abordagem de alinhamento de ontologias chamada QOM (*Quick Ontology Mapping)* que tem como objetivo um mapeamento que, em comparação com outras ferramentas de alinhamento e mapeamento de ontologias, tem melhor performance e baixo custo computacional. Para tal tarefa esse método traz como base o método NOM (já descrito). Também utilizando triplas RDF. Através de heurísticas específicas, o sistema diminui o número de conjunto de entidades candidatas ao alinhamento, depois disso cria-se uma ordem de descartes que aumentam ainda mais a performance do sistema, para isso é utilizado processamento dinâmico. Ele leva em consideração as seguintes táticas para candidatos ao alinhamento:

- Randômica: que simplesmente cerca a quantidade de entidades candidatas ao alinhamento;
- *Label*: Compara as entidades em que os *labels* são próximos;
- Aera: após as duas etapas anteriores as áreas em torno das instancias são comparadas;
- Propagação de mudança: o sistema utiliza o resultado da análise de área para propagar uma análise similar as adjacências das entidades encontradas;
- Hierarquia: a comparação é feita de forma *top down*, iniciando no topo da hierarquia taxonômica e caminhando para as folhas;
- Combinação: utiliza “subagendas” das etapas anteriores para combinação de entidades e otimização da analise.

Com a utilização das estratégias acima o QOM aplica as técnicas de cálculo de similaridade para o alinhamento; quanto maior a similaridade das agendas e entidades mais similares ontologias são.

Glue, que é descrito por Doan et al (2003), é um sistema que aplica métodos baseados em taxonomia através da utilização de aprendizagem de máquina, para produzir um método semiautomático de alinhamento e mapeamento de ontologias. Utiliza duas ontologias,

faz a busca dos conceitos na segunda ontologia e busca a melhor correspondência na primeira. Ele se vale de uma série de bases de aprendizado para aproximar as classificações das duas ontologias avaliadas, ou seja, trabalha com outras bases de comparação para melhor classificar os conceitos de uma ontologia O2 em uma ontologia O1. Após a faze de utilização de aprendizagem de máquina, o sistema utiliza uma tabela baseada no conhecimento dos usuários para refinar o alinhamento de ontologias. Faz a comparação dos labels com as informações dados por usuários que estão contidas em uma tabela de alinhamento.

Mascaradi, Locoro e Rosso (2009) trazem uma abordagem baseada na utilização de ontologias de alto nível (bfo, dolce, sumo, etc.) para o mapeamento e alinhamento de ontologias de forma automática. As comparações e caminhos para alinhamento de ontologias são feitas baseadas em uma das ontologias de alto nível, essas são usadas para o casamento entre os conceitos de duas ontologias com os conceitos superiores delas próprias.

Os algoritmos produzidos nessa abordagem levam em conta correspondências entre conceitos e subclasses apenas, não contemplando nenhuma outra característica da ontologia. Para o alinhamento, eles desenvolveram três algoritmos que são aplicados separadamente:

- *uo_match*: utiliza agregação entre os alinhamentos entre uma ontologia e uma ontologia de alto nível, para isso, utiliza a marcação de todas as uniões de todas as correspondências entre conceitos encontradas, alinhamento paralelo que corresponde a aplicação de métodos de alinhamento baseado em *string* e linguagem para obter os pares de conceitos alinhados e composição entre os alinhamentos entre uma ontologia e uma ontologia de alto nível, ou seja, ele é a composição do resultado de duas ontologias aplicadas a fase anterior ;
- *structural_uo_match*: trabalha da mesma forma que o uo_match, no entanto leva em conta a estrutura das ontologias e das ontologias de alto nível e não somente os conceitos;
- *mixed_match*: mescla as duas fases anteriores para o refinamento do alinhamento entre as ontologias e as ontologias de alto nível.

Felicíssimo e Breitman (2003) apontam uma abordagem relacionada à taxonomia das ontologias, pois em seu trabalho a visão de que a taxonomia de uma ontologia é seu ponto central é a base. Sua estratégia de alinhamento é subdividida em três fases distintas: comparação lexical que faz a comparação dos conceitos de ontologias utilizadas como entradas e os dados apresentados ao sistema como pontos de parada, isso visa buscar conceitos lexicamente iguais aos dados como referencia para poda da estrutura de uma

ontologia; a segunda etapa leva em conta a hierarquia de uma ontologia, para isso, são utilizadas ferramentas já estabelecidas dentro da área de engenharia de software; e a terceira etapa melhora o resultado da segunda etapa através da classificação do resultado das similaridades entre as ontologias em bem equivalentes e pouco equivalentes. Isso é feito através de parâmetros pré-estabelecidos de equivalência.

Atom

Segundo Raunich, e Rahm, (2011) esta ferramenta utiliza uma abordagem chamada fusão de taxonomias, que visa não utilizar especialistas humanos na função de tradução dos dados para a fusão, desta forma ele funde uma ontologia origem em uma ontologia alvo (*TargetDriven*). Utilizando a fusão assimétrica ele pode aumentar a ontologia alvo utilizando fontes externas, isso na prática poderia ter vantagens diversas dentro da websemantica. O sistema utiliza como entrada duas taxonomias e um mapa de equivalência entre os conceitos. Utiliza algoritmos que criam uma taxonomia que preserva os conceitos das taxonomias de entrada e a estrutura da taxonomia alvo.

COMA

O sistema COMA é uma plataforma que combina múltiplos "termos correspondentes" de forma flexível. Provendo/provedor de uma grande gama de "termos" individuais, em particular, uma nova estratégia que objetiva reutilização de dos resultados de operações/procedimentos anteriores, e diversos mecanismos para combinar os resultados das execuções sobre "termos". Tem-se COMA como um framework com compreensível avaliação da eficácia dos diferentes "termos" e suas combinações em "termos reais". Os resultados já obtidos mostram a superioridade/quão superior da/é a abordagem com combinações de "termos" e aponta/indica uma valorização para a reutilização como estratégia (AUMUELLER; DO; MASSMANN; RAHM, 2005).

Coma++

Utiliza uma base de informação contendo tipos de alinhamento pré-determinados que são escolhidos pelo usuário de acordo com suas necessidades de *Matching*, ele também permite ao usuário inserir novas técnicas dentro da base de informações sobre *Matching*, desta forma alem de utilizar a base pré estabelecida o usuário pode contar com novas abordagens que são necessárias dentro do escopo de alinhamento no qual estão trabalhando, além de permitir a combinação das várias técnicas contidas no sistema para trabalhar o alinhamento (AUMUELLER, D.; DO, H. ; MASSMANN, S. e RAHM , E. , 2005).

O sistema também permite criar *workflows* para criar possibilidade de alinhamento fracionado, vários passos dentro de um mesmo alinhamento entre duas ou mais ontologias. Para sua utilização conta com uma interface gráfica para a operação das tarefas as quais se propõe, desta forma busca melhor atender aos usuários que o utilizam.

O sistema tem suporte aos formatos, XML *Schema, Ontology Web Language* (OWL), XML *Data Reduced* (XDR) e esquemas relacionais.

Prompt* e *PromptDiff

PROMPT lhe permite criar um *diff* estrutural de duas ontologias (este componente de PROMPT é chamado PromptDiff). Ele apresenta duas visões no *diff*. Na primeira visualização (visualização de árvore), uma árvore de classes é feita para a primeira ontologia com frames novos mostrados sublinhados, frames deletados mostrados riscados e frames movidos ou modificados mostrados em diferentes cores e fontes. As "*tooltips*" apresentam informações adicionais a respeito da mudança. Quando se seleciona uma classe, não vê somente o formulário de classe do lado direito, mas também uma tabela comparando o frame atual à antiga versão. Você pode aceitar ou rejeitar as mudanças em vários níveis de granularidade. Em uma segunda visualização, (visualização de tabela), uma lista de frames das duas versões pode ser vistas lado a lado, com informações adicionais nas outras colunas e num diff individual (NOY; MUSEN, 2000).

Além do supracitado o PROMPT também funciona como uma ferramenta de fusão de ontologias. Durante o processo de fusão ele mostra divergências entre ontologias faz sugestões de como resolver tais divergências alem de ajudar na solução de problemas de *Matching* através de sugestões de estratégias a serem seguidas. As sugestões são baseadas em Heterogeneidade sintática levando em conta as diferenças entre os nomes das entidades. Depois da fusão o sistema verifica os problemas estruturais gerados e propõe soluções baseadas na resolução de problemas relacionados a heterogeneidade estrutural (NOY; MUSEN, 2004).

Anchor Prompt

Sistema produzido para dar suporte a sistemas já consolidados como PROMPT e Chimaera introduzindo no processo de fusão novos pontos para avaliação de similaridade entre ontologias (NOY; MUSEN, 2001).

Para ajudar no processo o sistema utiliza como entrada um conjunto de pares de itens inter-relacionados retirados de uma ontologia, essa entrada pode ser gerada automaticamente ou escolhida pelo usuário. A partir da entrada determinada o sistema gera

um novo conjunto de entidades relacionadas através de algoritmos de similaridade (AUMUELLER; DO; MASSMANN; RAHM, 2005).

***Onion*[2]**

O sistema ONION (*Ontology Composition*) foi desenvolvido por Mitra, Wiederhold e alguns colegas no grupo de bancos de dados da Universidade de Stanford. O projeto ONION propõe uma abordagem escalável e de fácil manutenção baseada na interoperação das ontologias. A motivação para o trabalho é manipular consultas distribuídas cruzando os limites de sistemas de informação subjacentes.

O sistema utiliza uma abordagem que leva em conta algebra para sobreposição de conhecimento desta forma faz uso de operadores algébricos tais como: união interseção e diferença. As dependências entre ontologias são expressas através de expressões algébricas que geram como resultado um grafo.

SMART

Sistema utiliza um algoritmo que fornece uma abordagem semi-automática para fusão e alinhamento de ontologias. A SMART auxilia o desenvolvedor da ontologia através da realização de determinadas tarefas de forma automática e guiando o desenvolvedor para outras tarefas para as quais a sua intervenção é necessária. A SMART também determina possíveis inconsistências no estado da ontologia que podem resultar de ações do usuário, e sugere maneiras para corrigir estas incoerências. É definido o conjunto de operações básicas que são executadas durante a fusão e alinhamento de ontologias, e determinado os efeitos que a invocação de cada uma dessas operações tem sobre o processo. SMART é baseado em um modelo de conhecimento extremamente geral e, portanto, pode ser aplicado em diferentes plataformas (NOY;MUSEN, 1999).

2.4 - Implicações para a pesquisa

O capítulo anterior buscou explorar algumas técnicas de avaliação e alinhamento de ontologias descritas na literatura. Através da pesquisa de dados chegamos aos padrões de avaliação e alinhamento citados. Não existiu a pretensão de se descrever a totalidade das

[2] ONION. Disponível em:< http://ontolog.cim3.net/cgi-bin/wiki.pl?ONION>. Acesso em: 27 abril de 2010.

iniciativas do campo, mas sim subsidiar este trabalho, exemplificando técnicas e tipos existentes dentro dos quatro padrões abordados de avaliação e alinhamento.

É importante frisar que cada uma das técnicas e tipos citados, dentro dos quatro padrões de avaliação e alinhamento, visam garantir a qualidade dos dados contidos em uma ontologia de alguma forma, mesmo tendo características específicas diferentes. Desta forma, pode-se inferir a necessidade de suporte adequado à análise da qualidade destes dados para que o processo de avaliação e alinhamento seja mais efetivo.

A análise de cada uma das técnicas apresentadas auxilia na busca por respostas para questões importantes para o andamento da pesquisa. Mostra-se, assim, a real a importância de se pesquisar uma forma em que as técnicas de visualização possam auxiliar a avaliação e alinhamento, já que entre as formas de representação de conhecimento pode-se considerar a visualização de informação como facilitador de compreensão dos dados e de sua organização. Desse ponto sugiram as seguintes questões inerentes a essa pesquisa:

- Como garantir a confiabilidade dos dados analisados através da utilização da representação gráficas destes? Tendo como partida, que a visualização tem como objetivo facilitar a compreensão e o trabalho de organização dos dados.
- Como as diferentes abordagens de avaliação podem contribuir para a construção do protótipo proposto? Essa questão se deve ao fato que, através da análise feita das abordagens pesquisadas, tem se percebido a demanda por suporte as técnicas de avaliação e alinhamento, o que pode ser obtido através de técnicas de visualização de informação. Também percebemos a necessidade da criação de um roteiro para nortear métodos de avaliação que serão executados.
- Como garantir a modelagem correta de um domínio não só em termos semânticos, mas também em termos hierárquicos? Nesse ponto temos que considerar como o alinhamento somado às técnicas de visualização pode tornar um roteiro/método de avaliação de ontologias mais eficaz, como podemos casar as duas constantes (visualização + alinhamento) no suporte a correta construção dos modelos abstratos. E, como eles podem auxiliar no melhor caminho de construção desde o início da modelagem.
- Como já citado, ontologias podem ser construídas para o auxilio à sistemas, não só em sua modelagem, mas também na comunicação entre sistemas distintos. Esse ponto nos leva, também, constatação de que a pesquisa na área de visualização e alinhamento nos permite verificar a interoperabilidade de

ontologias que visam a comunicação de sistemas, através de uma técnica de avaliação centrada na verificação da compatibilidade de modelos criados para dois sistemas distintos.

- Como encontrar discrepâncias entre as modelagens de ontologias novas e suas bases teóricas e de domínio? Através do alinhamento e de técnicas de avaliação voltadas para especialistas em domínio e engenheiros de ontologias uma melhor modelagem pode ser alcançada.

Junto a isso, quando consideramos as técnicas de alinhamento de ontologias, aliados aos métodos de visualização, como auxilio aos métodos de avaliação dos processos de construção de ontologias podemos obter melhores resultados desse e, consequentemente, a possibilidade de se alcançar melhores produtos finais. O conhecimento das técnicas de alinhamento, levando em conta a necessidade de melhores modelagens, guia a pesquisa no sentido de mostrar como podemos concatenar dois tipos de processos dentro da construção de ontologias. Podendo auxiliar no entendimento deles, além de modelagem e da utilização de ontologias.

Também se pode considerar os tipos e técnicas de alinhamento como suporte aos métodos avaliativos, pois esses tem como objetivos verificar não só a interoperabilidade entre ontologias, mas também auxiliar na verificação de erros de modelagem estruturais e em relação aos domínios que são base para a ontologia.

Na avaliação e alinhamento de ontologias encontramos métodos que auxiliaram na construção do protótipo, que é suporte para o roteiro proposto. O protótipo e o roteiro podem ser aplicados a grande variedade de métodos avaliativos e de alinhamento. A visualização de informação pode ser usada em outras técnicas avaliativas mesmo que isso ainda não tenha sido praticado.

3 - Visualização de Informação

Dados são resultados de observação ou de características de objetos que podem ser recolhidos e definidos de forma direta. Dados podem ser armazenados. É possível pensar na geração de significado para um corpus, a partir de um conjunto de dados interpretados por uma pessoa.

> […] dados são puramente sintáticos enquanto informação contém, necessariamente, semântica. Conhecimento é uma abstração interior […] relacionada a alguma coisa existente no mundo real e do qual temos uma experiência direta (SETZER, 1999, *on line*).

Dados fora do contexto de significação não possuem capacidade de representar algo. A conceitualização permite o entendimento prático para informação. Percebem-se assim, facilmente, dificuldades para usuários em face da complexidade do processo de consulta, aquisição e utilização de dados.

Métodos de modelagem de dados surgiram com o intuito organizar dados de forma que façam sentido em um contexto. Tais modelos vêm sendo desenvolvidos junto às técnicas de difusão e de processamento dos dados. Esses métodos são suportados por linguagens específicas para o processamento via máquinas (computadores, assistentes, celulares, etc.). As linguagens correspondem a uma série de estruturas textuais nem sempre próximas às linguagens naturais. Assim, fora de um círculo específico de especialistas, essas linguagens não são facilmente interpretadas.

Nesse contexto, a Ciência da Informação tem como um dos seus objetivos aproximar o domínio ou autor do conhecimento, o profissional da informação e o usuário final, além de auxiliar na disseminação e compreensão da informação:

> Ciência da Informação é a disciplina que investiga as propriedades e o comportamento da informação, as forças que regem seu fluxo e os métodos para processá-la, a fim de obter acessibilidade e utilização ótima. Esta interessada num conjunto de conhecimentos relacionados com a origem, coleção, organização, armazenamento, recuperação, interpretação, transmissão, transformação e utilização da informação (BORKO, 1968).

Há a necessidade de interpretar o que é desenvolvido para máquinas de forma que seja palatável aos olhos humanos. Uma alternativa é o uso de técnicas visualização de informação, ou seja, o conjunto de métodos e ferramentas que transformam linguagens estruturadas e semiestruturadas em impressões gráficas na tela de um computador. Com isso visa-se facilitar

o acesso ao conteúdo de ontologias, mas pode-se considerar sua utilização em outros processos informacionais. O grupo MHTX[3] de pesquisa, por exemplo, tem utilizações de formas de visualização de informação no tratamento de hipertextos.

> [...] tendo em vista que toda técnica de visualização é destinada a apoiar a realização de tarefas de análise de dados, é abordada a questão de caracterização dessas tarefas e dos mecanismos de interação necessários para suportá-las (LUZZARDI, 2002).

No restante da presente seção, apresentam-se tipos de visualização em relação à dimensionalidade dos dados. Na seção 3.1, descrevem-se técnicas de visualização de ontologias na seção 3.2 e por fim apresentam-se algumas implicações desta revisão de literatura para pesquisa nana seção 3.3.

3.1 - Tipos de Visualização

Sugerida a necessidade de uso de técnicas visualização, esta seção se destina a descrição dos tipos e de suas respectivas técnicas de visualização. Descrevem-se os tipos de visualização mais comuns, como: i) unidimensional, ii) bidimensional; iii) tridimensional; iv) multidimensional (que se utiliza dos três primeiros para sua efetivação); e v) visualização hierárquica. Essa ordem de apresentação permite o entendimento de como um tipo utiliza o outro para a continuidade e melhoria das formas de representar dados e informação.

3.1.1 - Unidimensionais

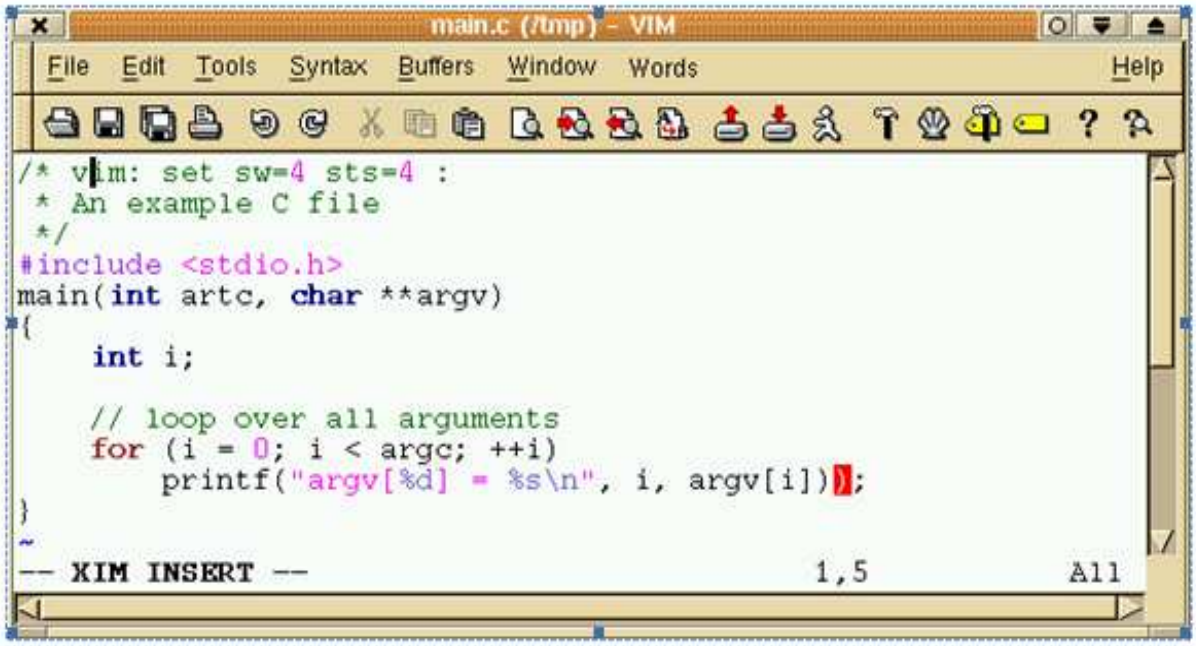
Correspondem a listas de símbolos lineares: códigos, textos ou listas de palavras (NASCIMENTO; FERREIRA, 2005). Como são sequenciais, cada linha contem uma parte do conjunto de dados. Visualizadores de texto são os mais utilizados para esse tipo de visualização, pois é necessário ter acesso a fontes, cores, letras (KINER; CALONEGO; BUK, 2004).

[3] PROJETO de pesquisa MHTX. Coordenação de Gercina Ângela Borém Lima. Belo Horizonte, 2012. Disponível em: <http://www.gercinalima.com/mhtx/pages/apresentacao.php>. Acesso em: 20 set 2011.

Os sistemas que dão suporte a este tipo de visualização necessitam de recursos na tela, como barra de rolagens, para que se possa analisar todo o corpus de dados visualizado. Tais sistemas permitem modificações, como por exemplo: mudança de cor, de tamanho, de formatação do texto pesquisado, dentre outras. O que permitiria facilidade no tratamento dos dados em si – comparação, verificação de erros (CARVALHO; MARCO, 2009) – acaba por gerar problemas na busca dos dados pelo usuário (KINER; CALONEGO; BUK, 2004). Como exemplos de visualização unidimensional apresentam-se o *Vim* e o *xdiff*.

O **Vim** é um editor utilizado para criação de códigos de *software* baseado no editor vi[4], que tem capacidade de editar mais de um arquivo de uma única vez. Proporciona formas de visualização e de alteração das informações contidas em textos, mas exige treinamento sobre os comandos, conforme FIG. 4. É bastante utilizado por desenvolvedores de software sendo conhecido como “editor de programadores”. Além disso, é um sistema de código aberto.

FIGURA 3 - Editor Vim



Fonte: VIM... (2010, *online*).

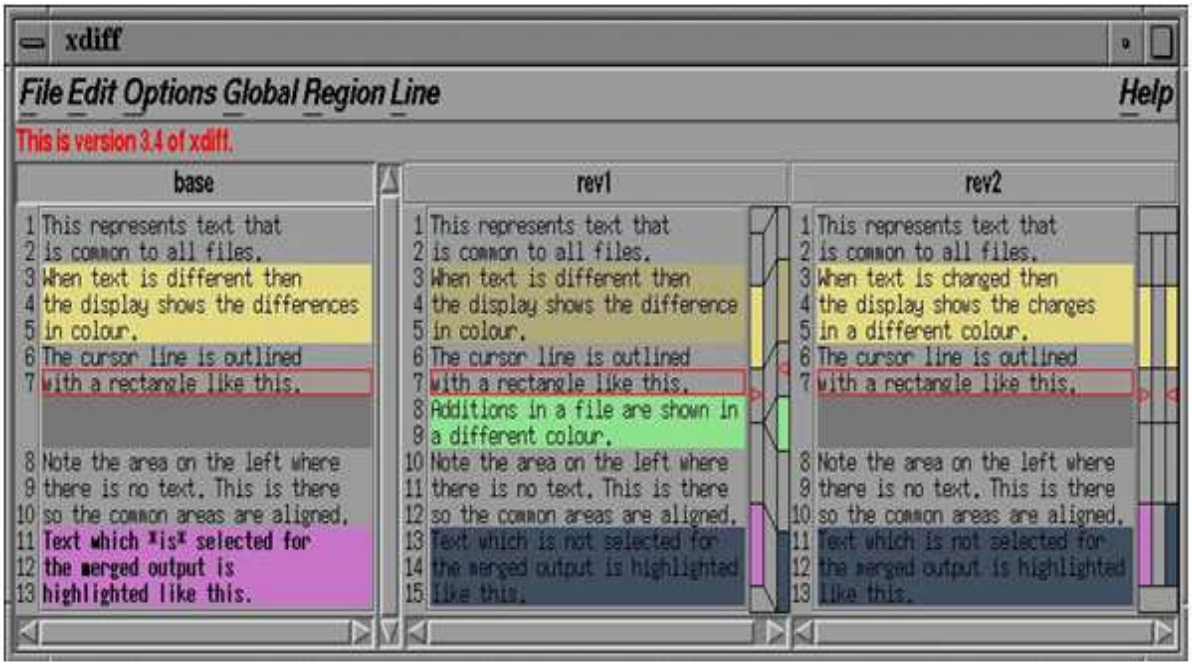
A ferramenta proporciona a diferenciação do texto através de cores, podendo assim, para vários tipos de linguagem de programação, estabelecer um padrão de cores compatível com sua utilização. Também permite a diferenciação de textos, ou seja, tem a capacidade de comparar dois textos e verificar o que há de diferente entre os dois.

[4] VIM é um editor de texto do sistema operacional Unix e semelhantes.

O **Xdiff** é baseado em uma ferramenta chamada *diff* que compara diferenças entre arquivos e permite mesclá-los ou transformar um arquivo numa versão mais nova alterada.

> xdiff é um utilitário gráfico para ver as diferenças de um arquivo e até quatro revisões do arquivo. O texto do arquivo é apresentado lado a lado com suas diferenças alinhados e em destaque em cores diferentes para fácil identificação. Xdiff pode ser usado para produzir uma versão mesclada incorporando qualquer combinação das diferenças. Funcionalidades de edição simples também são fornecidas (XDIFF, 2003, *online*).

FIGURA 4 - Xdiff



Fonte: XDIFF (2003, *online*).

2.1.1 Bidimensionais

Este tipo de visualização engloba a tradução de dados bidimensionais para um tipo de metáfora visual. Como exemplos de visualizações podem-se citar: mapas, jornais, sistemas de informação geográficas, dentre outros (KINER; CALONEGO; BUK, 2004; CARVALHO; MARCO, 2009; LUZZARDI, 2003).

Segundo Carvalho e Marco (2009), dados bidimensionais são compostos por atributos utilizados diretamente na sua representação. Como exemplo, citam-se dados de localização em mapas, altura e largura, ou qualquer outro conjunto de dados duais como esquerda e direita, acima e abaixo. Dados bidimensionais são dados obtidos de pares de variáveis.

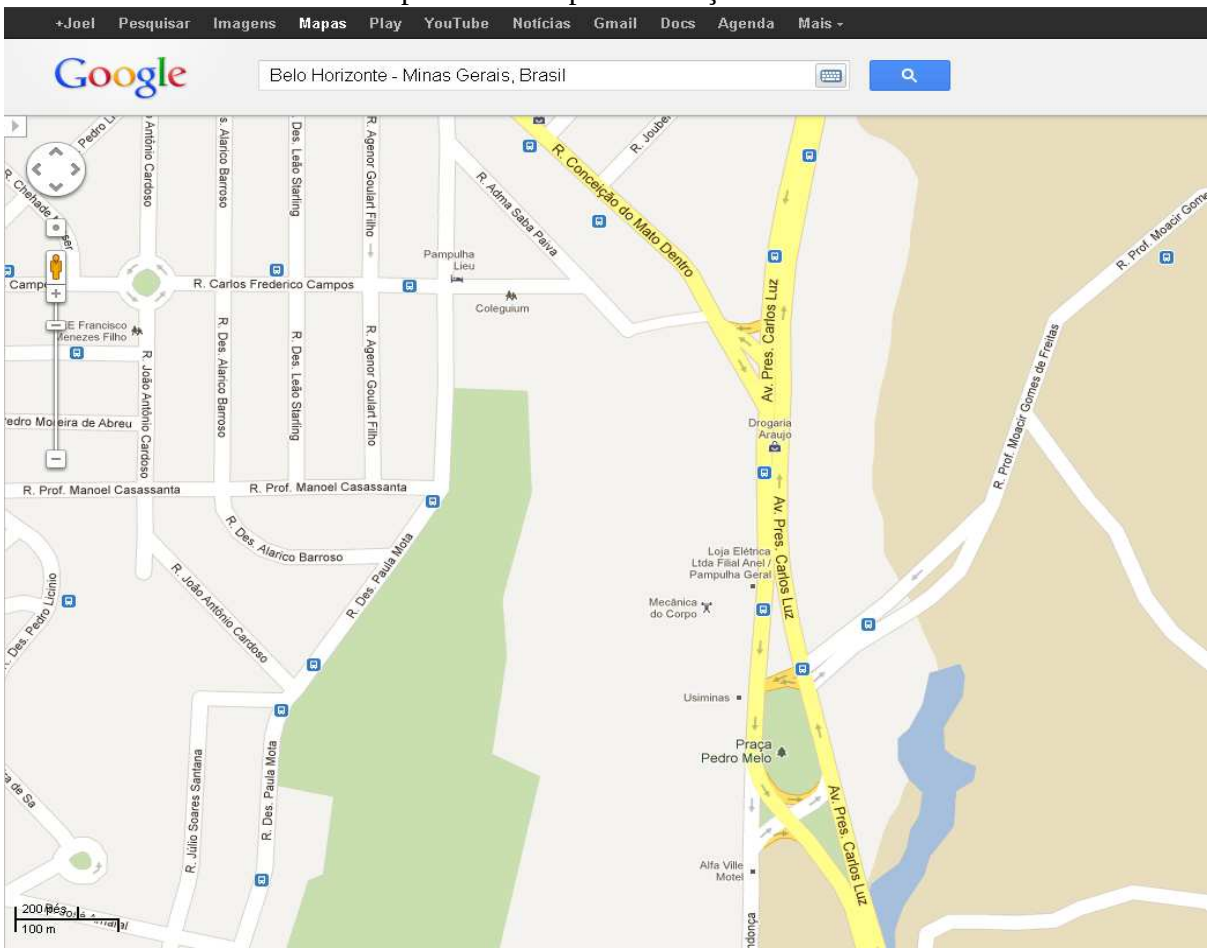
Utilizam-se os tipos de dados para a representação de um ambiente bidimensional (NASCIMENTO; FERREIRA, 2005; CARVALHO; MARCO, 2009), pois como as visualizações na tela do computador são em duas dimensões (2D), é necessário diferenciar entre uma visualização que realmente se enquadra dentro da caracterização de uma metáfora visual bidimensional (KINER; CALONEGO; BUK, 2004; CARVALHO; MARCO, 2009). Também são utilizados para criação de gráficos (tipo linha, pizza, etc), para estabelecer localização em mapas, para prover noções de localização na tela (direita e esquerda, alto e baixo), dentre outros (LUZZARDI, 2003).

Segundo Kiner (2004), apesar de existir classificação de dados em ambientes de visualização bidimensionais, os usuários enfrentam problemas para perceber outros atributos foram das características bidimensionais, como por exemplo: subconjuntos, dados hierarquicamente inferiores, dados adjacentes, dentre outros. Um exemplo popular dentre os sistemas que são exemplos de visualização 2D é o *Google Maps*.

Utilizando o *GoogleMaps*, um usuário é capaz de identificar posições no mapa, endereços, e outras informações específicas de cada endereço[5]. O sistema utiliza uma abordagem bidimensional de visualização, agregando outras informações aos dados representados no mapa, como por exemplo, um sistema de informação geográfico.

[5] GOOGLE MAPS. Desenvolvido por GOOGLE. 2010. Disponível : < http://maps.google.com/support/bin/answer.py?hl=br&answer=7060>. Acesso em 22 out 2010

FIGURA 5 - Exemplo de busca por endereço no *site* GOOGLEMAPS



Fonte: GOOGLE (2010, *online*).

Além dos recursos de visualização, a ferramenta engloba várias funcionalidades que respondem a pesquisa, tudo em um único local integrado. Dentre elas estão a busca por endereços e informações de contato, os mapas que podem ser arrastados para visualizar regiões limítrofes instantaneamente, as rotas detalhadas (GOOGLE, 2010, *online*).

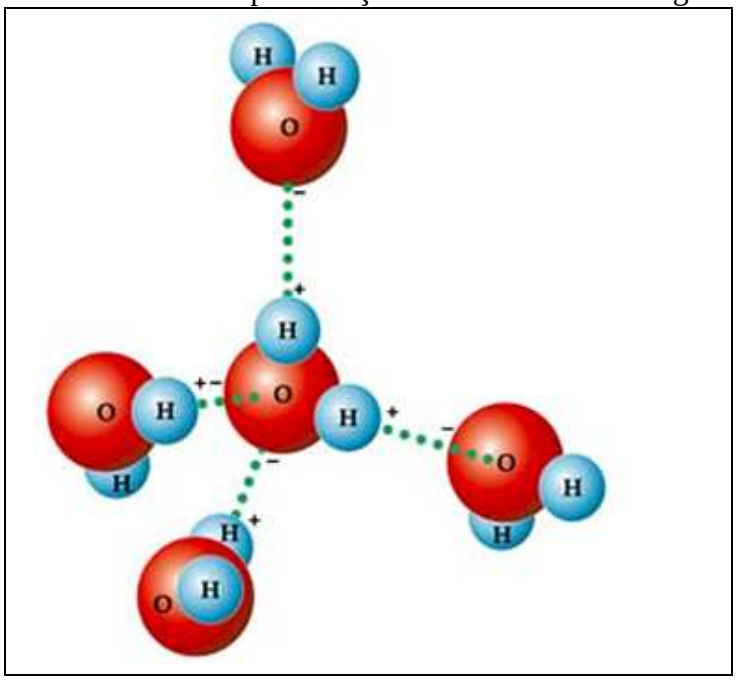
2.1.2 – Tridimensionais

Uma visualização tridimensional é representada utilizando as três dimensões: altura, largura e profundidade. É utilizada para representar dados que representam objetos reais como uma bola, um prédio ou uma molécula de água, desde que tal objeto possua

características tridimensionais (KINER; CALONEGO; BUK, 2004), ou seja, a possibilidade de visualização de exterior e interior.

Este tipo de metáfora visual também é utilizado para representar dados que tenham possibilidade de representações em três dimensões (3D) além das possibilidades de combinações em visualizações 2D (NASCIMENTO; FERREIRA, 2005). Exemplos dessas representações são arvores hierárquicas e representações que tenham ligações entre si (itens ou características), não necessariamente parte do mundo real (CARVALHO; MARCO, 2009). Uma ontologia, por exemplo, mesmo que represente algo real, não existe fisicamente e mesmo assim suas relações entre classes, instâncias e atributos podem ser representadas tridimensionalmente.

FIGURA 6 - Representação 3D da molécula de água



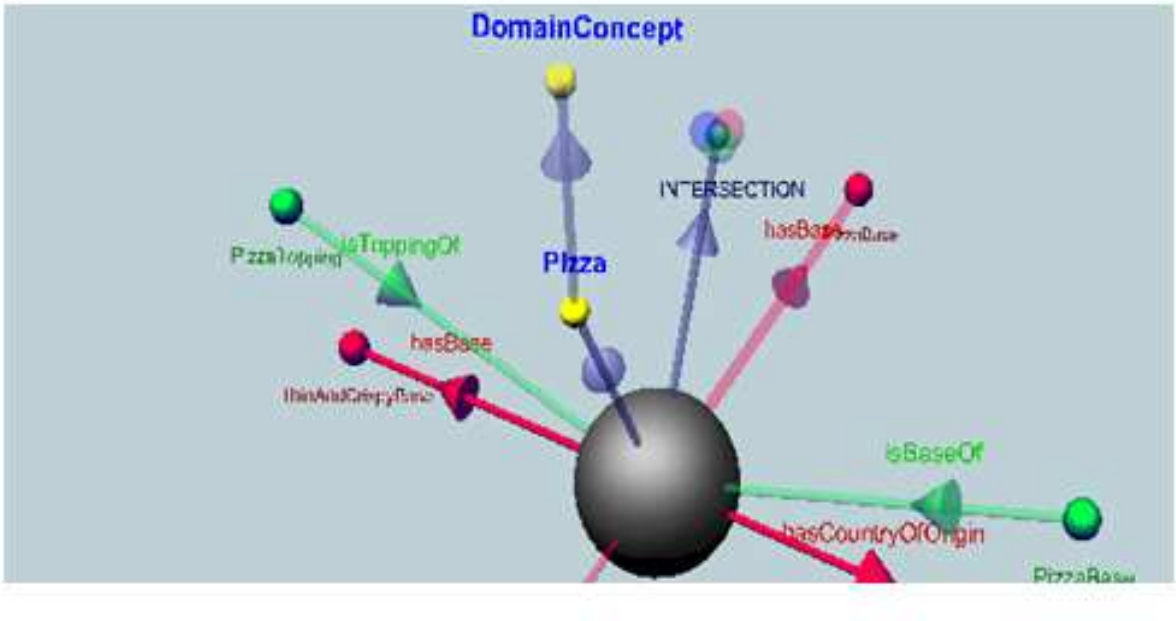
Fonte: BRASILESCOLA (2010, *online*).

Neste tipo de representação, o usuário navega e explora a imagem 3D como se estivesse interagindo com um objeto real. Além disso, existe a possibilidade de interação com o "espaço" no qual a imagem esta inserida. Existem também as representações 3D de objetos inicialmente concebidos para visualização bidimensional, como é o caso de um gráfico em formato de disco, que pode ser transposto para uma visualização tridimensional.

Um exemplo de visualização tridimensional é *OntoSphere (*KATIFORI et al., 2007*)*, que utiliza uma interface 3D para representação de dados. O *OntoSphere3D* é uma ferramenta para visualização de ontologias que faz uso de um espaço tridimensional,

onde a informação é destacada por estímulos visuais, como cor ou tamanho das entidades. A ferramenta transformar questões da ontologia em modelos visuais, através da adoção de um mecanismo dinâmico de expansão da imagem e da possibilidade da visão de pontos de vista diferentes, em diferentes granularidades. Dessa forma, proporciona navegabilidade ao modelo representado (FIG. 5).

FIGURA 7 - Exemplo de Representação OntoSphere



Fonte: ONTOSPHERE3D (2010, *online*).

2.1.3 – Multidimensionais

Para Carvalho e Marcos (2009), os tipos de visualizações descritas nas seções anteriores (unidimensional, bidimensional e tridimensional) podem ser consideradas subconjuntos da visualização de nDimensões.

Segundo Keim e Krigel (1994), um dos grandes desafios quanto a dados multidimensionais é se estabelecer e encontrar caminhos adequados para a visualização de dados multidimensionais para suporte à análise destes pelo usuário. Pohlheim (1999) diz que com a limitação visual humana de três dimensões as técnicas de visualização permanecem dentro de duas ou três dimensões e para vencer essa limitação é interessante que se “reduza” a dimensionalidade dos dados para 2D ou 3D para que esses dados possam ser representados.

Pillat (2006) coloca que visualização multidimensional visa representar em uma metáfora visual de tipos de dados multifacetados, ou seja, cada um dos dados é composto por vários atributos. A autora também expõe que podem existir categorizações dependendo de certos critérios, alguns deles sugeridos por Keim (1996): natureza dos dados; abordagem de mapeamento e métodos de interação e distorção disponibilizados pela ferramenta.

Algumas técnicas de visualização multidimensional são citadas por Keim (1996), projeções geométricas, iconográficas e orientadas a pixels.

Segue a breve descrição de cada uma das técnicas citadas:

- Projeções geométricas - dentro das técnicas de projeção geométricas são citadas por Valiati (2008): Coordenadas paralelas (INSELBERG; DIMSDALE,1990) que não utilizam eixos cartesianos ortogonais para representação dos dados, Visualização de coordenadas radiais que representam as “n dimensões em n linhas que emanam do centro de um circulo e terminam no seu perímetro” (Valiati, 2008, p.25) e Coordenadas Paralelas Circulares que se trata de uma versão de coordenadas paralelas (VALIATI, 2008).
- Iconográficas - existem tipos diferentes de visualização iconográfica, uma se valendo de representações de faces (CHERNOFF,1973) e a outra representação em glifos que segundo Valiati (2008, p. 30) “as dimensões são representadas como raios de ângulos iguais partindo do centro de um circulo”.
- Orientada a pixels - mapeia cada um dos atributos (seus valores) em pixels na tela (KEIM, 1996).

Os autores Carvalho e Marcos (2009) citam a Tomografia Computacional como um exemplo de visualização de dados multidimensionais, pois nesta aplicação todos os atributos, espaciais e os abstratos, são necessários ao mesmo tempo.

FIGURA 8 - Tomografia Computacional da cabeça humana

Fonte: TOMOGRAFIA... (2010, *online*).

A representação de dados multidimensionais, ou seja, representação de dados através de técnicas específicas de visualização, podem se valer da redução da dimensionalidade destes, por exemplo. Como outro exemplo temos bancos de dados estatísticos se comportam desta maneira, representando espaços tridimensionais a partir de múltiplos pontos de vista (KINER; CALONEGO; BUK, 2004)

3.1.5 - Hierárquicas

Os tipos visualização hierárquica particionam as múltiplas dimensões dos dados em um formato hierárquico de subespaços (GRÉGIO; FILHO; MONTES, 2009), normalmente em algum tipo de árvore, onde os dados são divididos entre nós e folhas que são interligados por linhas. A estrutura é representada de forma hierárquica no espaço, denotando "superioridade" dos nós em relação às folhas e de nós-pais em relação aos nós-filhos (CARVALHO; MARCO, 2009; LUZZARDI, 2003). Também se utilizam figuras geométricas, como cubos e paralelepípedos. Para representar nós e folhas da estrutura (LUZZARDI, 2003).

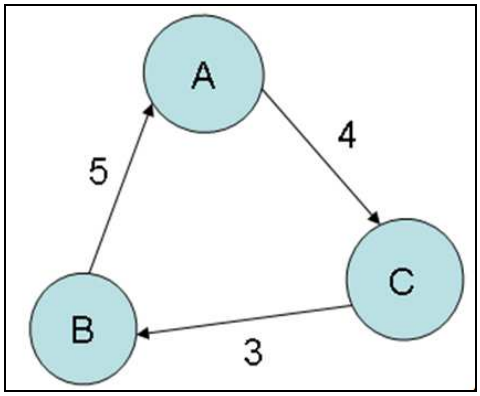
Este tipo de visualização busca representar dados hierárquicos, como taxonomias, estruturas de arquivos, classificação de doenças. Cada um dos dados, com exceção da “raiz”, tem uma ligação com o dado-pai e com os dados-filhos, de forma que a própria estrutura dos dados é modelada como uma árvore.

Várias técnicas existem para este tipo de visualização, variando de uma árvore simples ao tipo “árvore cone” (*Cone Tree*), que apresenta os dados em 3D. Pela grande quantidade de formas e técnicas de visualização Hierárquica existentes, descrevem-se aqui, a título de ilustração, quatro técnicas: arvores, arvore cônica *e árvore hiperbólica*. Esses exemplos são importantes para os objetivos do presente trabalho.

Para que alguns conceitos de desenvolvimento das ferramentas e técnicas de visualização sejam bem compreendidos sugere-se conhecimento básico sobre teoria de grafos. Alguns fundamentos do assunto são apresentados na sequencia.

Para Townsend (1987), um grafo é uma representação matemática de um conjunto de um ou mais vértices, também chamados de nós, que podem se relacionar através de arestas. Um grafo (V,A) é definido pelo par de conjuntos V, onde V significa vértice e A significa aresta (GROSS; YELLEN, 2004). Assim, V significa um conjunto finito de dados e A um multiconjunto finito de arestas, sendo cada aresta composta por dois vértices. Os vértices representam os dados e as arestas representam as ligações entre os vértices, uma vez que uma aresta tem necessariamente dois vértices.

FIGURA 9 - Grafo



Fonte: EDEN (2010, *online*).

Um grafo, em geral, é caracterizado por:

- Ordem: número de vértices de um grafo;
- Adjacência: em um grafo, dois vértices são adjacentes se existe uma aresta que os liga diretamente. No caso de existir uma direção explícita na aresta (como

na FIG. 6), tem-se um grafo dirigido. O direcionamento indica hierarquia, ou seja, o vértice apontado pela seta da aresta é o sucessor e o vértice de onde sai a aresta o antecessor;

- Grau: número de arestas de um vértice;
- Laço: Quando uma aresta liga um vértice a ele mesmo;
- Cadeia: qualquer conjunto de arestas que ligam dois vértices;
- Ciclo: um grafo interligado completamente, ou seja, onde o vértice inicial é o mesmo do final;
- Grafo conexo: um grafo em que um conjunto de arestas que liga todos os vértices (caso contrario seria chamado grafo desconexo).

Uma explicação detalhada da teoria dos grafos não faz parte do escopo do trabalho. Informações adicionais estão disponíveis em Townsend (1987). A seguir são apresentados os tipos mais comuns de grafos.

Árvores

Árvores são estruturas de dados formadas por nós interligados hierarquicamente. Esses nós são representações de um número finito de dados. O primeiro nó é denominado “raiz” seguido de “filhos”, o quais também podem ter filhos. Nós sem filhos são chamados de “folhas”. Com essa nomenclatura, modela-se um conjunto de dados.

Algumas características das árvores são destacadas por Wetherell e Shannon (1996)

- São grafos planares, ou seja, os vértices não estão interligados;
- Existem regras para as distancias entre os nós: nenhum nó deve estar mais próximo da raiz que seus antecessores;
- Em árvores balanceadas, os nós filhos devem ser equidistantes;
- Os nós de um mesmo nível em uma árvore estão alinhados, exatamente na mesma altura;
- o nível de um nó é a distancia desse nó ate o nó raiz.

Árvore simples

Árvores simples são utilizadas para representação de hierárquicas em sistemas de arquivo, dados na web, organização de gráfico, dentre outros (STOLTE; TANG,; HANRAHAN, 2008).

Wetherell e Shannon, (1996) abordaram as dificuldades de projeto de uma árvore simples, uma vez que suas próprias características geram limites. O espaço utilizado para a

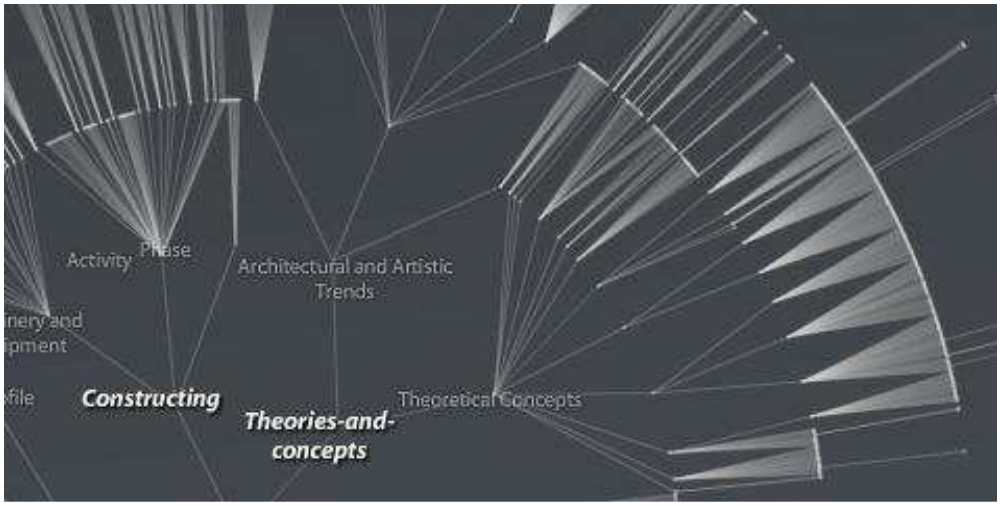
visualização é determinante e pode se impossível representar grande quantidade de dados devido à limitação de espaço.

Árvores Radiais

Em árvores radiais, o nó raiz é localizado no centro da figura e os outros nós são organizados em níveis circulares em volta do ponto central. Quanto mais distante estiver um nó do centro, menor é a sua representação. Esse tipo de árvore permite a visualização de grande quantidade de dados.

Muitos tipos de dados organizados em hierarquias, como vocabulários controlados ou estruturas de diretórios de arquivos, são muito amplos para representação em uma tela de computador. Árvores radiais podem representar conjuntos bem maiores do que as árvores simples são capazes, pois os usuários se concentram em partes específicas da árvore, sem perder o contexto. (STOLTE; TANG; HANRAHAN, 2008)

FIGURA 10 - Árvore radial



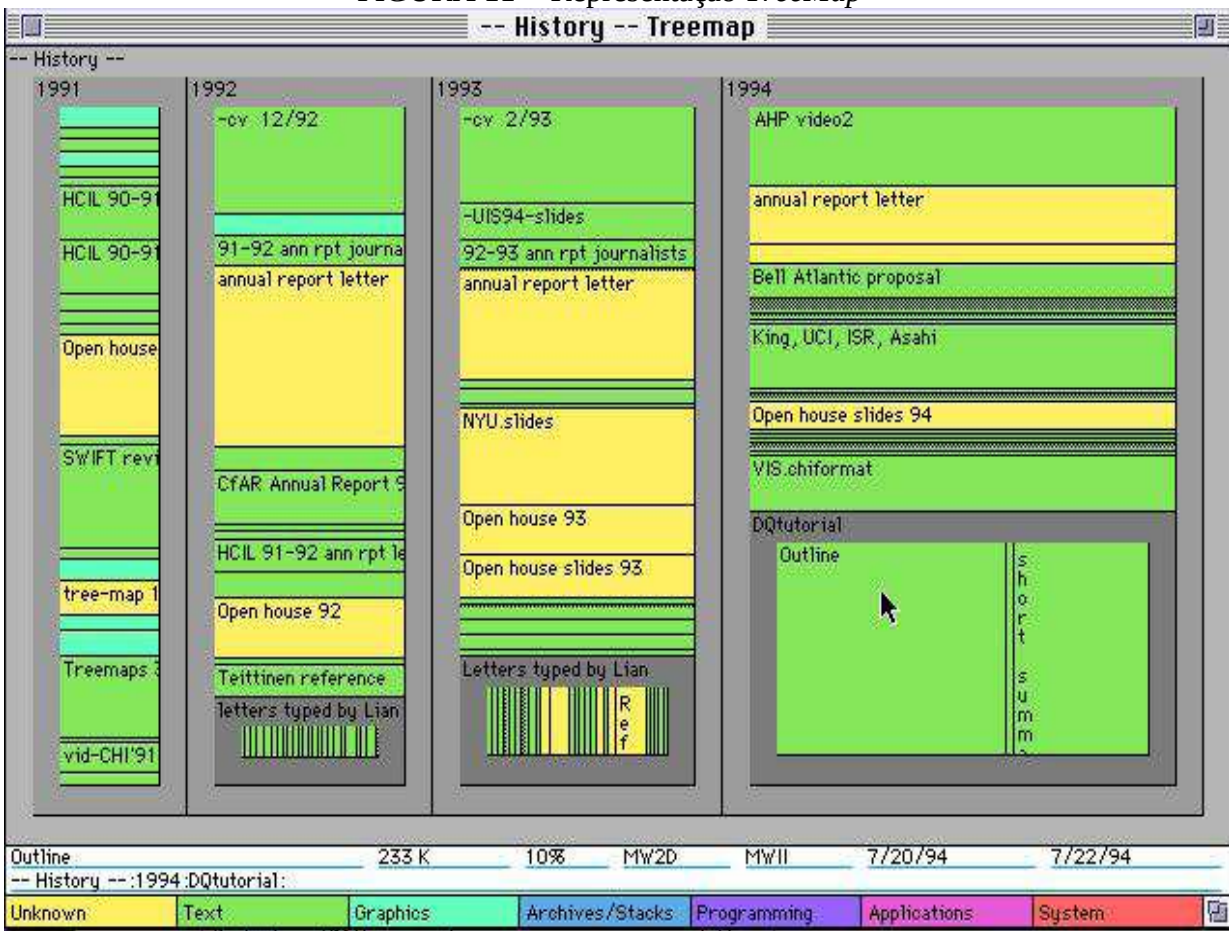
Fonte: Stefaner (2007, *online*).

Tree Map

TreeMap é uma técnica de visualização com limitações espaciais, que representa estruturas hierárquicas. Apesar da limitação, é muito eficaz em mostrar atributos de dados, representados como nomes de nós. Uma *TreeMap* permite ao usuário comparar os nós e subárvores, mesmo que eles estejam em diferentes profundidades dentro da árvore, podendo dessa forma ajudar na detecção de padrões. Segundo Shneiderman (1998), *treemaps* caracterizadas por nós aninhados para representação da hierarquia e partição espacial recursiva de acordo com o tamanho das subárvores. Além disso, são mais eficientes que a representação de dados árvore simples e para a visualização dos nós folha. Em contrapartida,

é ineficiente na representação dos nós que não são folhas.

FIGURA 11 - Representação *TreeMap*



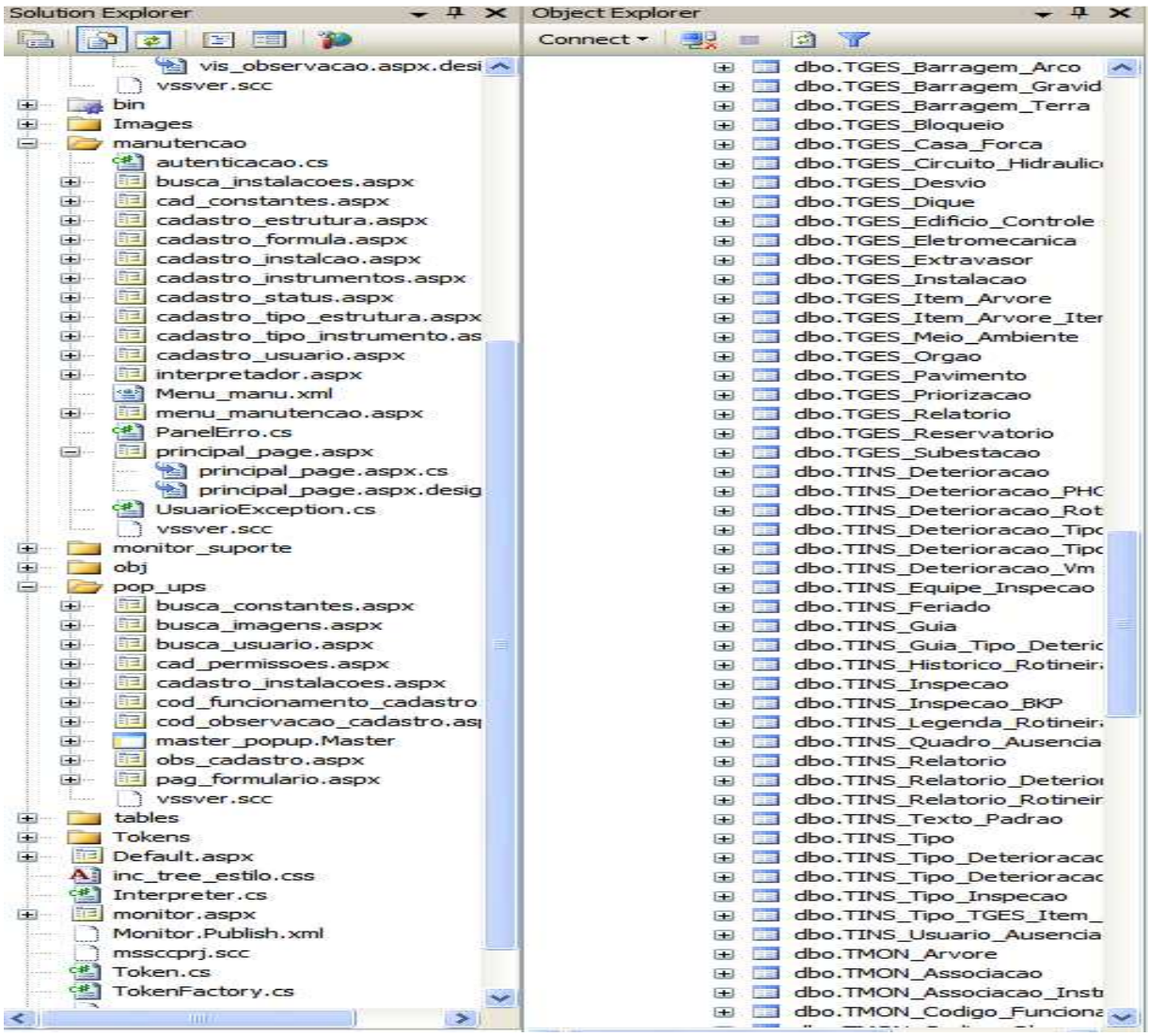
Fonte: Shneiderman (1998, *online*).

Árvores indentadas

Árvores indentadas são muito utilizadas para representação de sistemas de arquivo. Mesmo que tal tipo de árvore necessite de muito espaço vertical, permite a fácil exploração e busca de um nó específico. No entanto, não facilitam inferência multiescalar.

Além disso, o leiaute indentado permite fácil identificação dos nomes dos nós. Os dados de cada atributo podem ser visualizados ao lado da hierarquia. Para navegação o usuário deve clicar no nó para mostrar ou esconder seus filhos (STANFORD..., 2010).

FIGURA 12 - Árvore Indentada

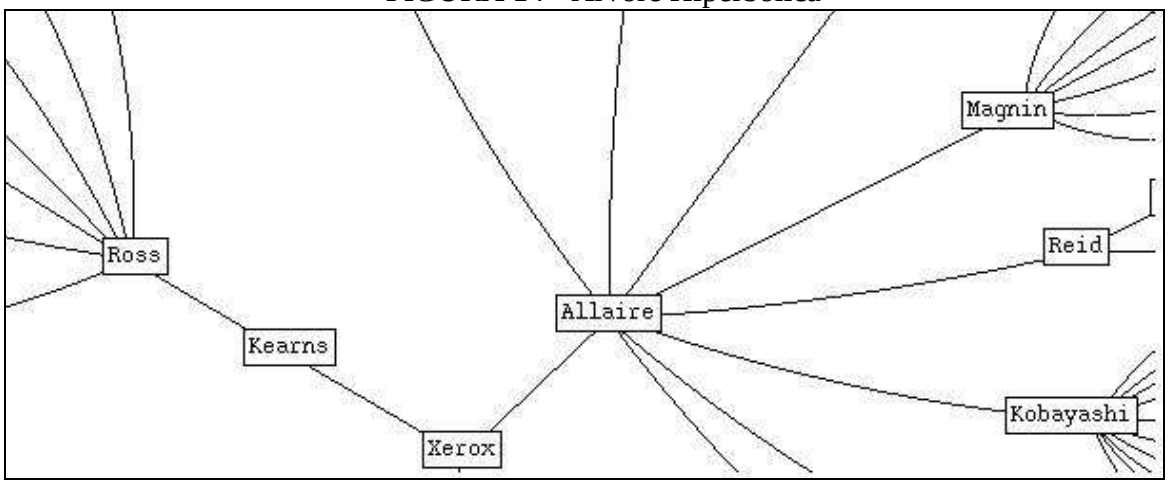


Fonte: Autoria própria

Árvore hiperbólica

Segundo os autores Lamping, Rao e Pirolli (1995), a técnica de visualização conhecida como árvore hiperbólica exibe inicialmente uma árvore com seu nó raiz no centro. Contudo, essa visualização pode ser modificada facilmente para trazer um outro nó para o centro da árvore, fazendo com que esta tenha um novo foco. Em todos os casos, a quantidade de espaço disponível para um nó cai como uma função de sua distância ao centro. Assim, o contexto inclui várias gerações de pais, irmãos e filhos, tornando mais fácil para o usuário para explorar a hierarquia, sem se perder.

FIGURA 14 - Árvore Hiperbólica



Fonte: SIGCHI (2010, *online*).

Essa técnica explora a geometria hiperbólica, de acordo com Lamping, Rao e Pirolli (1995). O propósito da abordagem é estabelecer a hierarquia no plano hiperbólico e mapear esse plano em uma região de exibição circular.

O plano hiperbólico consiste de um plano não-euclidiano, onde linhas paralelas divergem umas das outras. Graças a propriedade da circunferência de um círculo no plano hiperbólico, ele pode crescer exponencialmente com o seu raio. O espaço se torna mais disponível com o aumento da distância. Assim, hierarquias que tendem a se expandir exponencialmente com a profundidade, e podem ser definidas no espaço hiperbólico de maneira uniforme. Desse modo, a distância (medida na geometria hiperbólica) entre pais, filhos e irmãos é aproximadamente a mesma em todos os lugares da hierarquia.

Esta é a técnica de visualização de informação “foco+contexto” utilizada para representar grandes quantidades de dados. È caracterizada por: possibilidade de ampliar o foco e reduzir o resto; o foco é agregado ao contexto; por interação técnica, pode ser combinado com leiaut diferente.

Árvore em cone (*Cone Tree*)

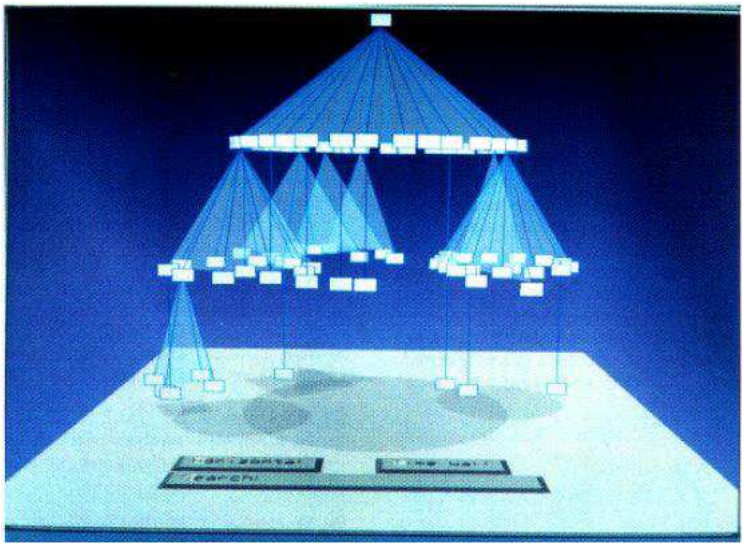
Essa técnica de visualização de informações hierárquicas consiste de uma representação 3D de um cone. O nó raiz fica em uma extremidade do cone, normalmente no topo da tela, ou à esquerda da tela. Os outros níveis da árvore, as sub-árvores, são desenhados um após os outros como pequenos cones, que têm seu raio diminuído na medida em que se afastam do nó raiz. Esse encaminhamento é seguido do nó raiz ate os nós folhas. Cada nível é representado por um cone translúcido. Os nós da árvore são normalmente representados como retângulos (ROBERTSON; MACKINLAY; CARD, 1996).

Essa técnica possibilita apresentar grande quantidade de dados de uma única vez sem a necessidade da rolagem da tela. Ela também permite a retração e a expansão de nós que não estão sendo visualizados no momento. Segundo Robertson, Mackinlay e Card (1996), a utilização do sistema pelo usuário é feita da seguinte forma: o usuário seleciona um nó; o nó selecionado vem para o centro da tela; os nós localizados no cone, onde nó selecionado está representado, ficam em evidência; a árvore, como um todo, e as suas sub-árvores seguem a rotação da árvore do nó selecionado mantendo assim a estrutura inicial.

A árvore também pode ser explorada através de interação animada, de forma que as relações hierárquicas das sub-árvores possam ser mais bem avaliadas pelo usuário. O usuário, nesse caso, rotaciona o cone principal e os sub-cones para enxergar as relações entre os dados. Para resolver problemas de visualização dos textos de atributos dos dados, no modelo *Cone Tree* (na vertical), o único texto exibido é do nó selecionado; na *Cam Tree* (na horizontal), no qual o cone fica posicionado horizontalmente em relação à tela, os textos relativos a todos os nós são mostrados na tela.

A representação 3D é utilizada para aumentar a quantidade de dados visualizada. As representações 2D nem sempre estão aptas á quantidade de dados na tela (ROBERTSON; MACKINLAY; CARD, 1996).

FIGURA 15 - Cone Tree



Fonte: SAP... (2010, *online*).

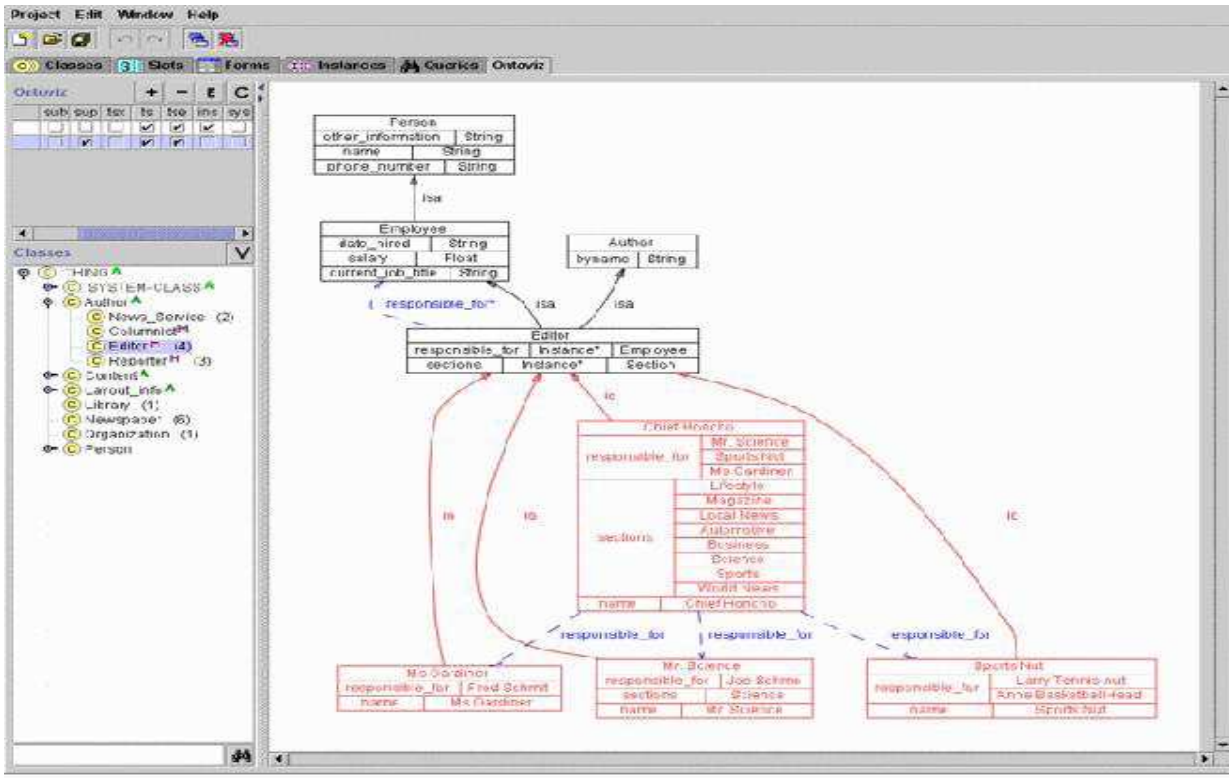
2.2 - Visualização de Ontologias

Katifori et al. (2007) apresentam uma lista de *softwares* de visualização de ontologias, a qual foi utilizada como ponto de partida para a descrição desta seção. Além dos *softwares* citados nessa lista, acrescentam-se outros de forma a melhor abranger o campo de estudo.

OntoViz

Trata-se de um *plug-in* do *software Protégé*[6] que permite visualizar ontologias utilizando o *GrafViz*[7]. Os tipos de visualização possíveis são configuráveis e incluem: a seleção do conjunto de instancias ou classes para visualização de uma pequena parte da ontologia; a exibição das relações (*slots*) e de suas arestas, a especificação de cores para nós, além da possibilidade de escolha de instâncias e classes para a aplicação de operadores.

FIGURA 16 - Tela Onto Viz



Fonte: ONTO... (2010, *online*).

Jambalaya

[6] Protégé: Editor de ontologias livre que tem seu código fonte aberto (Onto..., 2010, *online*).
[7] Uma ferramenta que provê uma forma de representar informações estruturais como diagramas de grafos abstratos e redes

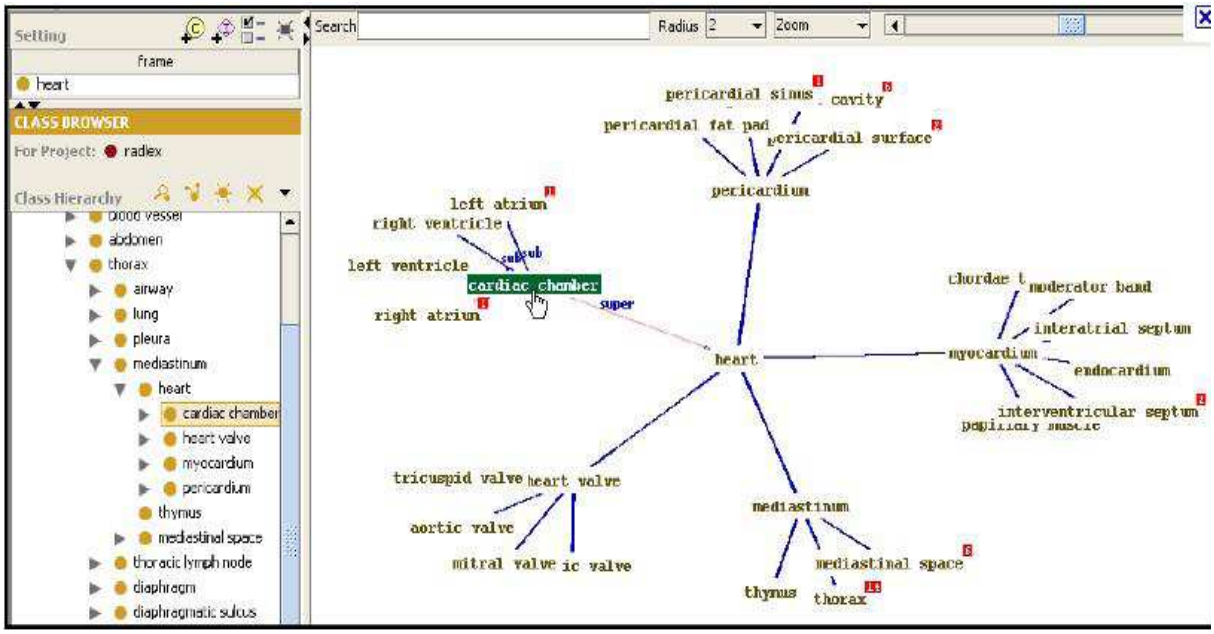
Também é um *plug-in* criado para utilização no Protégé, que utiliza um aplicativo denominado *Shrimp*[8] para visualização 2D de ontologias.

TGVizTab (*TouchGraph Visualization Tab*)

Integra a técnica *TouchGraph*[9] no Protégé para criar um sistema de visualização de ontologias, como uma rede de grafos direcionados de classes, instâncias e relações.

Segundo Alani (2003), o sistema permite aos usuários do Protégé visualizar gradualmente as ontologias exploradas através da navegação pelos subgrafos interconectados.

FIGURA 17 - Tela TGVizTab



Fonte: SPRINGERIMAGES (2010, *online*).

OntoSphere

OntoSphere3D é uma ferramenta para visualização de ontologias, já citadas anteriormente na seção 3.1.2, que faz uso de espaço tridimensional, enriquecendo a informação através estímulos visuais.

A ferramenta aborda as questões de visualização de ontologias através da adoção de um mecanismo dinâmico de expansão, e da apresentação de pontos de vista diferentes, em diferentes granularidades, a fim de conceder navegabilidade no modelo representado.

OntoTrack

Trata-se de uma abordagem que utiliza visão integrada para promover melhorias à navegação, edição e manipulação de grandes ontologias em formato OWL *Lite*[10]. É um

[8] *Shrimp*: é uma técnica e aplicação ao mesmo tempo. Foi desenvolvido para auxiliar na visualização e navegação de espaços de informação Cf CHISEL... (2010, *online*)..

[9] *TouchGraph* é um mecanismo desenvolvido em java para a criação e navegação de grafos

sistema baseado em *SpaceTree*[11] e implementado em Java2D. Utiliza um sistema animado para a visualização da expansão e retração das classes e suas subclasses, entre outras técnicas para facilitar sua utilização.

GoSurfer

Utiliza dados do projeto *Gene Ontology* [12] na análise de conjuntos de genes obtidos dos estudos de genoma, representando-os graficamente. As categorias são representadas como árvore, as quais podem ser manipuladas por um usuário. Permite comparações entre dados genéticos, então, pode ser utilizado para testes estatísticos, permite a marcação de partes da estrutura de dados, além da exportação dos dados e da representação gráfica gerada.

OZONE

Ferramenta utilizada para navegação e busca em ontologias. O sistema visualiza os parâmetros de consulta e fornece navegação interativa em ontologias DAML[13].

3.3 - Implicações para Pesquisa

Buscou-se analisar neste capítulo a visualização de informação sob duas perspectivas: tipos de visualização e técnicas de avaliação, sendo as duas codependentes. A técnica é necessariamente dependente do tipo de avaliação que foi escolhido. Com a pesquisa dos vários tipos de visualização descritos foi possível determinar quais tipos de técnicas são melhores para um determinado tipo de visualização de dados. O levamentos dos tipos possibilitou a descoberta que cada técnica está contida em um tipo de visualização. Já o levantamento das técnicas possibilitou a determinação do conhecimento necessário sobre o tratamento específico de dados citados (científicos, cartográficos, dados genéricos, etc.).

Com a pesquisa dos tipos de visualização, é possível imaginar suas potencialidades, como por exemplo, em algumas descobertas realizadas até aqui, durante essa pesquisa:

- Nos tipos de visualização unidimensionais é possível criar listas e símbolos lineares;

---

[10] Forma com menor expressividade que a OWL completa Cf. W3C (2011, *online*).
[11] *SpaceTree*: é um navegador que se baseia no tradicional leiaute de diagram de nós e *links*.
[12] Iniciatiava na área de bioinformática que tem como objetivo uniformizar a forma de representação de genes Cf THE GENE... (2010, *online*).
[13] DARPA *Agent Markup Language*

- Já nos tipos bidimensionais é possível criar dados bidimensionais em metáforas visuais;
- Sobre o tratamento específico de dados dos domínios de três dimensões (largura, altura, profundidade) podemos afirmar que tipos tridimensionais são utilizados para representação visual;
- Nos tipos multidimensionais são representados dados multifacetados;
- Os tipos de visualização hierárquica podem permear as quatro anteriormente citadas.

O levantamento dos tipos de visualização permitiu a diminuição do escopo de potenciais técnicas de visualização que podem ser utilizadas dentro da proposta deste projeto. Esse ponto é de grande importância na determinação do caminho de construção da ferramenta, e ajuda a determinar se a utilização de uma das ferramentas pesquisadas atenderia de melhor forma a pesquisa proposta.

4 – Fundamentos Técnicos

Nessa seção descreveremos brevemente as bases técnicas do presente trabalho. Ela estará dividida da seguinte forma: 4.1 Linguagens de programação introdução; 4.2 Bibliotecas, libxml2 e *Parser;* 4.5 Infovis*Tollkit*; 4.6 Zope.

4.1 - Linguagens de Programação

Linguagens de programação são formas estruturadas de se descrever um conjunto de tarefas, ou seja, uma maneira de se dar instruções para o computador. Através delas os programas e sistemas de computador são escritos. No entanto, ainda é necessário traduzir essa maneira de escrever para a linguagem alfanumérica que o computador "compreende". Esse processo leva o nome de compilação (e lincagem).

Existem definições de linguagens de programação que levam em conta como o programa escrito por elas são executados e traduzidos para linguagem de máquina e em relação ao quão distante da linguagem natural elas são.

Em relação à forma de execução e tradução dos programas gerados pelas linguagens temos a divisão: interpretadas versus compiladas. Na primeira a tradução da linguagem de programação para linguagem de máquina ocorre à medida que o programa é executado, enquanto na segunda essa tradução é feita uma única vez e armazenada em sua nova forma para que possa ser executada diversas vezes.

Em relação à distância da linguagem natural elas são divididas em alto e baixo nível. Alto nível são as linguagens consideradas mais próximas às linguagens naturais e baixo nível são as mais próximas às linguagens de máquina.

Nesse trabalho utilizamos três linguagens de programação específicas para o desenvolvimento das ferramentas utilizadas na aplicação da pesquisa:

- Python: linguagem interpretada, orientada a objetos com sintaxe clara e objetiva, em termos de classificação quanto ao quão distante é da linguagem natural é considerada alto nível. Incorpora várias bibliotecas, módulos, classes e tipos de

objetos diferentes, além de poder ser estendidas por outras linguagens como C e C++. Uma vantagem nesta linguagem é o fato de ser interpretada, isso a torna portável para vários sistemas operacionais diferentes;

- C++: linguagem multi-paradigma baseada na linguagem de programação C, ela é considerada uma linguagem de médio nível;
- Java Script: linguagem script, orientada a objetos, com suporte a programação funcional. É uma das principais linguagens utilizadas para interpretação nos navegadores. É utilizada em Ajax, interatividade web entre outras funcionalidades em programação voltada para internet.

4.2 – Bibliotecas, Libxml2 e *Parser*

Bibliotecas são coleções de códigos que podem ser utilizados por vários sistemas diferentes. Nelas são encapsulados códigos, dados e estruturas que são usadas como auxiliares no desenvolvimento de softwares. Desta forma pode-se compartilhar código, modularizar sistemas e acrescentar ou retirar pequenos pedaços de códigos específicos.

Dentro deste trabalho são utilizadas algumas bibliotecas chave para do desenvolvimento das ferramentas necessárias ao desenvolvimento da pesquisa. São elas, *jit* da linguagem de Java Script (JAVASCRIPT..., 2010, *online*) e *libxml* da linguagem C++ (THE XML C..., 2010, *online*), as duas serão explicadas nos próximos tópicos dessa seção.

Libxml2

Um *framework* e *toolkit* para análise sintática de XML desenvolvida na linguagem de programação C e C++ para o projeto Gnome (GNOME..., 2007, *online*). Está dentro de um projeto de software livre o que permite sua utilização dentro dessa pesquisa. Mesmo pertencendo de um projeto específico ele pode ser usado em uma série de plataformas diferentes (Linux, Unix, Windows, CygWin, MacOS, MacOS X, RISC Os, OS/2, VMS, QNX, MVS, VxWorks, ...). A biblioteca central do framework pode ser estendida por varias outras linguagens além da linguagem inicial pela qual foi feita, no caso, C.

A biblioteca implementa a analise sintática para uma série de padrões de metadados: padrão XML; Namespace XML; XML base; RFC; HTML4; XPATH; XPointer; XInclude; padrões ISO-8859-x; SGML; W3C XML SchemasPart 2: Datatypes; W3C xml:id.

Além de dar suporte a padrões não totalmente implementados: *DocumentObjectModel* (DOM); RFC959 (FTP); RFC1945 (HTTP); SAX.

Parser

Processo pelo qual se faz uma análise sintática em uma sentença dada. Essa análise visa gerar uma estrutura com as características gramaticais da sentença, essas características são dadas por uma gramática formal previamente estabelecida. Normalmente essas características são traduzidas em uma estrutura de dados na forma de árvore (grafo aberto).

Parsers são usados em uma variada gama de aplicações como compiladores, *crawlers*, máquinas de busca na web, entre outras aplicações. Cada uma das aplicações tem um papel específico dentro de sua definição inicial. Em um compilador, por exemplo, as análises sintáticas junto às análises léxicas e semânticas transformam linguagens de programação em linguagem de máquina. No caso de compiladores, os *parsers* têm processamento linear e determinístico.

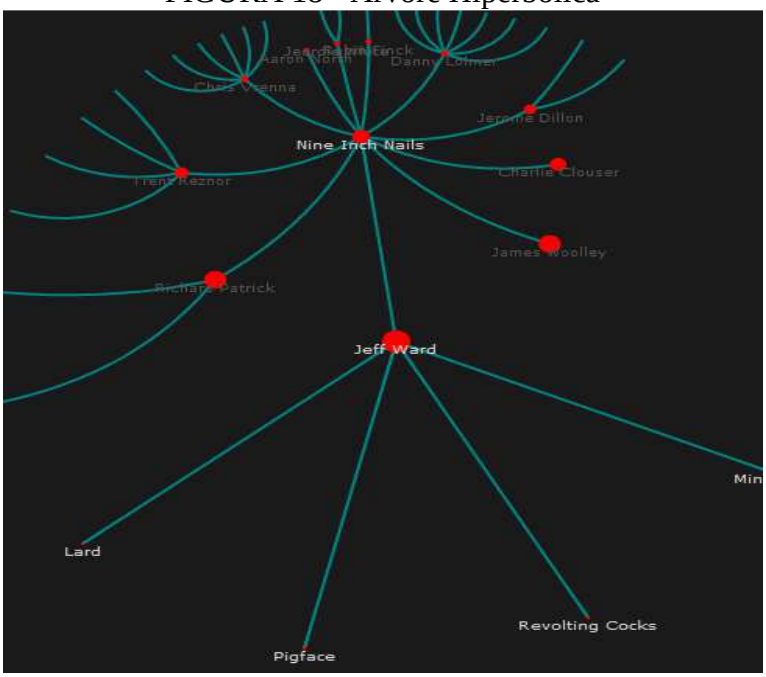
Já quando são utilizados para processamento de linguagem natural, não se espera o mesmo comportamento, uma vez que linguagem natural não necessariamente vai estar estritamente nas regras gramaticais formais.

4.4 - Infovis Toolkit

Biblioteca desenvolvida em Java com base na arquitetura de s*oftware Swing Gui*. Ela é organizada em cinco partes diferentes: tabelas, colunas, visualizações e componentes.

Ele utiliza as cinco partes diferentes para dar uma forma estável de construção de estruturas visuais. Utiliza o sistema de tabelas e colunas para melhor aproveitamento de memória. Mapeia a estrutura de dados em visualizações, que dão suporte a *labels* dinâmicos e foco contexto (FIG. 18). Para isso utiliza algoritmos de visualização encapsulados na biblioteca.

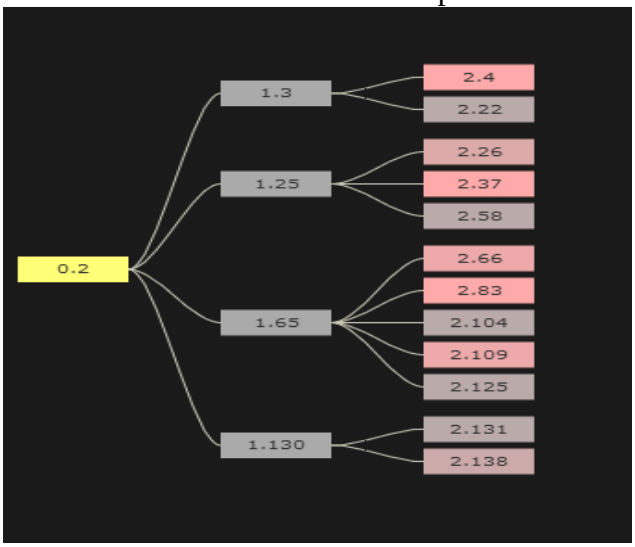
FIGURA 18 - Árvore Hiperbólica



Fonte: *InfovizToolkit* (2011, *online*)

Tem suporte a um conjunto de necessidades de visualização de informação considerável, além de conseguir encapsular as funcionalidades de modo a permitir modularidade e reuso. Os componentes são filtros e consultas dinâmicas que permitem a manipulação das imagens de forma eficaz (FIG. 19).

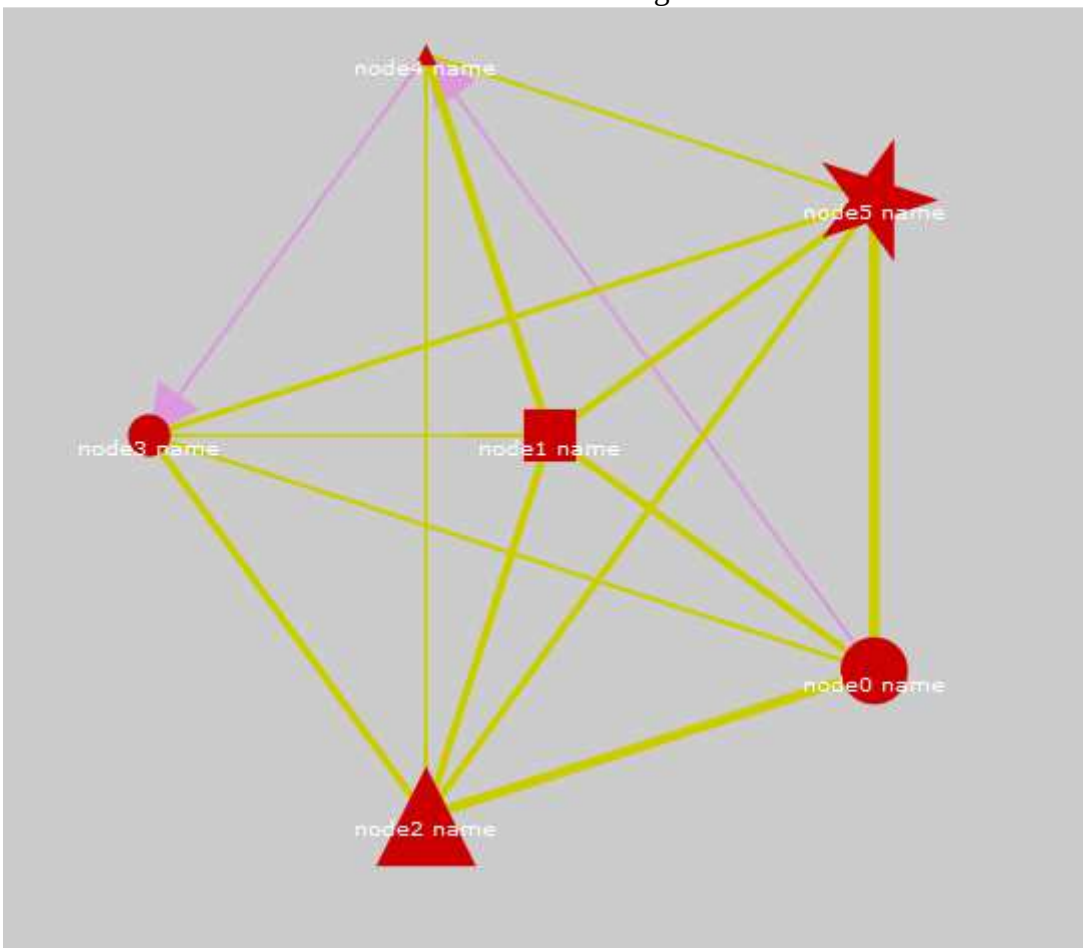
FIGURA 19 - Árvore Espacial



Fonte: *InfovizToolkit* (2011, *online*)

Também permite a criação de estruturas como grafos, barra, diagramas barras paralelas, *tree maps* entre outros tipos, como pode ser visto na FIG. 20.

FIGURA 20 – Infovis grafo



Fonte: *InfovizToolkit* (2011, *online*)

4.6 - Zope

Servidor voltado para web que permite disponibilizar uma serie de aplicações de forma unificada. Dentro dele é possível armazenar e instalar diferentes sistemas.

Zope é um framework para construção de aplicações para web que consiste em vários componentes diferentes que trabalham de forma unificada para a construção de tais aplicações. Os componentes são:

Servidor web

O Zope vem com um servidor web embutido que é utilizado como repositórios de conteúdo para usuários e administradores, ou seja, serve dados assim como apache e IIS14. O

[14] Aplicativos que guardam e mantem páginas da web, ou seja, servidores de páginas de *internet*

sistema permite que se trabalhe com servidores web diferentes caso o usuário não queira utilizar esse servidor interno. Além de trabalhar com qualquer outro servidor web que suporta WSGI[15] do Python padrão.

Interface de administração web

Interface de administração e desenvolvimento, nela é possível desenvolver páginas, scripts e aplicações web me geral. Tem suporte a Java *script*, *Python script*, dhtml e HTML.

Suporte a *script*

As aplicações dentro desse servidor podem ser feitas em várias linguagens de programações diferente, *Python* é uma linguagem nativa (ZPT) .

Implementação

A maioria do código Zope é implementado em Python, com a exceção de um pequeno número de módulos principais implementadas em C, isso feito por razões de desempenho. Python tem sido uma linguagem orientada a objetos desde sua primeira versão no início de 1990.

Zope utiliza banco de objetos ZODB, ou seja, um banco de dados orientado a objeto interno e leve. A maioria das soluções de CMS tende a usar um banco de dados relacional padrão. Zope sempre se aproximou da camada de armazenagem de uma maneira diferente. O ZODB é uma base de armazenamento de objetos utilizada para armazenar objetos Python. Usando um armazenamento de objetos para a programação é simples já que tudo é um objeto dentro de Zope e Python. O ZODB é adequado para pequenas e grandes quantidades de dados.

4.6 - Linguagens de representação

Apresentam-se nessa seção algumas informações básicas sobre as duas linguagens de representação baseadas na Web, em uso atualmente, a RDFS e a OWL.

[15] Interface de comunicação entre servidores web

4.6.1 - RDFS (*Resource Description Framework Schema*)

Segundo o W3C (2005), RDF Schema define classes e propriedades que podem ser utilizadas para descrever outras classes, propriedades e outros recursos.

RDF fornece uma maneira simples de expressar declarações sobre recursos, usando propriedades e valores. No entanto, integrantes de comunidades sobre RDF precisavam também ter a capacidade de definir quais vocabulários seriam utilizados em suas declarações, especificamente, para indicar que eles estão descrevendo tipos específicos ou classes, e que utilizarão propriedades específicas para descrever as classes. Dessa forma, o RDFS foi criado como uma padronização para o uso de RDF.

As facilidades do RDF Schema são fornecidas na forma de vocabulário RDF, ou seja, como um conjunto especializado de recursos RDF predefinidos com seus próprios significados.

4.6.2 – OWL (Ontology Web Language)

Segundo informações da W3C (2004), a OWL é uma linguagem criada para auxiliar na modelagem de ontologias. A OWL é utilizada para processar aplicações computacionais baseadas em informações contidas em documentos, contrapondo a situação das informações de um documento que são interpretadas apenas por humanos. OWL pode ser usada para representar o significado de termos em vocabulários e os relacionamentos entre esses termos. A representação dos termos e relações é chamada ontologia.

A OWL possui maior expressividade do que XML, RDF e RDFS, apresentando possibilidades de inferências computacionais. Existem três sublinguagens OWL:

- OWL *Lite*: criada para gerar hierarquias com restrições simples. É a linguagem mais fácil de migrar para outras linguagens de modelagem e de menor complexidade;
- OWL DL: desenvolvida para casos mais complexos que exigem restrições, mas ainda mantém a capacidade de ser processada por uma aplicação computacional com complexidade relativamente baixa, ou seja, uma obtenção de soluções ótimas

em tempo finito. OWL DL inclui todos os construtores da linguagem, porém sua utilização só pode ocorrer sob algumas restrições.

- OWL *Full*: utiliza maior capacidade de expressão, uma vez que independe da sintaxe RDF. Não existem garantias de que o resultado seja computável.

Cada uma das linguagens é uma extensão do seu predecessor. O que é válido no predecessor é valido na linguagem "derivada", ou seja, o que é valido na OWL *Lite* é valido na OWL DL, e o que é valido na OWL DL é valido na OWL *Full*.

4.7 - Implicações para pesquisa

O capítulo visou explicar alguns conceitos técnicos que foram utilizados na construção das ferramentas utilizadas para o suporte ao desenvolvimento da pesquisa. Com essa pesquisa conseguimos o subsídio necessário para o desenvolvimento das ferramentas que, baseadas nas técnicas de visualização de e alinhamento de ontologias, darão o suporte para a aplicação do método desenvolvido para avaliar ontologias baseado em alinhamento visual semiautomático. Além de dar apoio ao entendimento do andamento dos processos de desenvolvimento das ferramentas computacionais.

Outra motivação importante para esse passo de pesquisa é a necessidade de se tomar decisões técnicas em face aos desafios de construção e analise automática de ontologias modeladas em OWL. Com esse suporte foi possível escolher formas de se fazer essa analise com baixo custo computacional e principalmente construir ferramentas de acesso em rede, o que permite o trabalho colaborativo em uma etapa posterior do trabalho.

A análise de cada uma das bases técnicas apresentadas auxilia na formulação das maneiras de ser responder as questões levantadas nos objetivos da pesquisa.

5 - Metodologia

Esta seção descreve os passos seguidos para realização da pesquisa. Tal pesquisa compreendeu dois momentos: o primeiro concerne na criação de um método de avaliação e, no segundo momento foram realizados os testes.

O projeto desenvolveu-se através da busca do conhecimento já produzido a cerca da área de visualização de informação e do conhecimento acerca de avaliação e alinhamento de ontologias, para criação de um método de avaliação de ontologias baseado em alinhamento visual semiautomático. Esses passos podem ser contemplados por alguns tipos de pesquisa descritos na literatura, a saber:

- Quanto aos procedimentos técnicos, pesquisa bibliográfica que, segundo Cervo e Bervian (1995), busca explicar um problema a partir do que já foi publicado sobre o assunto que o permeia, ela pode ser utilizada como a pesquisa em si ou parte de outro tipo de pesquisa. No caso dessa pesquisa é utilizada como base para o desenvolvimento de um conjunto de artefatos que juntos compõem parte do método criado;
- Ainda quanto aos procedimentos técnicos, estudo de caso, pois envolve o estudo profundo de uns poucos objetos de maneira que se permita seu amplo conhecimento; como fase testes, um estudo de caso foi aplicado a duas ontologias biológicas descendentes da ontologia BFO. Isso foi feito para verificação da aplicabilidade dos métodos propostos à avaliação de ontologias;
- Quanto à natureza, pesquisa aplicada, pois objetivou gerar resultados de aplicação prática direta para as organizações;
- Quanto à abordagem do problema, pesquisa qualitativa, visto que as avaliações realizadas não puderam ser mensuradas quantitativamente em todos os seus aspectos;
- Quanto aos objetivos, pesquisa exploratória que, segundo Gil (1999), visa aprofundamento sobre um assunto ou fato específico e de preferência pouco explorado. Elas visam proporcionar maior conhecimento sobre o objeto de pesquisa, proporcionando desta forma a possibilidade de uma criação de hipótese que possa ser explorada dentro das pesquisas.

- Quanto aos sistemas tecnológicos desenvolvidos, pesquisa de *design*, pois ali cabe a análise de novos padrões e ferramentas criadas para desenvolver ou aprimorar um processo. Segundo Vaishnavi e Keuchler (2004, p.3) a pesquisa de design é definida como: “Pesquisa de design envolve a análise do uso e desempenho de artefatos construídos, com a finalidade de entender, explicar e melhorar comportamentos de aspectos do artefato”. Este tipo de pesquisa se encaixa dentro dos objetivos propostos quando nos permite avaliar o método desenvolvido no decorrer do processo.

Nas subseções seguintes, apresenta-se o método para avaliação, o qual é analisado através de sua aplicação em duas ontologias distintas. Em seguida, apresentam-se os instrumentos idealizados para a pesquisa sobre os benefícios das técnicas de visualização da informação para o processo de avaliação. Nisso se incluem *parser*, algoritmos de *matching*, roteiros de *matching*, desenvolvimento do protótipo e roteiro de avaliação de ontologias baseada em alinhamento. Pretende-se discutir sobre os testes que serão aplicados utilizando-se os algoritmos e a versão final do protótipo.

Com isso pretende-se responder à pergunta proposta na seção 1, acerca da utilização de ferramentas de visualização de conhecimento na avaliação de ontologias através do alinhamento semiautomático.

Desenvolveu-se um método de avaliação de ontologias baseado no alinhamento, mereológico e gráfico, semiestruturado de duas ontologias. Com ele, visa-se à busca de auxílio do desenvolvimento de ontologias através da análise da modelagem, baseando-se nas técnicas de alinhamento descritas. Acredita-se que essas técnicas podem mostrar possíveis falhas de modelagem. Portanto, buscou-se desenvolver um método que exponha esses erros, além de uma forma de análise desses possíveis erros.

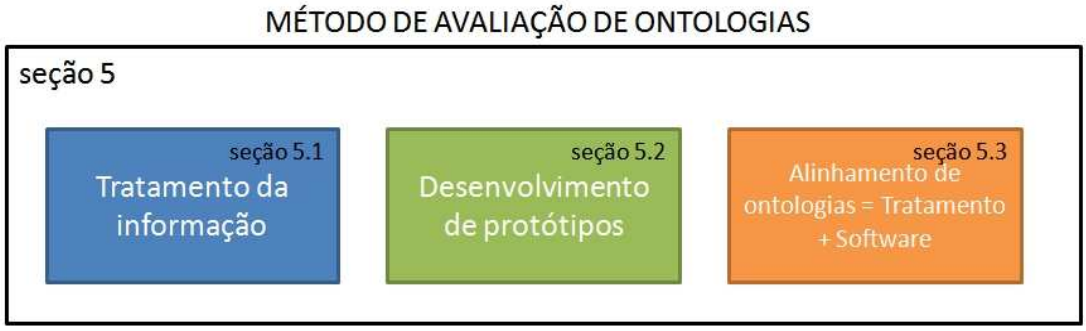
A fim de identificar possíveis falhas na modelagem, a criação do método foi dividida em três fases: Tratamento de informação, desenvolvimento de protótipos e alinhamento de ontologias.

- Roteiros: foram criados três roteiros; um de alinhamento semiautomático baseado em string, um baseado em estrutura e um de avaliação que utiliza como suporte as avaliações feitas durante a aplicação dos dois primeiros;
- Sistemas: Três algoritmos de alinhamento baseados em *string* (nome) e dois baseados em estrutura (mereológico) que geram de uma “meta-ontologia” cada; um protótipo de visualização que exibirá o resultado das etapas de alinhamento direto para o alinhamento visual e avaliação das ontologias;

- Aplicação: Junção das duas fases anteriores em uma série de processos que consolidam o método através de roteiros especializados voltados para a utilização do sistema criado.

O restante da presente seção está organizado da seguinte forma: a seção 5.1 (Tratamento de Informação) descreve os roteiros de avaliação de ontologias, a seção 5.2 (Desenvolvimento de protótipos) a construção do *software* que é utilizado como segunda parte do método de avaliação desenvolvido, enquanto a seção 5.3 (Alinhamento de Ontologias) apresenta a aplicação dos roteiros desenvolvidos para avaliação, o qual é desdobrado em duas fases de alinhamento, desta forma consolidando o método proposto. A figura 21 traz o esquema gráfico dos passos do método.

FIGURA 21 - Método de Avaliação de Ontologias

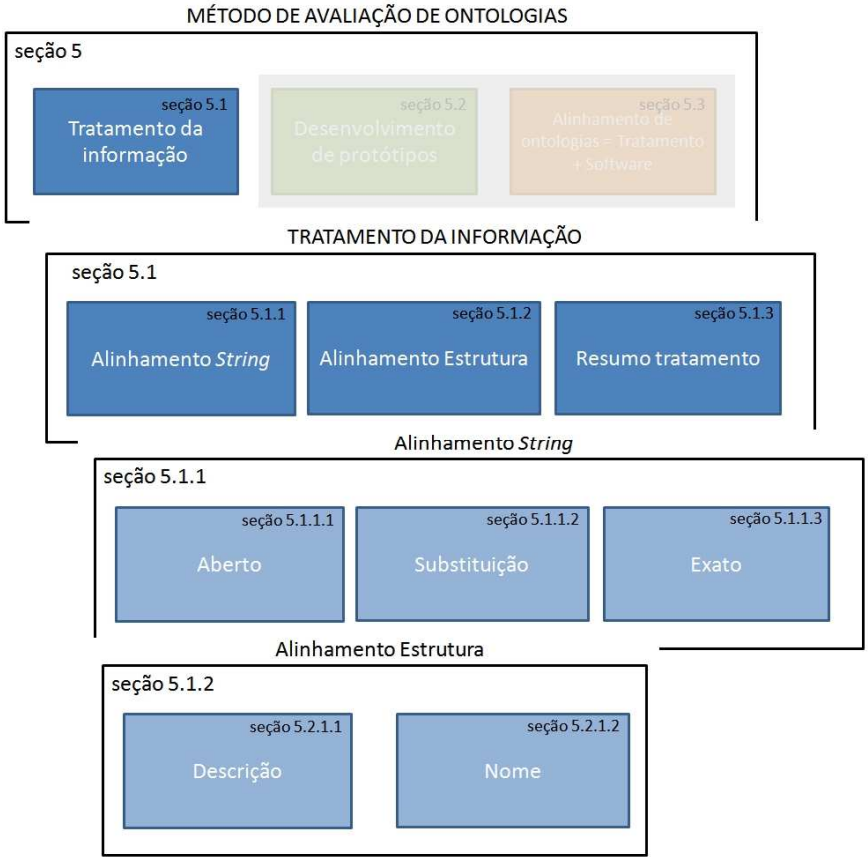


Fonte: Própria

5.1 – Tratamento de Informação

Esta seção traz a descrição do roteiro da primeira etapa do método de avaliação. Nesta parte do método, as avaliações necessitam mais dos conhecimentos do avaliador, pois métricas severas são adicionadas ao processo nas fases de sistema e aplicação. Um exemplo deste tipo de métrica são as medições do número de letras discordantes entre dois nomes retornados no processo de alinhamento por *string*. A figura 22 traz de forma esquemática como esta seção se desenvolve.

FIGURA 22 - Esquema: Tratamento de Informação



Fonte: Propria

Desta forma, apresentam-se aqui os procedimentos que devem ser executados para que duas ontologias sejam avaliadas através de alinhamento e transformação em metáforas visuais. Esses procedimentos foram desenvolvidos para ter como suporte o sistema descrito na seção 5.2 deste capítulo. No entanto, ele pode ser aplicado utilizando-se qualquer outro sistema (ou conjunto de sistemas) que tenha características semelhantes. O roteiro segue de acordo com os passos descritos à seguir.

Duas ontologias, que aqui chamamos de ontologias base, são escolhidas. Essa escolha leva em conta a possibilidade de alinhamento para a avaliação. As ontologias devem ter a mesma base ontológica e descritiva. Por exemplo, duas ontologias que visam descrever a anatomia humana e que estejam sob as normas da BFO[16] podem ser utilizadas no processo. Entretanto, duas ontologias que visam descrever coisas diferentes (anatomia e móveis de uma

[16] *Basic Formal Ontology*: ontologia de alto nível construída como uma série de ontologias menores que caracterizam facetas da realidade. Isso feito atrvés de vários níveis de granularidade. Disponível em : http://ncorwiki.buffalo.edu/index.php/Basic_Formal_Ontology_2.0

casa) ou são de bases diferentes (uma baseada na BFO e outra na DOLCE[17]) não seriam candidatas à aplicação deste método.

Além dessas regras, a escolha das ontologias para a avaliação através deste método precisa seguir uma visão hierárquica, ou seja, uma ontologia vai ser avaliada através da utilização de uma outra ontologia já consolidada para as comparações e a verificação de alinhamento. A ontologia avaliada deve se encaixar na ontologia consolidada. Como no caso das ontologias CELL e FAO; a primeira, uma ontologia geral sobre células, e a segunda sobre fungos apenas. A ontologia FAO, como sendo de mesma base e padrão de descrição, em teoria, deve se encaixar de forma adequada à ontologia CELL, mostrando possibilidades de interoperabilidade. Neste caso as duas ontologias têm sua base na BFO, que no caso, é considerada uma ontologia consolidada

À essas ontologias aplicam-se três tipos de alinhamento baseados em *string* (exata/exata moderada, substituição e abertos), já descritos no cap. 2 seção 2.3 deste trabalho; e dois tipos de alinhamento baseado em estrutura (*subclassof* e casamento de caracteres), também já descrito no já descritos no cap. 2 seção 2.3 deste trabalho. Para cada alinhamento analisa-se o seu resultado antes de ir para o próximo tipo. Faz-se análise final para caminhar para o próximo passo.

A cada uma das fases de alinhamento aplicam-se uma série de análises aos resultados. Essas análises são baseadas em: i) completude, considerado como o mais importante, uma vez que o objetivo é verificar se as informações estão corretas e completas; e ii) método de Gangemi et al., (2006), onde consideramos as métricas estruturais.

Os parâmetros foram escolhidos tendo em vista os objetivos que a pesquisa visa atender. Como a pesquisa se trata de avaliação para ajudar na melhor modelagem de dados utilizados na disseminação de conhecimento e interoperabilidade de sistemas, escolhemos parâmetros que atendem a esses objetivos.

Nesse contexto, a completude dos dados é escolhida, pois, além de dar a visão de qualidade das informações do domínio modelado, auxilia no alinhamento estrutural das ontologias e esse alinhamento estrutural permite avaliar a interoperabilidade de duas ontologias distintas. Cabe lembrar, que apesar do uso de um roteiro de avaliação, o objetivo geral diz respeito a avaliar uma ontologia específica a partir de sua interoperabilidade com uma ontologia já consolidada e, para isso, utilizar a visualização e o alinhamento como suporte nesse processo.

[17] Ontologia de alto nível que visa modelar as categorias ontológicas vistas dentro da linguagem natural e o senso comum. Disponível em: semanticweb.org/wiki/**Ontology**

Para a atividade de avaliação foram criadas três métricas (TAB.. 1) distintas para estrutura, completude (descrição) e alinhamento. Essas métricas de corretude podem variar de 1 a 5 de acordo com a tab. 3, nas quais 1 significa exclusão da possibilidade de alinhamento naquele ponto e 5 alinhamento ótimo. Para aplicação das métricas é feita uma inspeção visual nas estruturas geradas no processo de alinhamento, como será descrito nas seções 5.1.1, 5.12 e 5.1.3. Essa inspeção visual leva em conta a comparação das novas estruturas com as ontologias base (ontologias sob avaliação) e conhecimento sobre o domínio.

TABELA 1
Sumarização das Avaliações

Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude completude	Avaliação Final

Fonte: Autoria própria

Especificamente para a fase de avaliação que utiliza o alinhamento baseado em *string* é utilizada a tabela 2, que não contém o parâmetro completude.

TABELA 2
Sumarização das Avaliações Fase String

Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Avaliação Final

Fonte: Autoria própria

TABELA 3
Níveis de Equivalência

Equivalência	Pontuação
Ótimo	5
Bom	4
Médio	3
Baixo	2
Excluir	1

Fonte: Autoria própria

A seção 5.1.1 traz o roteiro de avaliação através do alinhamento baseado em *string*; já na seção 5.1.2, apresenta-se o roteiro de avaliação através do alinhamento baseado em estrutura, e a seção 5.1.3 apresenta avaliações dos aspectos gerais das ontologias.

5.1.1 – Roteiro de avaliação através alinhamento baseado em *String*

Nesta seção descreveremos os passos para da etapa de avaliação considerando o alinhamento baseado em *string*. Para cada parte do processo temos uma ação do usuário que gera um retorno de um sistema, desta forma, temos a sequência ação do usuário *versus* retorno para cada ponto de alinhamento e avaliação.

Inicialmente, entre os três tipos de alinhamento baseado em *string*, o baseado em comparações "abertas" deve ser aplicado às duas ontologias base. Para isso, deve-se utilizar um sistema que permita tal alinhamento. Alguns são descritos na literatura, no caso desta pesquisa é utilizado o sistema descrito na seção 5.2 deste capítulo (Protótipo de visualização e alinhamento de ontologias).

Uma análise deve ser feita em cima do resultado obtido após a aplicação desta etapa do processo: deve-se averiguar os pares de palavras que representam pares de classes das ontologias base retornados e atribuir uma nota a esses pares. A essa nota damos o nome de "nível de equivalência". A nota das palavras retornadas é registrada na TAB 4. Essa nota, que deve se basear na TAB 3, leva em conta a similaridade (que é medida de acordo com a ordem e quantidade de letras iguais), além do conhecimento prévio dos avaliadores. Além disso, de acordo com as notas dadas, as palavras são registradas no campo "Candidatas" da TAB 4 se esses pares de palavras são candidatos (registra-se "sim") ou não são candidatos (registra-se "não") ao alinhamento. Para notas menores que três marca-se "não" e exatamente três ou maiores marca-se "sim".

TABELA 4
Equivalência VS Candidatas

Ontologia 1 – Nome	Equivalência	Ontologia 2 – Nome	Candidatas
Classe	Nota para equivalência	Classe	Candidatas ou não ao alinhamento

Fonte: Autoria própria

Esse procedimento é repetido para todos os pares de palavra retornados da aplicação do alinhamento “aberto” às ontologias base até que todos os pares retornados recebam uma nota e as marcações se são ou não candidatos ao alinhamento sejam feitas.

Uma semi ontologia deve ser modelada com todos os pares de palavras marcadas com sim no campo “Candidatas” da TAB 4. Uma inspeção visual deve ser feita entre a nova estrutura e as ontologias base. Baseada nessa inspeção, a TAB 2 deve ser preenchida para essa fase do processo, que aqui chamaremos de *Fase-String-Aberta*. Para essa inspeção visual deve-se utilizar um sistema que permita transformar uma ontologia em uma metáfora visual. Alguns desses sistemas são descritos na literatura, para esta pesquisa é utilizado o sistema descrito na seção 5.2 deste capítulo (Protótipo de visualização e alinhamento de ontologias).

Para o segundo passo desta etapa, entre os três tipos de alinhamento baseado em *string*, o alinhamento baseado em substituições deve ser aplicado às duas ontologias base. Para isso deve-se utilizar um sistema que permita tal alinhamento.

O mesmo procedimento de notas e marcações deve ser aplicado a essa etapa e a partir desse resultado uma semi ontologia deve ser modelada com todos os pares de palavras marcadas com sim no campo “Candidatas” da TAB 4. Uma inspeção visual é realizada entre a nova estrutura e as ontologias base. Baseada nessa inspeção, a TAB 2 deve ser preenchida para essa fase do processo, que aqui chamaremos de *Fase-String-Substituta*. Para essa inspeção visual deve-se utilizar um sistema que permita transformar uma ontologia em uma metáfora visual.

Para o terceiro passo desta etapa, o alinhamento baseado em *string* pelo método exato deve ser aplicado às duas ontologias base.

Aqui não é necessário o procedimento de notas e marcações, pois não existirão pares com palavras diferentes. Uma semi ontologia deve ser modelada com todos os pares de palavras. Uma inspeção visual deve ser feita entre a nova estrutura e as ontologias base, baseado nessa inspeção a TAB 2 deve ser preenchida para essa fase do processo, que aqui chamaremos de *Fase-String-Exata*.

Após o preenchimento da TAB 2 para as três fases (*Fase-String-Aberta*, *Fase-String-Substituta* e *Fase-String-Exata*) deve-se fazer uma média das notas dadas para cada par de palavras existentes nas três fases do processo. Depois, calcular uma média somando todas as notas e dividindo pelo número total de pares e deve ser registrada na TAB. 5.

TABELA 5
Compilação Fase *String*

	Ontologia 1	Ontologia 2
Corretude Alinhamento	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de cada alinhamento.	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de cada alinhamento.
Corretude Estrutura	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades
Avaliação Final	$M = \frac{1}{3}\sum_{n=0}^{3} A_n$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela	$M = \frac{1}{3}\sum_{n=0}^{3} A_n$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela

Fonte: Autoria própria

5.1.2 – Roteiro de avaliação através de alinhamento baseado em estrutura

Nesta seção descrevem-se os passos da etapa de avaliação considerando o alinhamento baseado em estrutura. Aqui também para cada parte do processo tem-se uma ação do usuário que gera um retorno de um sistema, desta forma, temos a sequencia ação do usuário *versus* retorno para cada ponto de alinhamento e avaliação.

O processo começa ao aplicar às duas ontologias base o método de alinhamento baseado em *subclassof*, já descrito no cap. 2 seção 2.3. Para tal um sistema deve ser utilizado como suporte, no caso esta pesquisa utilizou o sistema descrito na seção 5.2 deste capítulo.

Uma análise deve ser feita em cima do resultado obtido após a aplicação desta etapa do processo. Deve-se averiguar os pares de classes (classes filho ou subclasses), suas classes pai e suas respectivas descrições e dar nota ao alinhamento dessas classes. As notas das classes retornadas são registradas na TAB. 6. A nota, que deve ser baseada na tabela 3, leva em conta a similaridade (que é medida de acordo com a ordem e quantidade de palavras iguais nas descrições das classes), além do conhecimento prévio dos avaliadores. De acordo

com as notas dadas, deve-se ser registrar as classes correspondentes no campo “Candidatas” da tabela 6 caso esses pares de classes são candidatos (registrar “sim”); ou não são candidatos (registrar “não”) ao alinhamento. Para notas menores que três marcar “não” e exatamente três ou maiores marcar “sim”.

TABELA 6 –
Comparação de candidatas ao alinhamento fase estrutura com classes pai

Ontologia 1	Equivalência	Ontologia 2	Subclasses	Pontos	Candidata

Fonte: Autoria própria

Esse procedimento é repetido para todos os pares de classe retornados da aplicação do alinhamento *subclassof* às ontologias base até que todos os pares retornados recebam uma nota e as marcações se são ou não candidatos ao alinhamento sejam feitas.

Uma semi ontologia deve ser modelada com todos os pares de palavras marcadas com sim no campo “Candidatas” da tabela 6. Uma inspeção visual deve ser feita entre a nova estrutura e as ontologias base. Baseada nessa inspeção a TAB. 1 deve ser preenchida para essa fase do processo, que aqui chamaremos de *Fase-Estrutura-Subclassof*. Para essa inspeção visual deve-se utilizar um sistema que permita transformar uma ontologia em uma metáfora visual. Alguns são descritos na literatura, no caso desta pesquisa é utilizado o sistema descrito na seção 5.2 deste capítulo.

Para o segundo passo da etapa de avaliação com aplicação de alinhamento baseado em estrutura, o alinhamento baseado em nome deve ser aplicado às duas ontologias base e a nova estrutura criada. Para isso deve-se utilizar um sistema que permita tal alinhamento.

O mesmo procedimento de notas e marcações deve ser aplicado a essa etapa e, a partir desse resultado, uma semi ontologia deve ser modelada com todos os pares de palavras marcadas com sim no campo “Candidatas” da TAB. 6. Uma inspeção visual deve ser feita entre a nova estrutura e as ontologias base, baseada nessa inspeção a TAB.1 deve ser preenchida para essa fase do processo, que aqui chamaremos de *Fase-Estrutura-Nome*. Para essa inspeção visual deve-se utilizar um sistema que permita transformar uma ontologia em uma metáfora visual.

Após o preenchimento da TAB. 1 para as duas fases (*Fase-Estrutura-subclassof e Fase-Estrutura-Nome*) deve-se fazer uma média das notas dadas para cada par de palavras

existentes nas três fases do processo. Depois uma média somando todas as notas e dividindo pelo número total de pares deve ser registrada na TAB. 7.

TABELA 7 –
Compilação Final de Notas estrutura

	Ontologia 1	Ontologia 2
Corretude Alinhamento	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota de cada alinhamento.	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota de cada alinhamento.
Corretude Estrutura	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades
Corretude Descrição	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota dada a corretude de descrição para cada conjunto de entidades	$\mathbf{M = \frac{1}{C}\sum_{n=0}^{C} A_n}$, na qual A é a nota dada a corretude de descrição para cada conjunto de entidades
Avaliação Final	$\mathbf{M = \frac{1}{3}\sum_{n=0}^{3} A_n}$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela	$\mathbf{M = \frac{1}{3}\sum_{n=0}^{3} A_n}$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela

Fonte: Autoria própria

Aqui foi incluída a corretude de descrição, pois essa é levada em conta dentro dos dois tipos de alinhamento aplicado na fase de alinhamento baseado em estrutura.

5.1.3 – Avaliações dos aspectos gerais das ontologias

Durante os dois primeiros processos descritos os aspectos gerais das duas ontologias base devem ser avaliados. Para isso as métricas da tabela 3 devem ser aplicadas as seguintes avaliacões:

- Avaliação do alinhamento: A navegação pelas novas estruturas criadas deve ser feita. Utilizando-se os suportes de visualizacao e alinhamento deve-se compará-las com as estruturas originais (ontologias base utilizadas no processo) deve-se avaliar o alinhamento. Isso é feito através da averiguação das descrições das classes alinhadas, além da análise das novas relações representadas. Isso tudo é feito em relação às ontologias iniciais do processo e com conhecimentos sobre o domínio modelado.
- Avaliação de estrutura: avaliando-se as novas estruturas geradas em relação as estruturas base, classifica-se o nivel estrutural de acordo com a TAB. 8. Para que isso seja possível, é necessário um sistema que de suporte a navegação entre as classes e subclasses, a análise das relações subclassof da lista de classes e subclasses retornada do alinhamento deve ser feita em comparacao com as originais.

TABELA 8
Tabela de avaliação preliminar de avaliação

Ontologia 1	Equivalência	Ontologia 2

Fonte: Autoria própria

- Avaliação de completude: verifica-se o conjunto de dados visualizados, através de um sistema de suporte. Utilizando recursos próprios, o participante (avaliador) avalia que tipo de inferência pode fazer a partir da visualização, considerando que o participante não é especialista no domínio. Para cada par de entidades alinhadas o usuário deve averiguar as relações entre os termos para avaliação de relações redundantes ou errôneas;

A seguinte tabela auxiliar (TAB. 9) deve ser utilizada para a metrificação dos pontos de avaliação considerados acima:

TABELA 9
Tabela auxiliar de avaliação fase avaliação

Tabela Auxiliar de Avaliação	
Classes/Subclasses	Nota

Fonte: Autoria própria

As avaliações acima devem ser feitas seguindo os passos:

- Carrega-se as ontologias participantes do processo;
- Ativa-se o processo de alinhamento;

- Analisa-se as listas de entidades, para cada par de entidades quantifica-se as avaliações de acordo com as métricas definidas na TAB. 3. Uma tabela auxiliar deve ser preenchida com as notas dadas;
- Carregam-se as novas estruturas nas telas de visualização, o tipo de visualização utilizada é escolhido de acordo com a preferência de quem esta aplicando o processo ou o sistema utilizado. Para cada agrupamento de entidades a metrificação deve ser feita e as tabelas de cada tipo de avaliação devem ser preenchidas;
- Após o término do preenchimento das tabelas auxiliares uma sumarização da TAB. 10 de avaliação é feita para que se possa reduzir à uma nota única por parâmetro de avaliação na TAB. 11.

TABELA 10 –
Sumarização das Avaliações

Tabela de Avaliação				
Classes/Subclasses	Corretude Alinhamento	Corretude Estrutura	Corretude Descrição	Avaliação Final

Fonte: Autoria própria

A tabela final de avaliação é obtida com a média simples entre as notas dadas para todos os pares de entidades em relação a cada um dos parametro relacionados (alinhamento, descrição e estrutura). Para a criação de uma tabela final de sumarização de notas, somamos todas as notas dadas aos conjuntos de entidades e dividimos pelo total de conjuntos analisadas. Cita-se como exemplo: colocamos a nota dos parâmetros (alinhamento, descrição e estrutura) como sendo A, sua média como sendo M e o número de conjuntos avaliados como sendo C, entao tem-se que o resultado final é:

$$\mathbf{M} = \frac{\mathbf{1}}{\mathbf{C}} \sum_{\mathbf{n=0}}^{\mathbf{C}} \mathbf{A}_{\mathbf{n}}$$

para cada um dos parâmetros (alinhamento, descrição e estrutura) aplicados.

A TAB 11 é preenchida com o resultado da média de cada um dos parâmetros:

TABELA 11
Compilação Final de Notas

	Ontologia 1	Ontologia 2

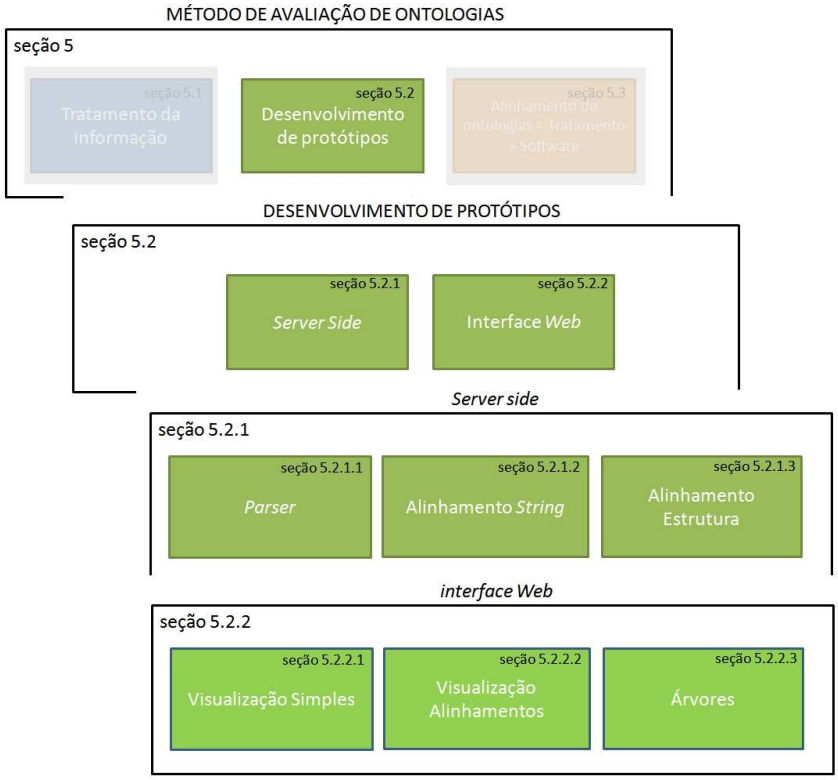
Corretude Alinhamento	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de cada alinhamento.	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de cada alinhamento.
Corretude Estrutura	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota de corretude de estrutura de cada conjunto de entidades
Corretude Descrição	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota dada a corretude de descrição para cada conjunto de entidades	$M = \frac{1}{C}\sum_{n=0}^{C} A_n$, na qual A é a nota dada a corretude de descrição para cada conjunto de entidades
Avaliação Final	$M = \frac{1}{3}\sum_{n=0}^{3} A_n$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela	$M = \frac{1}{3}\sum_{n=0}^{3} A_n$, na qual A são as nota obtidadas nos tres campos acima desta mesma tabela

Fonte: Autoria própria

5.2 – Desenvolvimento de Protótipos

Dentro desta seção descreveremos como foi construído o *software* de apoio ao método proposto neste trabalho. Começaremos com a explicação das ferramentas de tradução da linguagem semiestruturada OWL, descreveremos os algoritmos de *matching* utilizados e por fim descreveremos a ferramenta de visualização. Desta forma. o item 5.2.1 traz a descrição do *parser* utilizado; o item 5.2.2 descreve os algoritmos de *matching* criados; 5.2.3 o protótipo de visualização implementado e 5.2.4 os testes realizados. A FIG. 23 traz de forma esquemática como se deu o desenvolvimento desta seção.

FIGURA 13 - Esquema de Desenvolvimento de Protótipos



Fonte: Autoria Própria

5.2.1 – *Parser*

Aqui descreveremos o desenvolvimento dos sistemas para realização dos *parsers* nas ontologias em tratamento. A escolha de uma descrição detalhada é feita para permitir a possibilidade de implementações futuras, independentes dos produtos desenvolvidos durante essa pesquisa. Item 5.2.1.1 traz a descrição da Biblioteca *libxml2*; o item 5.2.1.2 descreve a ampliação da biblioteca *libxml2*; 5.2.1.3 as formas que são utilizadas para gerar a estrutura para visualização.

5.2.1.1 - Biblioteca *libxml2*

A biblioteca referenciada é utilizada para a quebra da ontologia em nós: o arquivo OWL é carregado através da interface web e enviado como arquivo texto para tratamento prévio. Isso é necessário, pois a biblioteca não trata alguns caracteres especiais dentro das *tags*. Então torna se necessário tratar *namespaces* e endereços de internet contidos dentro dos modelos. Por exemplo:

```
<owl:Class rdf:about="http://purl.obolibrary.org/obo/IAO_0000027#">
      <rdfs:subClassOf rdf:resource="http://purl.obolibrary.org/obo/IAO_0000030"/>
      <obo:IAO_0000119>OBO:imported &amp;quot;http://purl.obolibrary.org/obo/iao.owl&amp;quot;
      </obo:IAO_0000119>
      <obo:IAO_0000115 xml:lang="en">
          a data item is an information content entity that is intended to be a truthful statement …
     </obo:IAO_0000115>
</owl:Class>
```

Nesse caso alguns, caracteres devem ser tratados. Os mais comuns são “:” e ”#”. Para o tratamento esses e outros caracteres foram substituídos por identificadores específicos, necessários para reconstrução da ontologia na exibição e na busca automática de informações extras ao arquivo da ontologia. Como resultado do tratamento, teríamos:

```
<owl?DOT Class rdf:about="http?DOT//purl.obolibrary.org/obo/IAO_0000027?CHARP">
      <rdfs?DOT subClassOf rdf?DOT
resource="http?DOT//purl.obolibrary.org/obo/IAO_0000030?CHARP "/>
      <obo?DOT IAO_0000119>OBO:imported &amp;quot;http://purl.obolibrary.org/obo/iao.owl&amp;quot;
      </obo?DOT IAO_0000119>
      <obo?DOT IAO_0000115 xml:lang="en">
          a data item is an information content entity that is intended to be a truthful statement …
     </obo?DOT IAO_0000115>
</owl?DOT Class>
```

Um algoritmo simples de busca e mudança desses caracteres foi implementado. Ele basicamente busca os caracteres através de pesquisa por expressões regulares e os troca

pelos identificadores. No caso “:” é representado por “?DOT” e “#” é representado por “?CHARP”.

Após esse tratamento a biblioteca libxml2 transforma a ontologia em uma coleção de objetos. As limitações da biblioteca, a qual não é feita para owl, faz com que todas as *tags* que contêm abertura “<owl:class>” e fechamento “</owl:class>” sejam representadas como um objeto. Isso torna inviável o *parser* direto através da biblioteca. Desta forma é necessário se ampliar a biblioteca para recuperar corretamente as relações de hierarquia e afiliação do arquivo owl sob tratamento.

5.2.1.2 - Ampliação da biblioteca

Depois da transformação do arquivo owl para uma lista de objetos é necessário se ampliar a biblioteca para que seja possível a análise interna das classes descritas dentro do documento owl. Foi necessário implementar formas de buscas mereológicas e formas de obter as informações de cada entidade modelada, por exemplo como: descrição, propriedades, relacionamentos, cardinalidades, etc. Para buscar informações dentro dos objetos vindos de um OWL foram implementadas as seguintes características dentro do *parser*: buscas de subclasses e busca de propriedades, os quais são detalhados a seguir.

Busca subclasses

Para criar uma estrutura que mapeia as relações entre classes e subclasses foi utilizada uma busca linear. A informação sobre a relação “subClasseDe” de uma entidade está contida dentro da própria entidade. Isto torna necessário verificar a maioria das entidades modeladas para se alcançar o mapeamento total dessas relações. Por esse motivo a escolha da busca linear.

Para a efetivação desta busca foi necessário utilizar formas que tornassem a pesquisa dentro das classes mais rápida. Escolhemos uma estratégia de criação de quatro dicionários de objetos que são preenchidos ao longo da leitura sequencial das classes da ontologia e, para cada entidade anotada, a retiramos da lista de objetos.

Para cada classe anotamos se ela tem a construção *subclassof, c*aso não contenha, a inserimos (a classe que não contêm a construção subclassof) dentro de um dicionário voltado para armazenar “classes pai”, ou seja, classes que não subclasses de nenhuma outra.

No caso de uma ontologia ser baseada em uma ontologia de “alto nível” esse dicionário permanece vazio até o término do processamento dos outros dicionários e partir deles montamos a TAB 12.

TABELA 12
Estrutura de dados *Parser*

Dicionário de Objetos Classe Pai			Objetos
Nome	Objeto	Nomes SubClasses	
Entity	<owl:Class rdf:about=”Entity">	[sub1; sub2; sub3]	<owl:Class rdf:about="&bfo;Entity"> <rdfs:label rdf:datatype="&xsd;*string*"> Entity </rdfs:label> <owl:unionOf rdf:parseType="Collection"> <owl:Class rdf:about="&snap;Continuant"/> <owl:Class rdf:about="&span;Occurrent"/> </owl:unionOf> </owl:Class>

Fonte: Autoria própria

No caso acima, temos o mapeamento da ontologia BFO, na qual *Entity* é a primeira classe considerada, o pai de todas as outras classes. Esse dicionário tem a coleção de subclasses, neste caso o campo de subclasses é preenchido após a construção dos outros dois dicionários.

Na segunda interação na lista de objetos temos a construção de um segundo dicionário que contem o nome da classe, o objeto em si e o nome de sua superclasse. Para isso verifica-se o nome da superclasse dentro da *tag* que contem a informação *subClassOf*. A tabela de classes pai é consultada e caso o nome da superclasse esteja presente nela o objeto é excluído da lista, caso contrário, não. Caso o dicionário pai esteja preenchido neste novo dicionário só existirão os filhos diretos das classes pai; caso contrário, essa redução é feita após o cruzamento dos outros dicionários.

TABELA 13
Estrutura de dados *Parser* 2

Dicionário de Objetos Classe			Lista Objetos
Nome	Objeto	Nomes SuperClasses	
Continuant	<owl:Class rdf:about="&snap;Continuant">	[Entity]	<owl:Class rdf:about="&snap;Continuant"> <rdfs:subClassOf rdf:resource="&bfo;Entity"/> . . . </owl:Class>
Occurrent	<owl:Class rdf:about="&span;Occurrent">	[Entity]	<owl:Class rdf:about="&span;Occurrent"> <rdfs:subClassOf rdf:resource="&bfo;Entity"/> . . . </owl:Class>

Fonte: Autoria própria

No caso acima também temos o mapeamento da ontologia BFO, na qual *Entity* é a primeira classe considerada. Os filhos diretos dela são *Continuant* e *Ocurrent* e, nesse caso, eles preenchem o dicionário e são retirados da lista de objetos. Esse dicionário tem a coleção de superclasses, neste caso o campo de superclasses é preenchido através dos dados contidos no campo *subClassOf* modelado o qual, no caso das duas classes em evidência na TAB. 13, é a classe *Entity*.

A partir da terceira interação na lista de objetos temos a construção do terceiro dicionário que contém também o nome da classe, o objeto e o nome de sua superclasse. A diferença a partir deste ponto é que as classes são retiradas da lista de objetos independente de qualquer possibilidade de existência anterior, uma vez que são guardadas no dicionário e a busca de informações de forma rápida será feita a partir das informações contidas no dicionário. Para isso, verifica-se o nome da superclasse dentro da *tag* que contém a informação *subClassOf*. Aqui os dicionários, construídos nas etapas anteriores, não são consultados, pois a construção do terceiro dicionário não os leva em conta.

TABELA 14
Estrutura de dados *Parser* 3

Dicionário de Objetos Classe			Lista Objetos
Nome	Objeto	Nomes SuperClasses	
ProcessualEntity	<owl:Class rdf:about="&span;ProcessualEntity">	[Ocurrent]	<owl:Class rdf:about="&span;ProcessualEntity"> <rdfs:subClassOf rdf:resource="&span;Occurrent"/> . . . </owl:Class>
DependentContinuant	<owl:Class rdf:about="&snap;DependentContinuant">	[Continuant]	<owl:Class rdf:about="&snap;DependentContinuant"> <rdfs:subClassOf rdf:resource="&snap;Continuant"/> . . . </owl:Class>
. . . .	. . . .	. . . .	. . . .
Disposition	<owl:Class rdf:about="&snap;Disposition">	[RealizableEntity]	<owl:Class rdf:about="&snap;Disposition"> <rdfs:subClassOf rdf:resource="&snap;RealizableEntity"/> . . . </owl:Class>

Fonte: Autoria própria

A TAB. 14 mostra o mapeamento geral das classes da ontologia BFO que possuem superclasses. Nessa estrutura estão contidas todas as classes a partir do terceiro nível da ontologia. Este dicionário será utilizado junto aos anteriores para desenvolvimento do quarto dicionário o qual é chamado de dicionário objeto-objeto. Ele mapeia toda a estrutura da ontologia a partir do cruzamento das estruturas criadas nas três fases anteriores.

O quarto passo desta etapa da ampliação da biblioteca utilizada é feito através do cruzamento dos dois passos anteriores. Desta forma conseguimos gerar a estrutura objeto-objeto que mapeia toda a ontologia através da ligação de objetos pais a objetos filhos e estes são constituídos de dicionários de objetos contendo todos os filhos dos objetos pai. Desta

forma, segue-se de forma recursiva até o que consideramos a folha de uma árvore gerada, ou seja, até a entidade modelada que não tem mais subclasses.

Para se alcançar essa última estrutura, seguimos os seguintes passos de processamento das informações já estruturadas:

- Comparação entre a terceira estrutura e ela mesma: uma busca linear ocorre na terceira estrutura comparando-se o campo superclasse de cada classe armazenada com o campo nome de todas as outras classes dentro da estrutura. Para cada nome encontrado, um novo registro é colocado na quarta estrutura (objeto-objeto). Desta forma temos uma primeira camada estrutural com dois níveis na quarta estrutura, ou seja, temos uma camada que consiste de uma superclasse referenciando suas subclasses como na TAB,15. Cada registro contém o nome, superclasse e um objeto (no campo que representa a lista de objetos), isso faz com que ela se torne uma estrutura recursiva

TABELA 15
Primeira camada estrutural da quarta estrutura

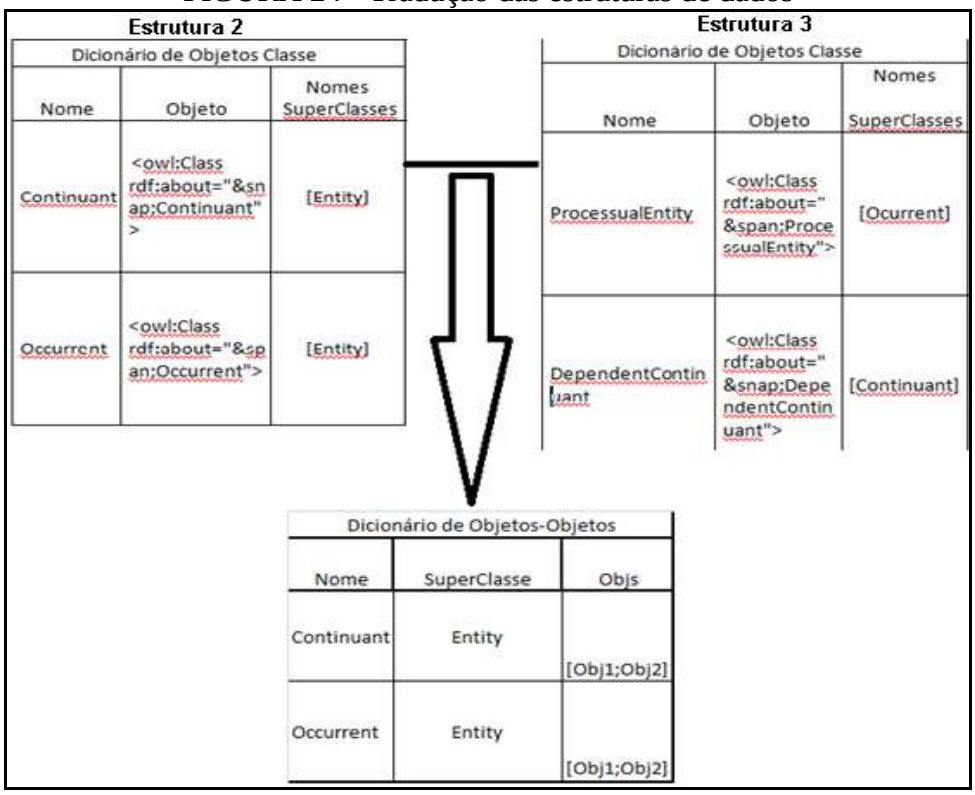
SuperClasse	SubClasse
Continuant	DependentContinuant
Ocurrent	ProcessualEntity

Fonte:

- Uma busca recursiva é feita na quarta estrutura para se reduzir ao mínimo o número de dados redundantes. Esta busca tem como suporte a estrutura três. Assim, fazendo comparações com os dados de superclasse já determinados, diminuímos o número de interações necessárias. Para cada classe buscamos suas subclasses ou superclasses, na própria estrutura e na estrutura três. Quando uma dessas é encontrada o campo é reorganizado armazenando-se a nova sub ou superclasse no campo encontrado e apagando o campo anterior. Isso faz com que a estrutura se mantenha reduzida e não tenhamos mais de uma cópia desnecessária de uma classe;
- Mais uma busca é feita na primeira camada do dicionário objeto-objeto para garantir a não repetição de dados nesta camada;
- Neste ponto estamos com uma estrutura passível de comparação com a segunda estrutura. Assim fazemos uma busca pelos nomes contidos no campo superclasse

do quarto dicionário na estrutura dois. Para cada nome encontrado é criado um novo registro com os dados das duas estruturas envolvidas na busca, no entanto nesse caso adicionamos o registro dono do campo superclasse procurado na lista de objetos do novo registro criado. Assim que isso é feito o registro antigo é apagado.

FIGURA 24 - Tradução das estruturas de dados



Fonte: Autoria própria

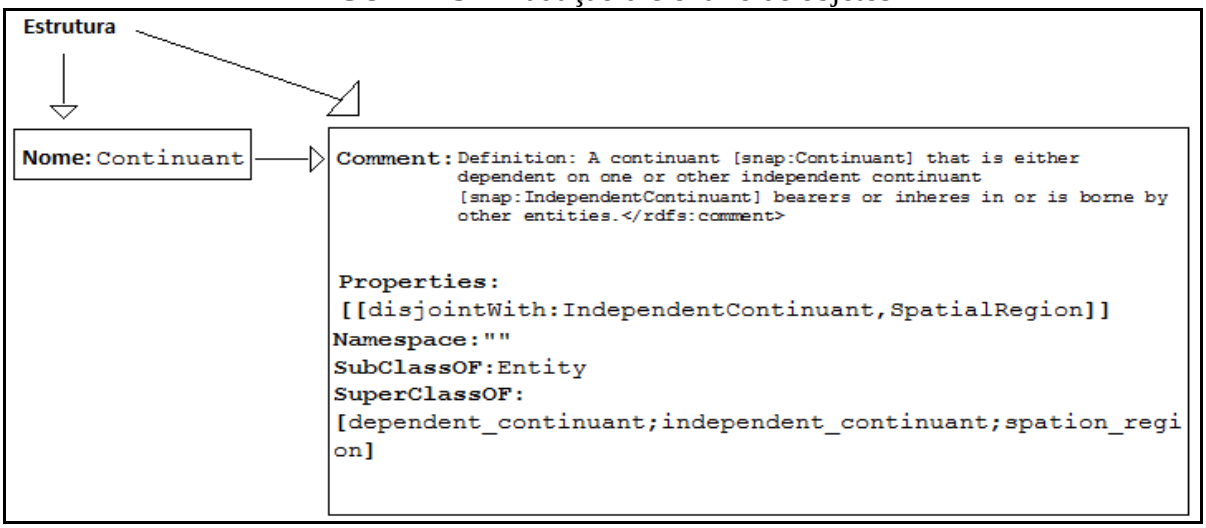
Nesse ponto, temos um conjunto de dados modelados de forma a dar suporte à criação do dicionário necessário para a visualização da ontologia. Um algoritmo simples de transformação é aplicado. Assim cada classe existente é reescrita dentro do novo modelo utilizado pelas bibliotecas de visualização utilizadas.

Busca propriedades

Uma estrutura representativa de um objeto foi criada para a representação das propriedades de cada entidade da ontologia. Essa estrutura gera um objeto com os dados das entidades já estratificados, ou seja, nós níveis da ontologia. Um dicionário representativo é utilizado para armazenar esses dados. Nele são armazenados o nome da entidade e o objeto criado com as informações sobre ela. Dentro da estrutura os seguintes dados estão contidos:

- *Comment;*
- *Properties*: um dicionário de dados, comumente representado por "["+"nome:descrição"+"]" FIG. 25, com nome e descrição de cada propriedade; desta forma pode acrescentar uma quantidade sem limites de propriedades;
- *Namespace;*
- *Subclassof;*
- *Superclassof.*

FIGURA 25 - Tradução dicionário de objetos



Fonte: Autoria própria

Essa estrutura representa um dicionário de dados que tem como chave o nome da classe e como conteúdo um objeto com as informações supracitadas. Ela é utilizada como base para o desenho final da lista de dados para transformação da ontologia em uma visualização gráfica e como base para alinhamentos baseados em características internas de cada entidade.

5.2.1.3 - Geração de estrutura para visualização

Aqui descreveremos os passos para geração dos mapeamentos de dados necessários para tradução da ontologia em uma metáfora visual. Neste trabalho foi utilizada

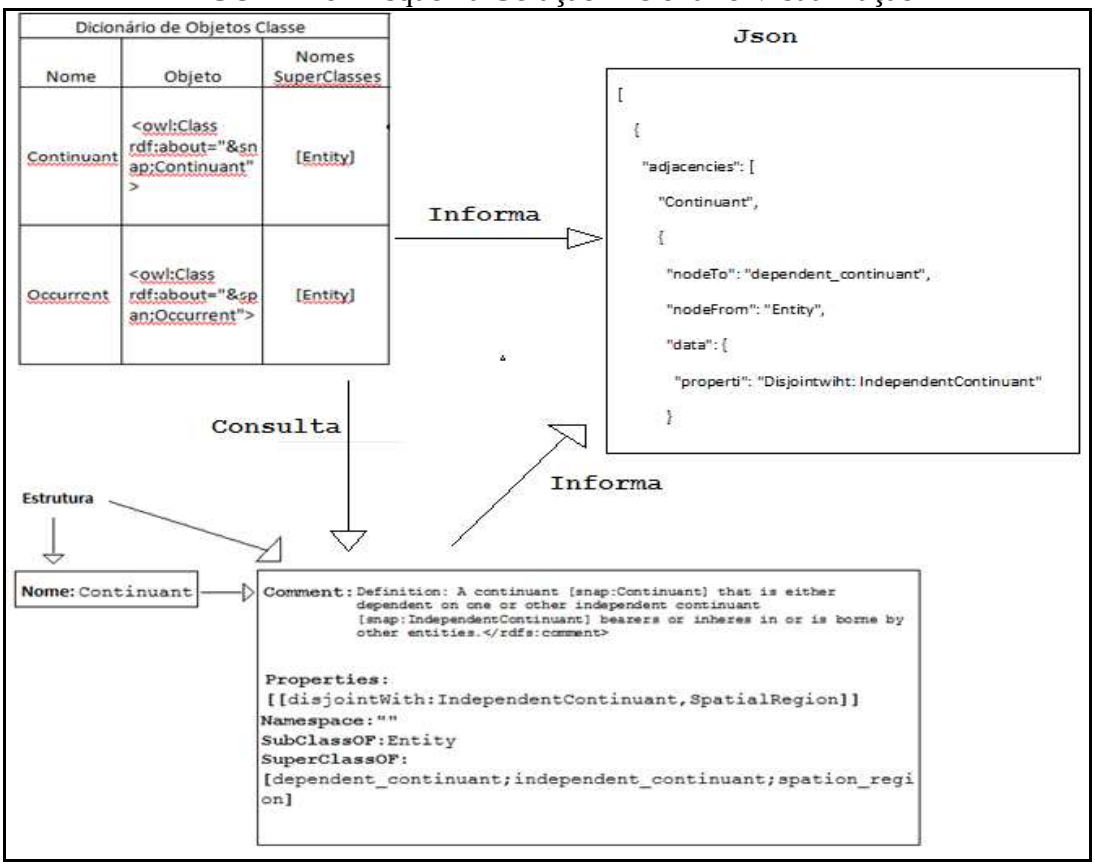
como base para a visualização a biblioteca web *infovistoolkit* que utiliza a estrutura *json* para modelar os dados que serão visualizados.

Três estruturas (a classe-pai, objeto-objeto e propriedades) criadas nos passos de utilização da *libxml2* e ampliação da biblioteca são utilizadas através do cruzamento das informações contidas nas três. Um sistema de varredura começa a criação pelo dicionário classe-pai e depois passa para o dicionário objeto-objeto. A varredura consulta a estrutura propriedade para cada classe.

Para cada classe que é lida nas estruturas de dados, um mapeamento da entidade é criado dentro do dicionário *json* FIG. 26 utilizado pelo *infoviztoolkit*. Isso acontece da seguinte forma:

- O algoritmo lê a classe nos dicionários de classe gerados e a transcreve para a estrutura *json;*

FIGURA 26 - Esquema Geração Dicionário Visualização



Fonte: Autoria própria

- Para cada classe a estrutura propriedades é consultada e transcrita para a área "data" dentro do *json;*
- Após essa fase inicial, consulta-se a estrutura objeto-objeto através de uma busca

iterativa por todas as classes; o processo é o mesmo quando explicado em alto nível, só se diferenciando na implementação específica para perpetuar a busca dentro de estruturas fractais de objetos;

- Isso é feito repetidamente até o termino de modelagem de todas as classes.

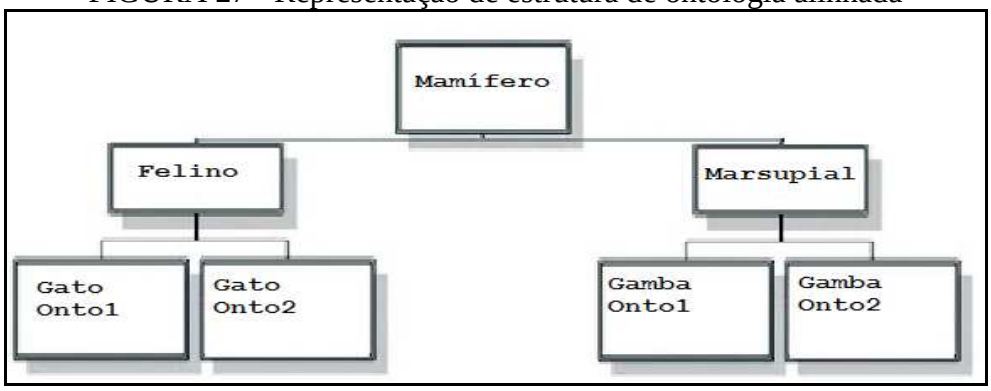
Ao término desse processo, o resultado é uma estrutura completa pronta para ser transformada em uma metáfora visual. Daqui em diante, explicaremos os algoritmos de *matching*, a construção da parte gráfica, roteiro de alinhamento e método de avaliação de ontologias baseado no tipo de alinhamento proposto.

5.2.2 – Algoritmos de *Matching*

Esta seção se destina a descrever os algoritmos propostos para o suporte ao roteiro de *matching* semiestruturado desenvolvido. Estes são descritos em alto nível, de forma detalhada, para que seja possível sua implementação utilizando essa seção como guia.

Ao fim do procedimento de *matching* temos um novo modelo que contempla o resultado do alinhamento das duas ontologias de entrada. Cada dupla de entidades será colocada abaixo de sua classe pai teórica. Cada uma das entidades será identificada de acordo com sua ontologia de origem FIG. 27.

FIGURA 27 - Representação de estrutura de ontologia alinhada



Fonte: Autoria própria

Iniciaremos com a descrição dos algoritmos baseados em comparação de lista de caracteres (*string*), em seguida descreveremos os baseados na estrutura da ontologia e por final descreveremos um algoritmo que une as duas técnicas aplicadas. Na seção 5.2.2.1

descreve-se *Matching* baseado em string, e na secção 5.2.2.2 descreve-se *Matching* baseado em estrutura.

5.2.2.1 *Matching* baseado em *string*

Inicialmente, as estruturas geradas no passo anterior são importadas. Em cima destas os processos e algoritmos de *matching* serão aplicados para que possamos gerar uma nova ontologia com o mapeamento das duas ontologias utilizadas na aplicação do método.

Uma função chamada *matchingStringName* utiliza os seguintes algoritmos de comparação:

- Exata que, como já descrito, faz a comparação de duas cadeias de caracteres e retorna positivamente somente se essas forem exatamente iguais;
- Exata "moderada": algoritmos que ignoram maiúsculas e minúsculas que, comparam as cadeias de caracteres da mesma forma do algoritmo anterior, mas ignora a caixa dos caracteres;
- Substituição: algoritmos que substituem caracteres como letras acentuadas por suas formas básicas;
- "abertos": algoritmos que aceitam caracteres diferentes à direita, à esquerda ou no meio da *string*: algoritmos que desconsideram pequenas diferenças entre os *strings*.

Cada um dos tipos de busca de caracteres descritos é aplicado numa etapa do *matching*. Para a utilização de tais algoritmos de casamento de caracteres utilizamos bibliotecas de busca em texto já existentes dentro da linguagem *Python*, não sendo dessa forma necessário reimplementá-los. Assim, para cada tipo de busca e subsequente comparação dentro das ontologias, é aplicado um método dessa biblioteca. Essas aplicações geram uma lista de entidades semelhantes e um grau de semelhanças, que chamamos aqui de "equivalência", o qual é estabelecido entre os nomes listados. À medida que as comparações são feitas e, de acordo com a flexibilidade de comparação do algoritmo, o nível de equivalência é mudado, ou seja, os níveis de equivalência irão ser alterados de acordo a tolerância a diferenças de cada algoritmo de comparação de *strings* aplicado: quanto maior a

tolerância maior o nível de equivalência entre todos os tipos de classe; já quanto menor o nível de tolerância somente entidades similares terão altos níveis de equivalência.

Estipulamos aqui cinco os graus de semelhança aplicados de acordo com a busca de cada um dos tipos de casamento: baixa, razoável, média, boa e ótima. A equivalência leva em conta o número de letras iguais e suas respectivas ordens, por exemplo, entre os nomes “Joel” e “Jor-El” a comparação de equivalência é feita em cima no número de letras da maior palavra, então temos 4 letras iguais em 6 caracteres que nos dá uma porcentagem aproximada de 66% de equivalência. No entanto a ordem das letras é de 2 iguais em 6 possibilidades o que dá aproximadamente 33% de equivalência. Isso torna a equivalência total aproximada 49%. Esse nível de equivalência é considerado pelo sistema como médio, pois para cada espaço de porcentagem ele dá uma nota de equivalência diferente: 10 a 20% baixa; 20 a 40% razoável; 40 a 60% média; 60 a 80% boa e por fim de 80 a 100% ótima.

Iniciamos com os algoritmos de comparações abertas, criando uma lista inicial de entidades semelhantes. Nesse caso a equivalência é, por varias vezes, considerada ótima. Desta forma obtém-se uma lista que pode ser exemplificada conforme segue na TAB.16;

TABELA 16
Retorno do sistema de alinhamento baseado em nome.

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinos
Gato	Ótima	Gato
Leao	Ótima	Leões
Tigre	Ótima	Tiges
Leo pardo	Ótima	Leopardo
J aguatirica	Ótima	Jaguatirica
Onça	Ótima	Onca

Fonte: Autoria própria

Na segunda fase é aplicado o algoritmo de substituição e temos mais um refinamento (TAB. 17).

TABELA 17
Retorno do sistema de alinhamento baseado em nome 2ª fase

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinos
Gato	Ótima	Gato
Leao	Ótima	Leões
Tigre	Baixa	Tiges
Leo pardo	Média	Leopardo
J aguatirica	Média	Jaguatirica
Onça	Ótima	Onca

Fonte: Autoria própria

Dessa maneira, continua-se com os refinamentos até que todos os tipos de algoritmos de comparação sejam aplicados (aberta, substituição, exata “moderada”, exata). No caso do exemplo teríamos o seguinte resultado (TAB. 18).

TABELA 18

Resultado do processo de alinhamento baseado nome

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinos
Gato	Ótima	Gato
Leao	Boa	Leões
Tigre	Baixa	Tiges
Leo pardo	Baixa	Leopardo
J aguatirica	Baixa	Jaguatirica
Onça	Boa	Onca

Fonte: Autoria própria

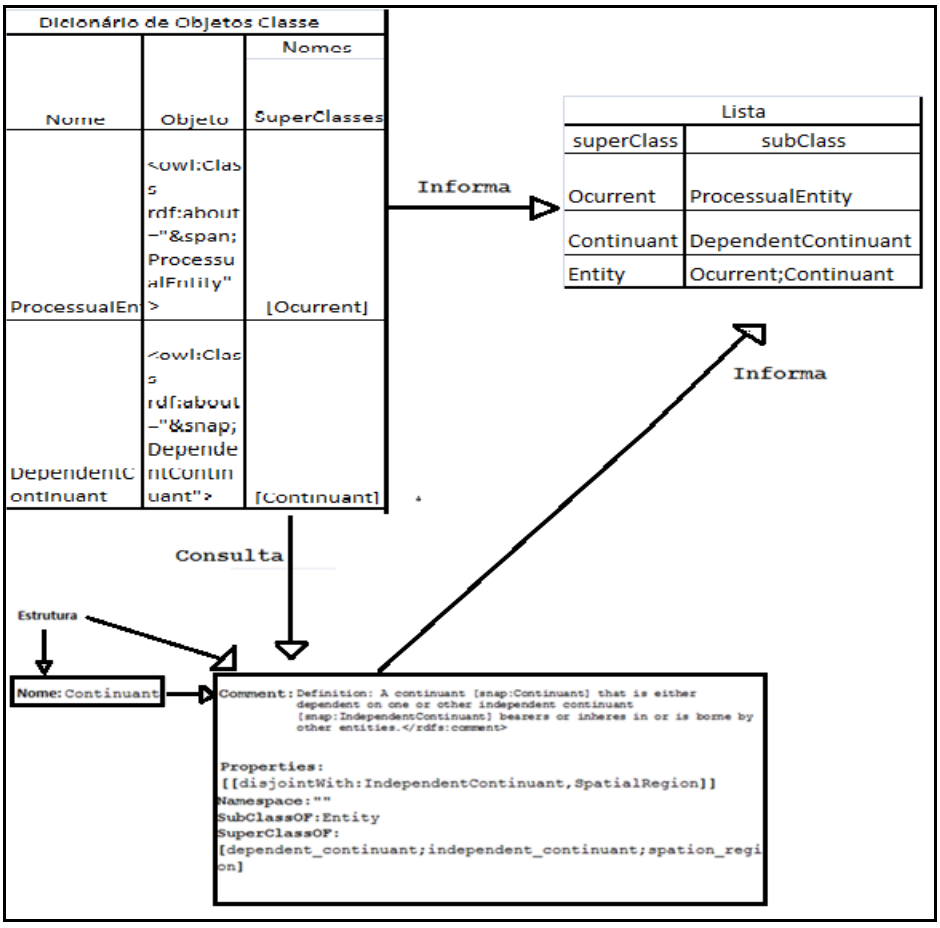
A partir desta lista final construímos uma “meta-ontologia”, já citada no início da seção 5.2.2, com cada dupla de entidades modeladas. Para cada dupla adicionamos as informações das entidades o quão elas são semelhantes. No caso de duas ontologias com nomes diferentes, a nova ontologia será identificada com a soma do nome das duas.

5.2.2.2 - *Matching* baseado em estrutura

Para a implementação desta técnica de *matching* consideramos a união de análise *subclassof* com o casamento de caracteres, ou seja, consideramos a análise das superclasses das entidades junto a uma comparação de caracteres que, no caso, além do nome, é aplicada às suas descrições.

Inicialmente, aplicamos uma busca dentro das estruturas das duas ontologias geradas no *parser* para criação de uma lista de todas as classes e suas respectivas superclasses. Após a lista criada, aplicamos dois algoritmos distintos: compara_superclasse e compara_subclasse. Os dois utilizam os algoritmos de casamento de caracteres, expandidos, para auxiliar nas buscas.

FIGURA 28- Construção dicionário fase estrutura



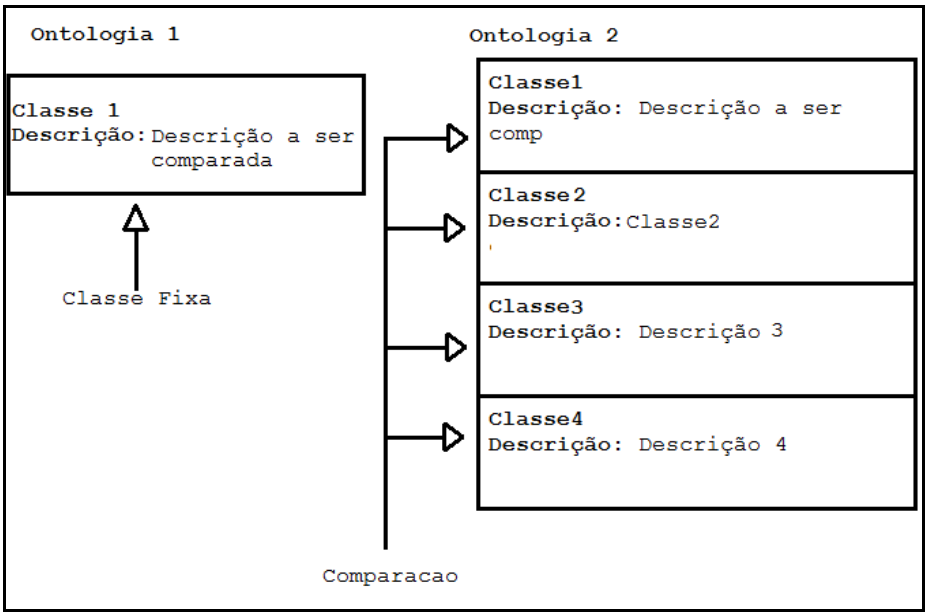
Fonte: Autoria própria

O casamento de caracteres aqui é utilizado em duas comparações distintas: nomes e descrições. Na primeira as descrições são comparadas. Para comparação das descrições

temos um acréscimo de complexidade para se considerar níveis de similaridades. Já na segunda a comparação de nomes abertas é feita no campo *labels* das entidades pai.

No primeiro caso de comparação, o qual leva em conta as descrições, buscamos, para cada entidade que tem uma superclasse, a descrição da superclasse. Fixamos uma descrição e fazemos uma comparação dessa descrição com as descrições na segunda ontologia de forma sequencial (FIG. 29).

FIGURA 29 - Representação das comparações de descrições



Fonte: Autoria própria

A comparação de forma sequencial tem o objetivo de criar uma lista de superclasses candidatas a terem seus filhos alinhados. Para a criação dessa lista levamos em conta a proximidade de suas descrições, dividindo essa proximidade em níveis de similaridade de acordo com porcentagem de equivalência. Esse nível é calculado em relação à descrição fixada. Para comparação, calculamos a porcentagem de palavras similares, o que é feito através da contagem de palavras da classe fixa e, em relação ao numero total de palavras, calculamos a porcentagem de palavras semelhantes nas descrições das classes comparadas. Os níveis são os seguintes.

TABELA 19
Níveis de similaridade aplicados pelos algoritmos de estrutura

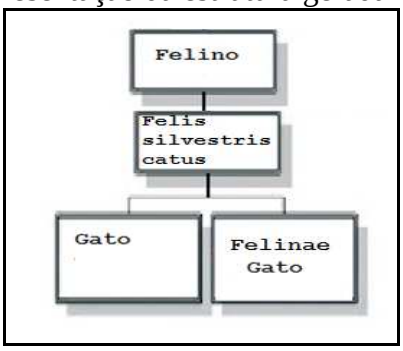
Níveis de Similaridade	
% de Equivalência	**Similaridade**
10 a 30	Ruim
30 a 50	Baixa
50 a 70	Media
70 a 90	Boa
90 a 100	Ótima

Fonte: Autoria própria

Uma estrutura com os resultados dessa equiparação é criada, e nela estão os nomes das classes pai, as subclasses e seu grau de similaridade de acordo com a tabela acima. Isso é feito para todas as superclasses listadas nas estruturas iniciais (estrutura 1 e 2 da fase *parser*). Desta forma, temos uma lista preliminar de entidades que terão seus filhos alinhados e, consequentemente, serão alinhados.

Nesta etapa, uma semi ontologia (FIG. 30) é gerada com as informações adquiridas na primeira rodada de alinhamento estrutural. Para cada par de superclasses todos os filhos são importados das estruturas que contêm os dados da primeira ontologia. Os filhos tem seu nome alterado para “[nome Entidade] +[nome da ontologia]”. Desta forma, temos uma marcação de como as entidades foram modeladas.

FIGURA 30 - Representação da estrutura gerada na 1ª fase estrutura



Fonte: Autoria própria

A segunda etapa é aplicada em cima da estrutura gerada pela comparação de descrições da seguinte forma:

- O nome das entidades filhas é comparado aplicando-se as regras de similaridade mencionadas na seção *matching* baseado em *string*;
- Valor de similaridade baseado nas descrições é consultado, e é feita uma relação entres as duas métricas de similaridade;
- Caso exista um alto grau de similaridade em nome junto a um alto grau de similaridade de descrições não há modificações;
- Já no caso de baixa similaridade de nomes e alta similaridade de descrições o nível de equivalência dado é modificado para baixo;
- Caso o nome e descrição sejam baixos o alinhamento naquele ponto é descartado (FIG. 31);

FIGURA 31 - Representação da aplicação das métricas feitas pelo algoritmo

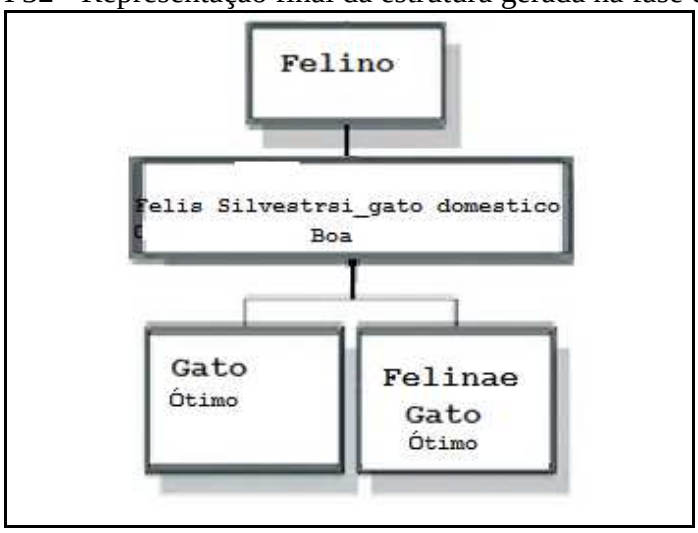
Niveis de Similaridade	
% de Equivalencia	**Similaridade**
10 a 30	Ruim
30 a 50	Baixa
50 a 70	Media
70 a 90	Boa
90 a 10	Otima

Ontologia 1 - Felinos	Equivalência	Ontologia 2 - Felinos
Gato	Ótima	Gato
Leao	Ótima	Leões
Tigre	Ótima	Tiges
Leo pardo	Ótima	Leopardo
J aguatirica	Ótima	Jaguatirica
Onça	Ótima	Onca

Fonte: Autoria própria

- Para cada nível abaixo das relações de equivalência baseada em nome, a marcação de equivalência baseada em descrição desce um grau;
- Uma nova estrutura que representa uma semi ontologia (FIG. 32) é criada;
- No campo *label* é somado junto ao nome das duas entidades o grau de equivalência entre as duas;
- Os nomes das entidades pai são comparados e também valorados em termos de equivalência, uma marcação no campo *label* é colocada para explicitar esse grau;

FIGURA 32 - Representação final da estrutura gerada na fase estrutura



Fonte: Autoria própria

- A estrutura é escrita em uma linguagem de representação de ontologias.

Às novas semi ontologias geradas, é aplicado o *parser* para que se possa criar a tradução de dados necessária para a visualização. Nas próximas seções serão explicadas as interfaces gráficas, roteiros de alinhamento e avaliação e a junção em um único método.

5.2.3 – Protótipo de Visualização

Nesta seção descreveremos a construção da interface gráfica do *software* de apoio. Começaremos pela escolha das técnicas de visualização e após, descreveremos a construção. Na seção 5.1.3.1 descreve-se as técnicas de visualização selecionadas e na seção 5.1.3.2 descreve-se a construção do protótipo.

5.2.3.1- Técnicas de Visualização selecionadas.

Uma ontologia é estruturada em classes, subclasses, atributos, instâncias e relações. Essa estrutura permite que representação através de grafos abertos (árvores), os quais são passíveis de visualização. No entanto, para um grande volume de dados, a

visualização e navegação através dos itens de uma ontologia pode se valer de um sistema de árvore dinâmico para ajudar no processo.

Desta forma, selecionaram-se algumas técnicas de visualização em árvore que permitem a navegação pelos itens de uma ontologia. Dentre as várias técnicas pesquisadas foram selecionadas algumas técnicas baseadas em *árvore hiperbólica* e árvore radial, pois estas permitem a visualização dos dados que crescem de forma exponencial (MUNZNER, 1998). Mais especificamente, as visualizações em *árvores hiperbólicas* e radiais permitem visualizar mais de uma direção de dimensões dentro de um espaço delimitado por uma superfície esférica, onde os nodos folhas são representados por pirâmides das relações existentes. Desta forma, é possível ver maior número de classes, instâncias, e relações.

Esta técnica, junto com elementos de atualização de dados dinâmica, permitirá que os dados contidos em uma ontologia possam ser organizados na forma de um grafo que se expandirá a medida em que os novos dados são requisitados durante a navegação.

5.2.3.2 – Construção

No desenvolvimento do protótipo foram utilizadas as linguagens de programação Python, C++ e *Javascript*, e como o servidor de aplicativos, utilizou-se Zope. O protótipo foi dividido em dois módulos distintos e independentes: um módulo local, de tratamento e pesquisa direta na ontologia; e uma interface web, responsável por transformar a estrutura em uma árvore. Os módulos internos já foram descritos nas seções 5.1.1 e 5.1.2, de forma que, nessa seção, somente descreveremos a interface web e seus suportes diretos.

A interface web incrementa a interação entre o participante e a ferramenta e foi dividida em três módulos: busca, apresentação e árvore (a visualização em si). O módulo *web* contará com ferramentas disponibilizadas pelo servidor de aplicativos Zope e Java, e sua aparência será definida por um CSS.

Esse módulo utiliza duas bibliotecas; uma de representação em árvores indentadas e uma de visualização já descrita (*infoviztoolkit)*. Na primeira, um dicionário direto é utilizado. Ele é montado como uma lista de dados que são interpretados pela biblioteca *Dynatree* que os transforma em uma cadeia de imagens identadas. A segunda utiliza o dicionário *json* gerado na fase de parser dos sistemas.

Normalmente, uma ontologia pode ser considerada pesada, em termos computacionais, para ter toda sua visualização processada de uma única vez. Então para contornar esse problema utilizamos busca dinâmica de dados e, para execução desta busca dinâmica, utilizamos requisições assíncronas. Estas são feitas através de *scripts* produzidos para requisitar ao servidor novas informações de acordo com as classes selecionadas.

Quando se clica nas entidades representadas, *scripts* diferentes são ativados para busca de informações diversas. Cada um desses *scripts* gera uma nova parte nos dicionários representados pelas bibliotecas de visualizações e ativam seus métodos de atualização.

São utilizados três tipos de técnicas de visualização e quatro telas de visualizações de dados distintas dentro do protótipo. Todas as visualizações contam com formas extras de exibição dos dados referentes a cada entidade, como *tooltips*, barras laterais com informações sobre filhos, caixas de texto com descrições e propriedades, etc.

Os tipos de técnicas visualização adotadas são:

- Árvore identada: utiliza espaço vertical multifacetado, onde encadeia as entidades da ontologia. O sistema constrói a visualização das classes e subclasses adjacentemente aos meios de se ver os dados de cada classe;
- Árvores radiais: o nó raiz é localizado no centro da figura e os outros nós são organizados em níveis circulares em volta do ponto central. Não necessariamente o nó raiz estará no centro da figura, pois dentro das interfaces é possível se navegar pela movimentação das árvores pela tela;
- Árvore Hiperbólica: exibe inicialmente uma árvore com seu nó raiz no centro. Utilizamos a capacidade de crescer exponencialmente com o seu raio inerente a este tipo de visualização para representação da ontologia. Aliando isso a busca dinâmica de dados temos uma capacidade de exibição de entidades relativamente superior a tipos de visualização euclidiana. A distancia entre as represantações de classes é importante, pois as hierarquias tendem a se expandir exponencialmente com a profundidade, e podem ser definidas no espaço hiperbólico de maneira uniforme. Isso faz com que as distancias entres os itens da ontologia sejam aproximadamente os mesmos independente de sua situação hirerárquica.

5.2.4 – Testes: visualização de ontologias

O método foi aplicado em duas ontologias da área biológica, pois dentro dessa área concentra-se grande parte do volume de desenvolvimento de ontologias. Com a aplicação

do método proposto visou-se mostrar sua aplicabilidade, além de sua consolidação dentro dos padrões de avaliação.

A aplicação seguiu como um estudo de um caso específico de avaliação baseada no alinhamento e visualização de ontologias. Neste trabalho não se visou aplicar em campo, mas sim a criação de um método a ser usado como ferramenta para futuras avaliações e desenvolvimento de ontologias.

5.3 – Alinhamento de Ontologias: Tratamento + *Software*

Descrevem-se na presente seção, os roteiros desenvolvidos durante a pesquisa para a consolidação do método proposto. Aqui o método propriamente dito é descrito. Cada passo do que deve ser feito durante o processo de avaliação, além das métricas e utilização das ferramentas desenvolvidas é descrito nesta seção.

Inicialmente descrevemos o roteiro de alinhamento semiautomático que utiliza técnicas de alinhamento *e* visualização de informação como suporte; em seguida a avaliação das características gerais das ontologias baseado no alinhamento. Na seção 5.3.1 descreveremos a utilização do alinhamento desenvolvido; já na seção 5.3.2 a avaliação que tem o alinhamento como base. A figura 33 traz o esquema visual de como essa seção foi desenvolvida.

FIGURA 33 - Esquema alinhamento de Ontologias

Fonte: Autoria Propria

5.3.1 – Alinhamento

Aqui será descrito o procedimento para o alinhamento semiautomático de duas ontologias utilizando-se as ferramentas desenvolvidas durante o processo. Aqui, o roteiro de alinhamento tem o objetivo de ser suporte ao método de avaliação de ontologias proposto nesse trabalho. O alinhamento vai seguir a ordem de descrição dos algoritmos. Cada uma das fases vai levar em conta os níveis de equivalência estabelecidos na seção 5.1.

No início de cada subseção será colocado um organograma que servirá de guia para o desenvolvimento do processo de avaliação. Cada fase é representada nestes organogramas através de seus nomes.

Para iniciar o processo de alinhamento devem-se carregar duas ontologias candidatas ao alinhamento. Cada uma delas precisa ser carregada separadamente na interface web. As fases do processo podem ser disparadas separadamente, no entanto elas devem ser feitas sequencialmente. A seção 5.3.1.1 traz o guia de alinhamento baseado em *string*; já na seção 5.3.1.2 o roteiro de alinhamento baseado em estrutura.

5.3.1.1 - Baseado em *string*

Aqui será descrito o procedimento para o alinhamento semiautomático de duas ontologias levando-se em conta os alinhamentos baseados em *string.* Para tal alinhamento são utilizadas as ferramentas desenvolvidas durante o processo.

O primeiro passo é carregar duas ontologias para o sistema, isso deve ser feito através da utilização da interface web. Ativa-se, então, através do sistema a primeira fase (FIG. 33) do alinhamento baseado em *string.* O sistema retorna como resultado desta primeira fase a tabela de equivalência VS candidatas (TAB. 4) já preenchida com os pares de palavras e, para cada par, a nota dada automaticamente pelo sistema para seu nível de similaridade. Cada par representa duas classes, uma de cada ontologia, que foram retornadas como candidatas ao alinhamento. Em cima deste resultado deve ser feita uma análise que deve ser baseada no conhecimento que o engenheiro tem do domínio modelado dentro das ontologias.

Após essa análise a tabela de equivalência VS candidatas deve-se ter seu preenchimento finalizado. Nesta tabela deve ser colocado no campo “Candidatas” sim ou não,

isso sinaliza se aquele par deve ou não ser alinhado. A TAB. 20 (abaixo) traz um exemplo de como a tabela de equivalência VS candidatas se apresentaria depois de preenchida.

Como neste caso temos somente altos valores de similaridade, a avaliação não visa retirar os pares de candidatos e sim criar uma marcação de candidatos fracos. Essa análise é simples, no entanto não é dispensável. Ela deve ser feita considerando todos os pares retornados. Por exemplo, na lista apresentada na seção 5.2 sabemos que guepardo e leopardo são diferentes, então já se pode marcar sem a espera das outras listas serem geradas como não candidatas. A marcação dessa fase deve ser feita na tabela equivalência VS candidatas que é gerada pelo sistema, a qual possui campo para tal marcação. Por exemplo, a TAB. 20

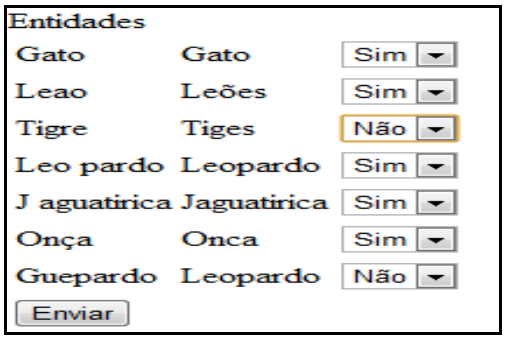
TABELA 20
Equivalência VS candidatas (tabela auxiliar de avaliação)

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinos	Candidatas
Gato	Ótima	Gato	Sim
Leao	Ótima	Leões	Sim
Tigre	Ótima	Tiges	Sim
Leo pardo	Ótima	Leopardo	Sim
J aguatirica	Ótima	Jaguatirica	Sim
Onça	Ótima	Onca	Sim
Guepardo	Ótima	Leopardo	Não

Fonte: Autoria própria

Após o preenchimento da TAB. 20 os resultados devem ser informados ao sistema. Isso é feito através de uma tela do sistema específica para esse fim (FIG.34). O sistema mostra os mesmos pares retornados na TAB. 20 e abre um campo *dropdown* para receber os resultados.

FIGURA 34 - Tela de entrada de dados do sistema desenvolvido



Fonte: Autoria própria

O sistema constrói uma semi ontologia com as informações retornadas e as apresenta ao usuário pelas telas de visualização construídas. A semi ontologia desenvolvida é semelhante à FIG. 35.

A partir da visualização da semi ontologia gerada a TAB. 2 (Sumarização das avaliações fase *string*) deve ser preenchida, e para cada fase uma cópia com os dados dessa fase deve ser mantida. Utilizamos essa tabela como uma tabela preliminar para comparação com as listas que serão geradas nas próximas etapas. Ela é uma das bases para marcação de geração da meta ontologia da ontologia final.

Ativa-se, então, através do sistema a segunda fase do alinhamento baseado em *string* (FIG. 33). O sistema retorna como resultado desta fase a tabela de equivalência VS candidatas (TAB. 4) já preenchida com os pares de palavras e, para cada par, a nota dada automaticamente pelo sistema para seu nível de similaridade. Na segunda fase a nota retornada pelo sistema é mais "severa" por isso as exclusões indicadas por ele devem ser consideradas de forma mais efetiva. Para cada par deve ser feita uma análise levando em conta o conhecimento sobre o domínio do avaliador.

Após essa análise, como na primeira fase, a tabela de equivalência VS candidatas deve ser preenchida. O campo "Candidatas" deve ser preenchido com sim ou não. Da mesma forma que na fase anterior isso marca as candidatas ou não ao alinhamento.

Neste caso temos valores de similaridade mais restritos, então os pares de candidatos já podem ser considerados descartados para o alinhamento. Essa análise é mais profunda que a anterior e já utiliza de forma mais intensa os processos semi-automáticos. Ela também deve ser feita considerando todos os pares retornados. Aqui também podemos exemplificar o processo com a TAB. 20.

Após a finalização das marcações na tabela equivalência VS candidatas (TAB 4) os dados devem ser retornados ao sistema através da tela representada pela FIG 35 (adiante). Com esse retorno o sistema cria uma nova semi ontologias e a apresenta através das telas de visualização. A semi ontologia ficará ao exemplo representado pela FIG, 35.

A partir da visualização da semi ontologia gerada a TAB. 2 (Sumarização das avaliações fase *string*) deve ser preenchida. Utilizamos essa tabela como uma tabela preliminar para comparação com as listas que serão geradas nas próximas etapas. Ela é uma das bases para marcação de geração da meta ontologia final.

Terceira fase alinhamento baseado em *string*

A terceira fase do alinhamento baseado em *string* (FIG. 34) é ativada. Nela percorrem-se os mesmos procedimentos das duas fases anteriores. O sistema retorna como resultado desta fase

a tabela de equivalência VS candidatas (TAB.. 4,) já preenchida com os pares de palavras e, para cada par, a nota dada automaticamente pelo sistema para seu nível de similaridade. Nessa fase a nota retornada pelo sistema é quase dicotômica, pois ele utiliza o método de comparação exata de strings, e assim as exclusões indicadas por ele devem ser consideradas de forma definitiva.

Após a análise, como nas fases anteriores, a tabela de equivalência VS candidatas deve ser preenchida e o campo “Candidatas” deve ser marcado com sim ou não. Aqui, também essa marcação indica de aquele par é ou não candidato.

Nesta fase do processo os valores de similaridade são diretos, então os pares de candidatos devem ser eliminados do alinhamento. Essa análise é mais direta que a anterior e já utiliza de forma plena a parte semiautomática dos processos.

Da mesma forma que nas fases anteriores os dados devem ser retornados ao sistema (FIG. 33) após a finalização das marcações na tabela equivalência VS candidatas (tab. 4). O retorno permite ao sistema gerar a semi ontologia referente à terceira fase de alinhamento baseado em string. A semi ontologia ficará similar ao exemplo representado pela FIG. 35.

Como nas fases anteriores a visualização da semi ontologia gerada nos ajuda a fazer inferências e análises que permite preencher a TAB. 2 (Sumarização das avaliações fase *string*). A tabela também será utilizada como suporte a comparação com os resultados retornados pelos próximos pares.

5.3.1.1.1 - Sumarização das notas de Avaliação fase baseado em *string*

Após a marcação como candidatas ou não de cada uma das duplas de entidades nas fases anteriores, fazemos o cruzamento desses resultados. Isso possibilita o retorno para que o sistema possa junto com os cálculos internos recriar a semiontologia final da fase baseada em *string,* alinhada para visualização. O cruzamento ocorre seguinte com o preenchimento da TAB. 21:

TABELA 21
Tabela sumário primeira fase avaliação

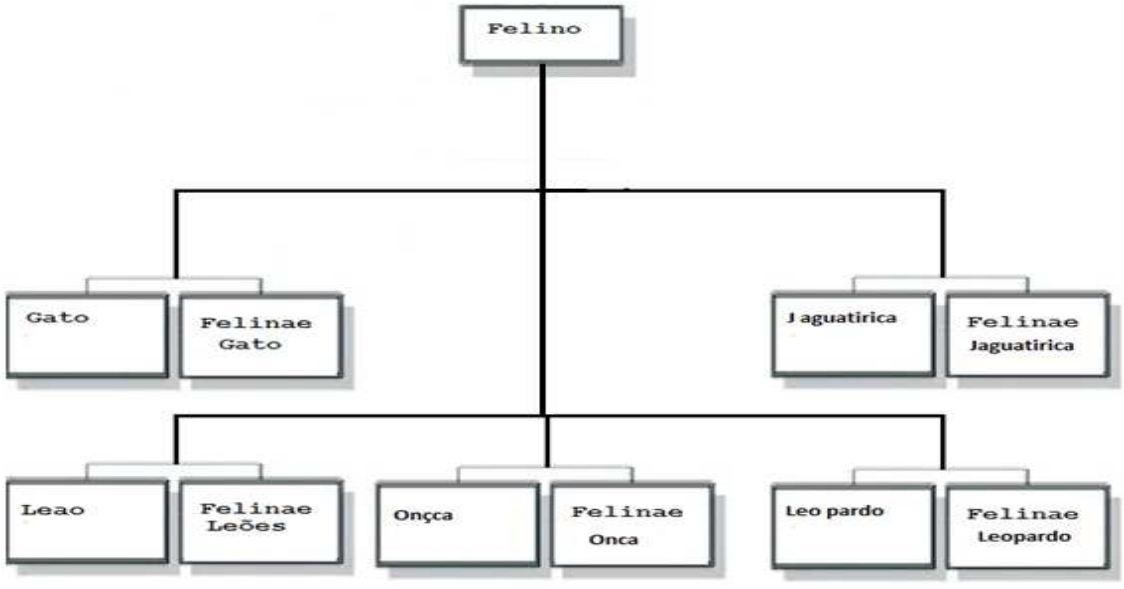
Fase 1	Fase 2		Fase 3		Resultados	
Candidatas	Pontuação	Candidata	Pontuação	Candidata	Pontuação	Candidata
Sim	5	Sim	5	Sim	5	Sim
Sim	5	Sim	4	Sim	5	Sim
Sim	2	Não	2	Não	2	Não
Sim	3	Sim	2	Não	3	Sim
Sim	3	Sim	2	Não	3	Sim
Sim	5	Sim	4	Sim	4	Sim
Não	2	Não	1	Não	1	Não

Fonte: Autoria própria

As marcações de “Candidatas” feitas nas três primeiras fases do procedimento baseado em *string* são transferidos para a TAB. 14 e a média das marcações e notas nos dá um resultado final para retorno ao sistema.

Da mesma forma que nas fases anteriores os dados devem ser retornados ao sistema (FIG. 32) após o preenchimento da TAB. 21. O retorno permite ao sistema gerar a semi ontologia referente à sumarização de todas as marcações da fase de alinhamento baseado em string. A semi ontologia ficará similar ao exemplo representado pela FIG. 35.

FIGURA 35- Representação de semi ontologia



Fonte: Autoria própria

Neste ponto do processo uma média com as avaliações registradas nas tabelas auxiliares, que tem como modelo a TAB. 2, ao longo do processo. Como exemplo de

preenchimento temos a TAB. 22. Desta forma temos uma média geral de avaliação ate o momento. Essa média é registrada em uma tabela auxiliar que tem como modelo a TAB. 5, como exemplo temos a TAB. 23.

TABELA 22
Tabela Auxiliar Avaliação

Tabela de Avaliação			
Classes/Subclasses	Corretude Alinhamento	Corretude Estrutura	Avaliação Final
Felis silvestris catus [Gato;Gato]	5	5	5

Fonte: Autoria própria

TABELA 23
Tabela Auxiliar sumarização notas

	Ontologia 1	Ontologia 2
Corretude Alinhamento	5	4
Corretude Estrutura	4	5
Avaliação Final	4,5	4,5

Fonte: Autoria própria

5.3.1.2 - Baseado em estrutura

Aqui será descrito o procedimento para o alinhamento semiautomático de duas ontologias levando-se em conta os alinhamentos baseados em estrutura. Para tal alinhamento são utilizadas as ferramentas desenvolvidas durante o processo. Nesse roteiro de alinhamento temos duas etapas que são ativadas e analisadas de forma similar ao alinhamento baseado em *string*.

A primeira fase desta etapa é ativada ao fim da etapa de alinhamento baseado em *string*. Aqui, temos como retorno dos dados enviados ao sistema uma lista de super e subclasses a serem alinhadas, além de uma semiontologia. Da mesma forma que nas fases anteriores o sistema gera uma tabela semelhante à tabela 6 já com as classes e subclasses candidatas ao alinhamento e suas respectivas notas (TAB.25).

Assim, como já feito anteriormente nos outros processos de alinhamento do método, uma avaliação em cima dessa lista deve ser feita. No entanto nesta fase temos uma

avaliação mais detalhada que também leva em conta uma estrutura preliminar apresentada pelo sistema (FIG. 38). Essa estrutura é descrita a frente nessa mesma seção.

Neste ponto é analisada a estrutura que essas entidades listadas representam, além da similaridade do alinhamento direto. O caráter semiautomático é aplicado da mesma forma que nas outras fases do processo, o conhecimento prévio de ontologias e do domínio de cada uma delas deve ser utilizado para a caracterização das candidatas ao alinhamento. Assim a TAB. 16 deve ter seu campo “Candidatas” marcados com sim ou não.

Detalhamento

Além de uma lista escrita de candidatas, nesta fase do processo também temos uma estrutura candidata (um exemplo é a FIG. 36) montada (entidade-lista e estrutura) que deve ser considerada para o alinhamento e posterior avaliação. Os níveis de similaridade continuam baseados nas notas dadas pelo sistema às entidades. No entanto para a avaliação deve-se considerar a relação entre as classes e suas subclasses.

Essa análise segue os seguintes passos:

- Analisar todos os pares de sub e superclasses retornados através de uma lista de pares pelo sistema;
- Analisar a estrutura da semi ontologia (FIG.36) apresentada e verificar sua concordância com a lista de classes retornada pelo sistema. Para isso todos os pares, de sub e superclasses, retornados na estrutura são considerados;
- Uma métrica simplificada foi criada para as duas análises anteriores, para primeira a mesma da fase anterior foi utilizada e para segunda segue a seguinte ordem, que é baseada no conhecimento do domínio e da estrutura esperada;

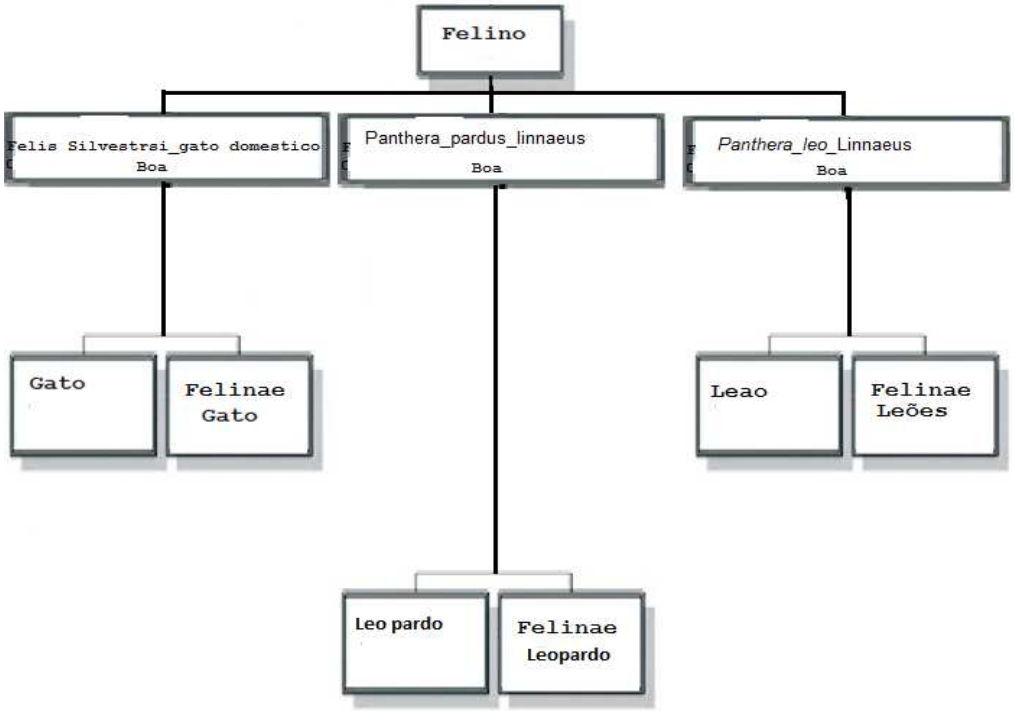
TABELA 24
Níveis de equivalência fase estrutura

Equivalência	Pontuação
Ótimo	5
Bom	4
Médio	3
Baixo	2
Excluir	1

Fonte: Autoria própria

- As notas são aplicadas a lista de entidades e à estrutura apresentada (FIG. 36);

FIGURA 36- Representação estrutura criada



Fonte: Autoria própria

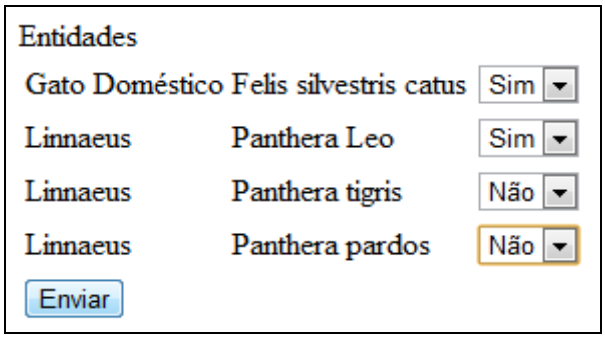
- Marca-se “sim” ou “não” para o campo “Candidatas” da TAB. 16 de acordo com a análise da estrutura acima (FIG. 36). No caso, não alinharemos leopardos e leões, então os leopardos “estão fora”, pois com a análise da comparação de *string* e conhecimento prévio sobre o assunto sabe-se que leões e leopardos não são a mesma “coisa”;

TABELA 25
Tabela de avaliação candidatas ao alinhamento fase estrutura

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinae	Pontuação	Candidata
Gato Doméstico	Boa	*Felis silvestris catus*	*5*	*Sim*
Linnaeus	Boa	*Panthera Leo*	*5*	*Sim*
Linnaeus	Boa	*Panthera tigris*	*2*	*Não*
Linnaeus	Boa	*Panthera pardus*	*2*	*Não*

- A marcação dessa fase deve ser feita na tabela gerada, como na TAB. 25. O sistema possui campo para tal marcação (FIG.37).

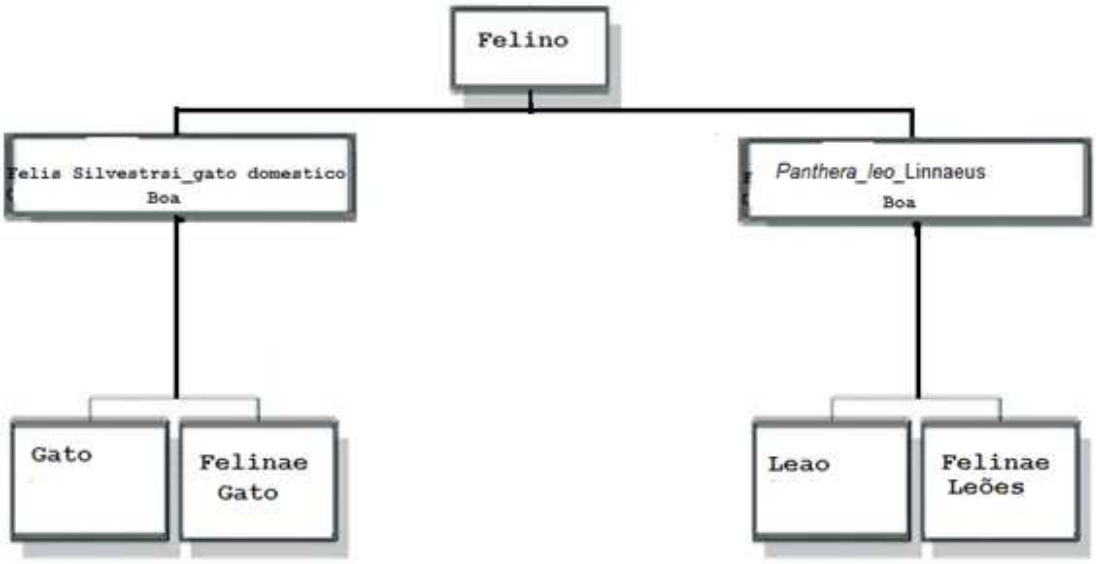
FIGURA 37 - Tela de retorno ao sistema fase estrutura



Fonte: Autoria própria

Após a marcação, a estrutura final é gerada, conforme apresentado na FIG. 38.

FIGURA 38 - Representação semi ontologias gerada fase estrutura final



Fonte: Autoria própria

Utilizamos essa estrutura preliminar para comparação com as listas que serão geradas nas próximas etapas. Ela é uma das bases para marcação final de geração da meta ontologia. A tabela de sumarização de avaliações deve ser preenchida (TAB. 1), criando desta forma uma tabela de avaliação preliminar (TAB. 26).

TABELA 26
Avaliação preliminar (tabela Auxiliar) fase estrutura

Tabela de Avaliação					
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude Descrição	Avaliação Final
Felis silvestris catus	[Gato;Gato]	5	5	5	5
Panthera Leo	[Leao;Leões]	5	5	5	5
Panthera tigris	[Tigre;Tiges]	5	5	5	5
Panthera pardus	[Leo pardo;Leopardo]	5	5	5	5

Fonte: Autoria própria

Após o preenchimento da TAB. 26, o resumo das notas dessa etapa deve ser colocado na forma de média simples na TAB. 7, criando uma nova tabela auxiliar que ficará como na TAB. 27.

TABELA 27
Sumarização Auxiliar primeira fase alinhamento estrutura

[redacted]	Ontologia 1	Ontologia 2
Corretude Alinhamento	5	5
Corretude Estrutura	5	5
Corretude Descrição	5	5
Avaliação Final	5	5

Fonte: Autoria própria

Segunda fase do alinhamento baseado em Estrutura

A segunda fase desta etapa é ativada ao fim da primeira fase de alinhamento baseado em estrutura. Aqui, também temos como retorno dos dados enviados ao sistema uma lista de super e subclasses a serem alinhadas, mas não uma semi ontologia. Da mesma forma que na primeira fase o sistema gera uma tabela semelhante à tabela 6 já com as classes e subclasses candidatas ao alinhamento e suas respectivas notas (TAB. 28).

De forma similar a primeira fase desta etapa de alinhamento, uma avaliação em cima dessa lista deve ser feita junto a uma avaliação mais detalhada da estrutura final da fase anterior apresentada pelo sistema (FIG. 40).

As análises aplicadas durante a segunda fase do processo de alinhamento baseado em estrutura é aplicado da forma descrita a seguir:

Detalhamento

A segunda etapa é aplicada sobre a estrutura gerada pela comparação de descrições da seguinte forma:

- Pegamos os nomes das subclasses que acompanham a lista e estrutura geradas anteriormente (TAB. 26 e FIG. 38) e submetemos a análise desenvolvida na primeira etapa do roteiro. Utilizamos a mesma regra de similaridade adotadas na seção "*matching* baseado em *string*";

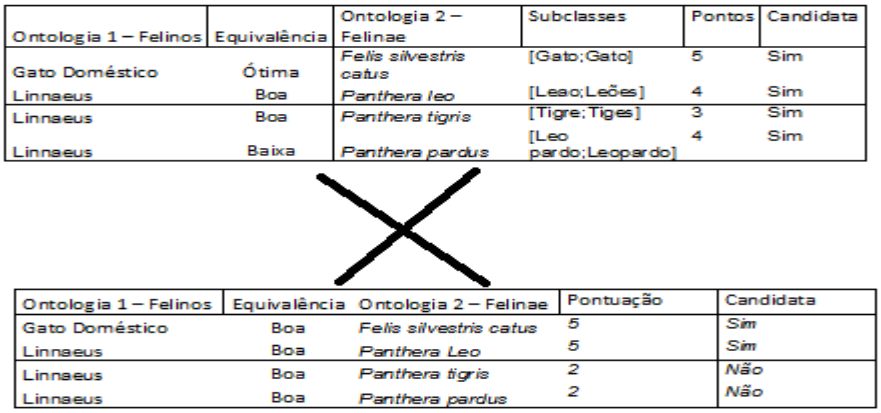
TABELA 28
Comparação de candidatas ao alinhamento fase estrutura com classes pai

Ontologia 1 – Felinos	Equivalência	Ontologia 2 – Felinae	Subclasses	Pontos	Candidata
Gato Doméstico	Ótima	*Felis silvestris catus*	[Gato;Gato]	5	Sim
Linnaeus	Boa	*Panthera Leo*	[Leao;Leões]	4	Sim
Linnaeus	Boa	*Panthera tigris*	[Tigre;Tiges]	3	Sim
Linnaeus	Baixa	*Panthera pardus*	[Leo pardo;Leopardo]	4	Sim

Fonte: Autoria própria

- A TAB. 28 é comparada com a tabela de superclasse como exemplificado na FIG. 39. Para isso, utiliza-se somente o campo pontuação das tabelas. Daí, tiramos uma "média simples" e somamos às notas dadas pelo usuário (que utiliza conhecimento prévio sobre o domínio) para criar a tabela final de marcação;

FIGURA 40 - Cruzamento de Tabelas de alinhamento



Fonte: Autoria própria

- Somente o campo que representa a pontuação dada, pelo sistema e usuários, é levado em conta para cada par de entidades, ou seja, não se leva em conta, aqui, análise de informações das entidades ou qualquer outra análise que pode influenciar o resto do processo. Desta forma temos a seguinte TAB. 29 como resultado;

TABELA 29
Tabela de Avaliação pontuação

Pontuação	Candidata	Pontos	Candidata	Final
5	*Sim*	5	Sim	Sim
5	*Sim*	4	Sim	Sim
2	*Não*	3	Sim	Não
2	*Não*	4	Sim	Sim

Fonte: Autoria própria

- O resultado final será marcado na tela de marcação já apresentada nessa seção FIG. 37;

A FIG. 40 representa o alinhamento final entre as duas ontologias base. Utilizamos essa estrutura para comparação e avaliação em relação às ontologias base do processo de alinhamento. A tabela de sumarização de avaliações deve ser preenchida (TAB. 1), criando desta forma uma tabela de avaliação preliminar (TAB. 30)

TABELA 30
Uma tabela de avaliação preliminar segunda fase estrutura

Tabela de Avaliação					
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude Descrição	Avaliação Final
Felis silvestris _gato Domestico	[Gato;Felinae Gato]	5	5	5	5
Panthera Leo	[Leao;Felinae Leões]	5	5	5	5
Panthera pardus	[Leo pardo;Leopardo]	5	5	5	5

Fonte: Autoria própria

Após o preenchimento da TAB. 30, o resumo das notas dessa etapa deve ser colocado na forma de média simples na TAB. 7, criando uma nova tabela auxiliar que ficará como na TAB. 31.

TABELA 31
Tabela sumarização auxiliar segunda fase estrutura

	Ontologia 1	Ontologia 2
Corretude Alinhamento	5	5
Corretude Estrutura	5	5
Corretude Descrição	5	5
Avaliação Final	5	5

Fonte: Autoria própria

5.3.2 – Avaliação

Nesta seção descreveremos somente as ações avaliativas por parte do usuário. Neste ponto todos os dados de retorno do sistema ja estão em mãos.

Como já descrito na seção 2 deste trabalho, existem quatro categorias principais de avaliação de ontologias: avaliação baseada na comparação com um padrão; avaliação baseada no uso da ontologia em uma aplicação e a subsequente avaliação dos resultados; avaliação que faz comparações com fontes de dados sobre o domínio; avaliação realizada por pessoas que tentam averiguar quão bem a ontologia atinge um conjunto pré-definido de critérios.

Para concepção do roteiro proposto toma-se como base a terceira e a quarta categorias de avaliação, uma vez que todo o processo de avaliação e de visualização será conduzido por pessoas. Além disso, seleciona-se um método de avaliação já consolidado para fundamentar as atividades de avaliação.

Para a atividade de avaliação foram criadas três métricas (TAB. 10) distintas para estrutura, completude e alinhamento. Para aplicação das métricas é feita uma inspeção visual nas estruturas geradas no processo de alinhamento. Essa inspeção visual leva em conta a comparacao das novas estruturas com as ontologias base e conhecimentos sobre o domínio.

TABELA 32
Métricas de avaliação

Avaliação	
Nível	Nota
Ótimo	5
Bom	4
Médio	3
Baixo	2
Ruim	1

Fonte: Autoria própria

As métricas da TAB. 32 devem ser aplicadas às seguintes avaliacões:

- Avaliação do alinhamento: A navegação pelas novas estruturas criadas (as semi ontologias exemplificadas nas FIG. 35, 38 e 43) deve ser feita. Utilizando-se as telas de visualizacao e alinhamento é possivel compará-las com as estruturas originais (ontologias iniciais utilizadas no processo) e, através dos outros suportes à visualização de dados (*labels*, caixas de descrição, informações laterais, etc), deve-se avaliar o alinhamento. Isso é feito através da averiguação das descrições das classes alinhadas, além da análise das novas relações representadas. Isso tudo é feito em relação às ontologias iniciais do processo e com conhecimentos sobre o domínio modelado;

- Avaliação de estrutura: avaliando-se as novas estruturas geradas em relação as estruturas base, classifica-se o nivel estrutural de acordo com a TAB. 32. Para que isso seja possível, a navegação entre as classes e subclasses deve ser feita, a análise das relações *subclassof* da lista de classes e subclasses retornada do alinhamento deve ser feita em comparação com as originais;
- Avaliação de completude: verifica-se o conjunto de dados visualizados ao clicar em cada nó os filhos deste são representados na tela. Utilizando recursos próprios, o participante avalia que tipo de inferência pode fazer a partir da visualização, considerando que o participante não é especialista no domínio. Para cada par de entidades alinhadas o usuário deve averiguar as relações entre os termos para avaliação de relações redundantes ou errôneas;

A TAB. auxiliar 33 deve ser utilizada para a metrificação dos pontos de avaliação considerados acima:

TABELA 33
Tabela auxiliar de avaliação fase avaliação

Tabela Auxiliar de Avaliação		
Classes/Subclasses		Nota
Felis silvestris catus	[Gato;Gato]	
Panthera Leo	[Leao;Leões]	
Panthera tigris	[Tigre;Tiges]	
Panthera pardus	[Leo pardo;Leopardo]	

Fonte: Autoria própria

As avaliações acima devem ser feitas seguindo os passos:

1. Carregam-se as ontologias participantes do processo;
2. Ativam-se inicialmente o processo de alinhamento;
3. Analisa-se as listas de entidades, para cada par de entidades quantifica-se as avaliações de acordo com as métricas definidas na TAB. 32. Uma tabela auxiliar deve ser preenchida com as notas dadas;
4. Carregam-se as novas estruturas nas telas de visualização, o tipo de visualização utilizada é escolhido de acordo com a preferência de quem esta aplicando o processo. Para cada agrupamento de entidades a metrificação deve ser feita e as tabelas de cada tipo de avaliação devem ser preenchidas;
5. Após o término do preenchimento das tabelas auxiliares uma sumarização da TAB. 34 de avaliação é feita, para que se possa reduzir à uma nota única por parâmetro de

avaliação na TAB. final 35.

TABELA 34
Sumarização das Avaliações

Tabela de Avaliação				
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude Descrição	Avaliação Final
Felis silvestris catus	[Gato;Gato]	5	5	5	5
Panthera Leo	[Leao;Leões]	4	4	5	4.5
Panthera tigris	[Tigre;Tiges]	3	2	1	2
Panthera pardus	[Leo pardo;Leopardo]	3	2	1	2

Fonte: Autoria própria

A tabela final de avaliação (TAB.11) é obtida com a média simples entre as notas dadas para todos os pares de entidades em relação a cada um dos parametro relacionados (alinhamento, descrição e estrutura).

A TAB. 35 é utilizada como exemplo da aplicação da compilação final de notas das três fases do alinhamento:

TABELA 35-
Compilação Final de Notas

Compilação Final		
[redacted]	Ontologia 1	Ontologia 2
Corretude Alinhamento	3,7	4
Corretude Estrutura	3,2	3,8
Corretude Descrição	3	3,5
Avaliação Final	3,3	3,8

Fonte: Autoria própria

Após o preenchimento da tabela final, temos como criar uma nota geral de avaliação das ontologias, além de algumas notas intermediárias. Aqui optamos pela média simples, mas outras métricas e formalizações matemáticas podem ser efetuadas a partir das notas dadas para cada passo do processo de avaliação.

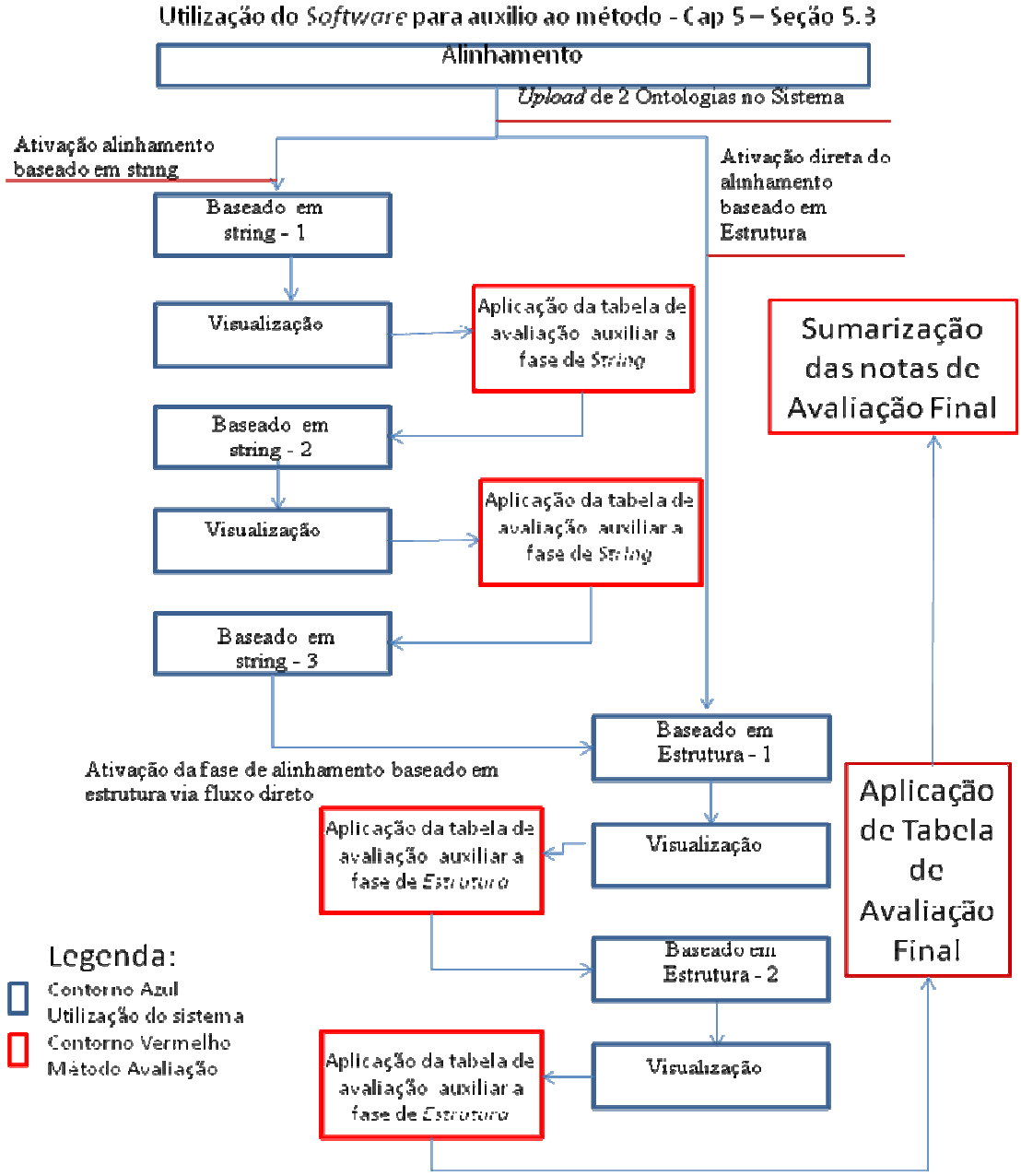
Além das notas finais resumidas, também se averiguaram os números relativos à quantidade de classes alinhadas e classes descartadas em relação ao número total de classes e em relação ao número de classes candidatas. Isso deve ser utilizado como um suporte quantitativo total de verificação de interoperabilidade entre as duas ontologias base. Isso também pode ser utilizado como indicador de qualidade das ontologias envolvidas no processo e alinhamento, pois ontologias que se encaixam na escolha determinada na seção 5.1

deste trabalho devem ter baixos valores nas relações entre classes alinhadas e classes descartadas.

Organograma final

Os passos descritos nessa seção podem ser sumarizados com mais detalhamento no organograma apresentado na FIG.41.

FIGURA 41 - Fluxo de passos dos roteiros de avaliação



Fonte: Autoria própria

6 – Resultados e Aplicação

Neste capítulo, será descrita a aplicação do método proposto em duas ontologias, em caráter de estudo específico dos resultados que o método pode apresentar, além de se apresentar os resultados parciais que permitiram a aplicação principal. A descrição é feita em detalhes apresentando, para cada passo, tabelas e imagens de suporte. O desenvolvimento das avaliações seguiu os passos descritos no Capítulo 5. Para cada seção do Capítulo 5 foram mapeadas seções do Capítulo 6 da seguinte forma: Seção 5.2 foi mapeada na seção 6.1, a seção 5.3 na seção 6.2.

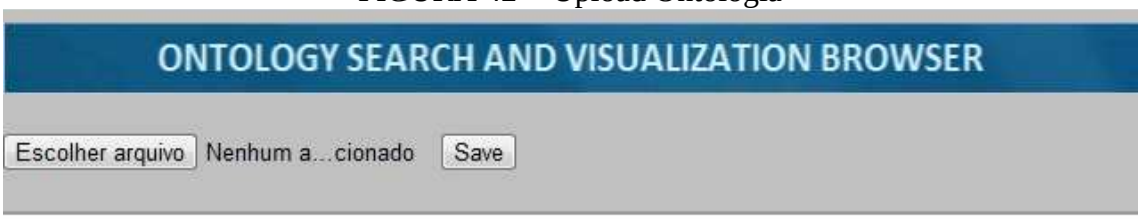
6.1 – *Software*

Aqui será apresentado o *software* desenvolvido de acordo com o desenvolvimento descrito na seção 5.2 do capítulo de metodologia. 6.1.1 apresentará o protótipo de visualização desenvolvido; No Anexo 1 serão descritos os resultados parciais equivalentes ao *Parser* e os algoritmos de alinhamento em alto nível.

6.1.1 - Protótipo de Visualização e *Matching*

Nesta seção apresentamos o protótipo criado, bem como suas telas e funcionalidades. A tela abaixo representa a funcionalidade de carregar uma nova ontologia no sistema.

FIGURA 42 - Upload Ontologia

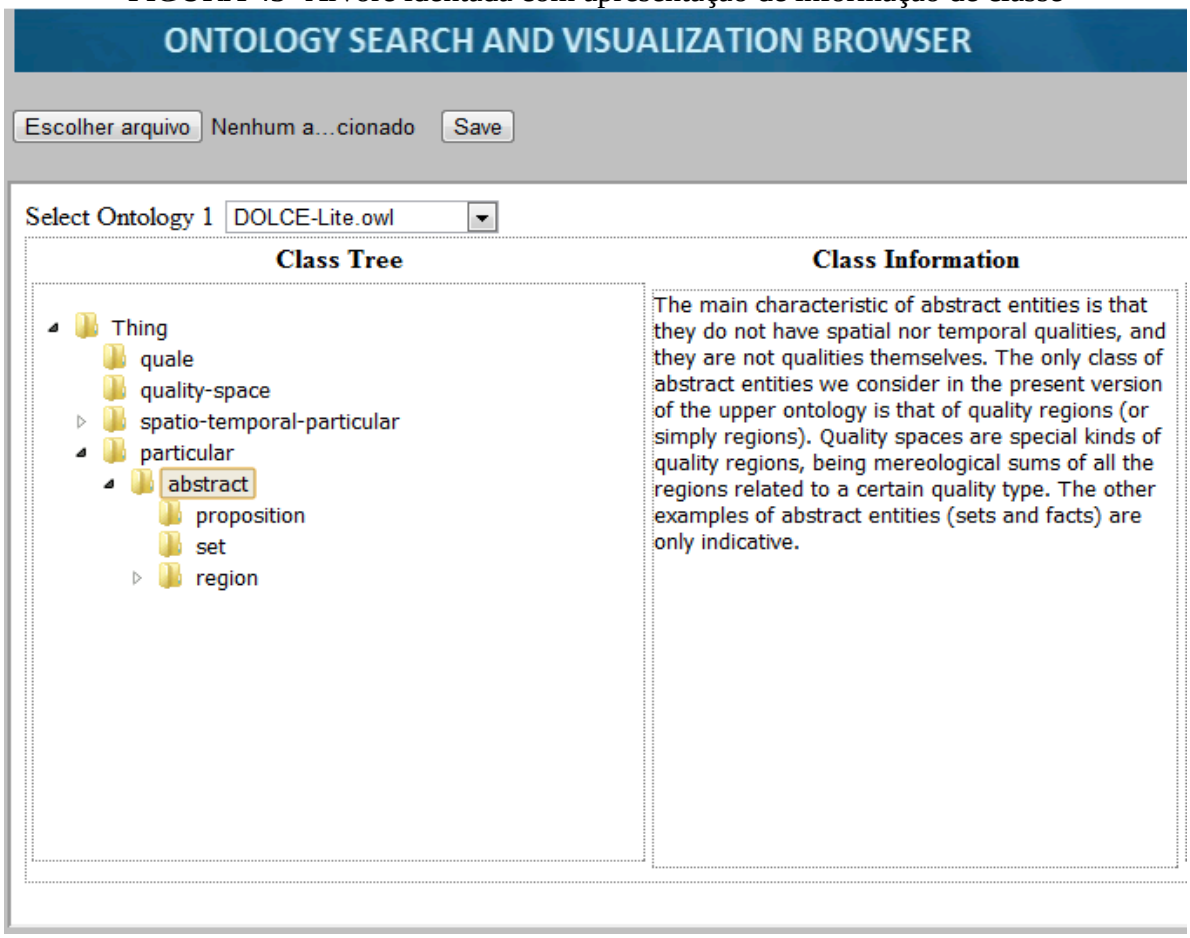


Fonte:

A tela (FIG. 42) é utilizada para se escolher um novo arquivo OWL na máquina local e carregá-la no sistema. Quando isso é feito o sistema de *parser* é ativado

automaticamente, apresentando as várias telas de visualização criadas para a manipulação de ontologias.

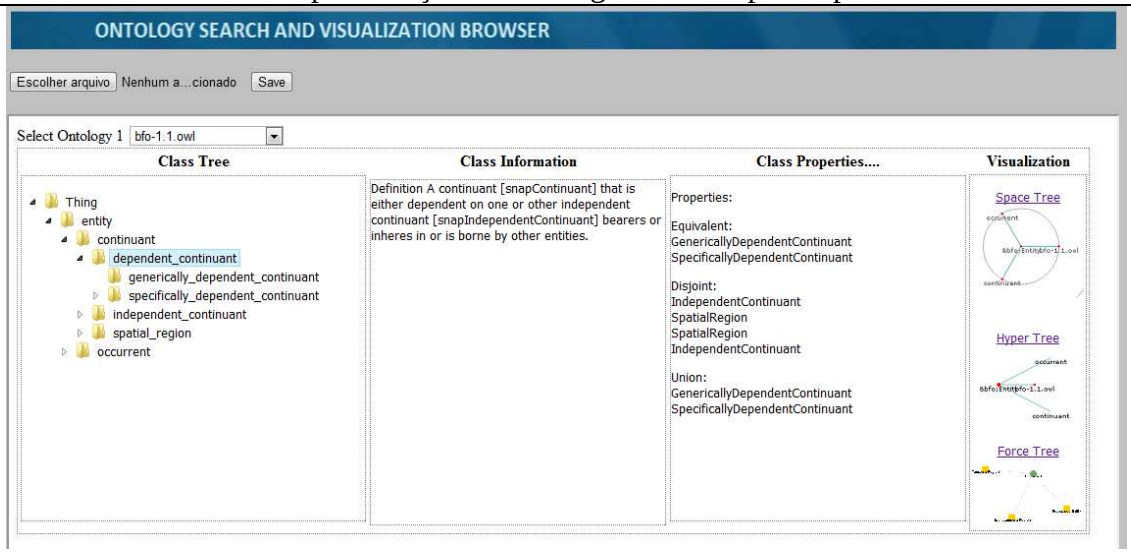
FIGURA 43- Árvore identada com apresentação de informação de classe



Fonte: Própria

Na FIG. 43 temos a visualização parcial da ontologia BFO. O padrão de visualização é subdividido em quatro espaços distintos com as apresentações de informações e a possibilidade de outros três tipos de visualização, conforme ilustrado na FIG. 44.

FIGURA 44 - Representação da ontologia BFO no protótipo desenvolvido



Fonte: Autoria própria

Na FIG. 44 temos a visualização da ontologia BFO. O padrão de visualização é subdividido em quatro espaços distintos.

- A árvore criada pela relação entre classe e subclasses da ontologia (*classtree*);
- A informação sobre classe/entidade modelada. Essa informação é obtida dentro das classes em suas *tags comments;* dentro dessas *tags* específicas está contido a descrição literal do que significa cada classe (*class Information*);
- Propriedades: aqui representamos as propriedades de cada uma das entidades selecionadas (*class properties*);
- Visualização: local onde três outros tipos de visualização são disponibilizados ao usuário (*visualization*).

A navegação por essa interface é bem similar a interação feita nos sistemas de organização de arquivos conhecidos. Primeiro se escolhe a ontologia clicando no *dropdown* em frente ao label “Ontology 1”; ao se escolher a ontologia sua árvore é gerada e as outras três interfaces de visualização são disponibilizadas. Para visualizar as informações da cada entidade o usuário deve clicar sobre ela; assim que isso ocorre as informações ao lado são apresentadas (se elas existirem dentro da ontologia) . O restante da presente seção descreve os tipos de visualização possíveis dentro do protótipo.

Arvore radial

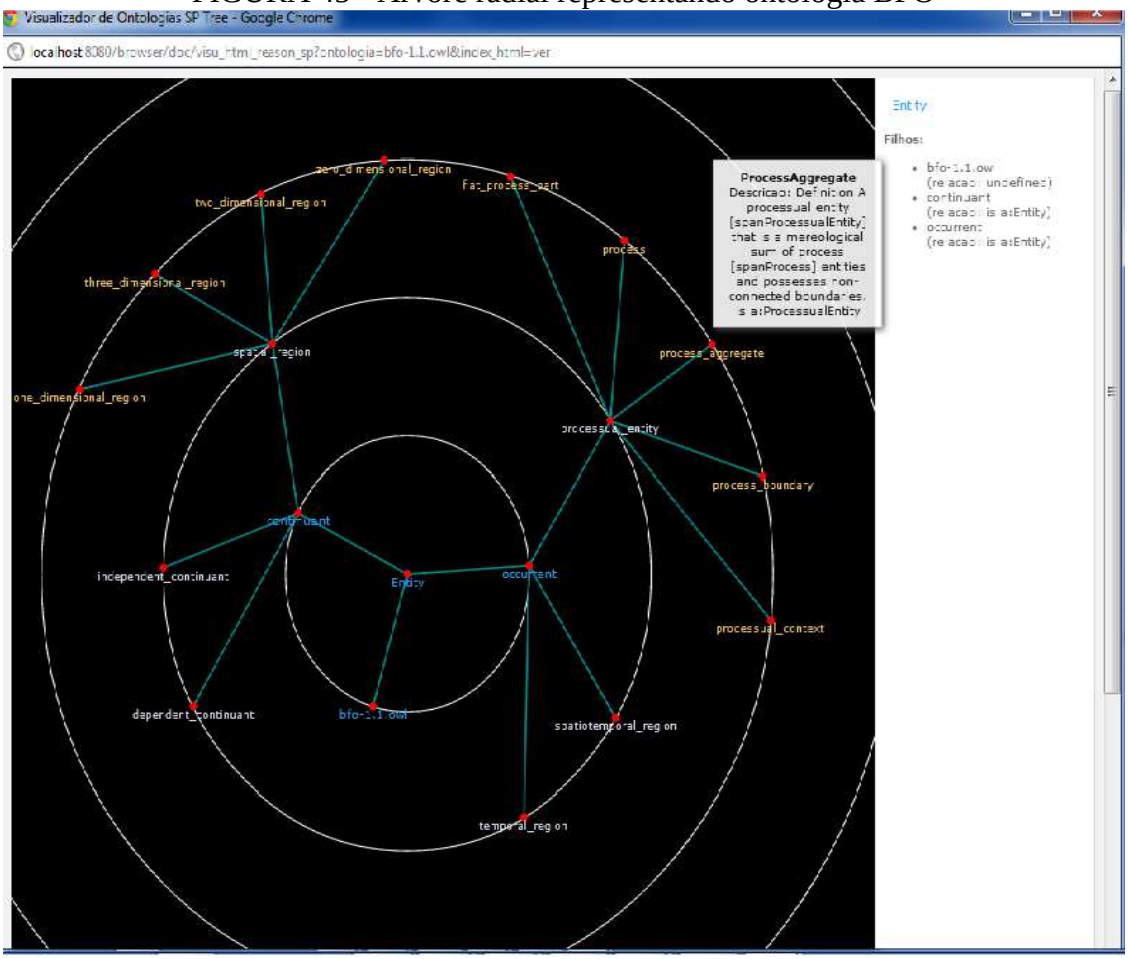
Duas telas distintas representam este tipo de árvore dentro do sistema. São elas a

Space Tree e a *Force Tree*, descritas a seguir.

Space Tree que tem visualização bem definida e com caminhos previsíveis durante a navegação. Círculos de parametrização estão presentes, de forma que é possível identificar com clareza qual nível da ontologia está sendo acessado.

A navegação se dá através de cliques em cada uma das pequenas esferas coloridas (representantes das classes). Quando se clica em uma das esferas ela se posiciona no centro da tela, seus filhos são apresentados na barra lateral e suas informações apresentadas em um *tooltip*.

FIGURA 45 - Árvore radial representando ontologia BFO

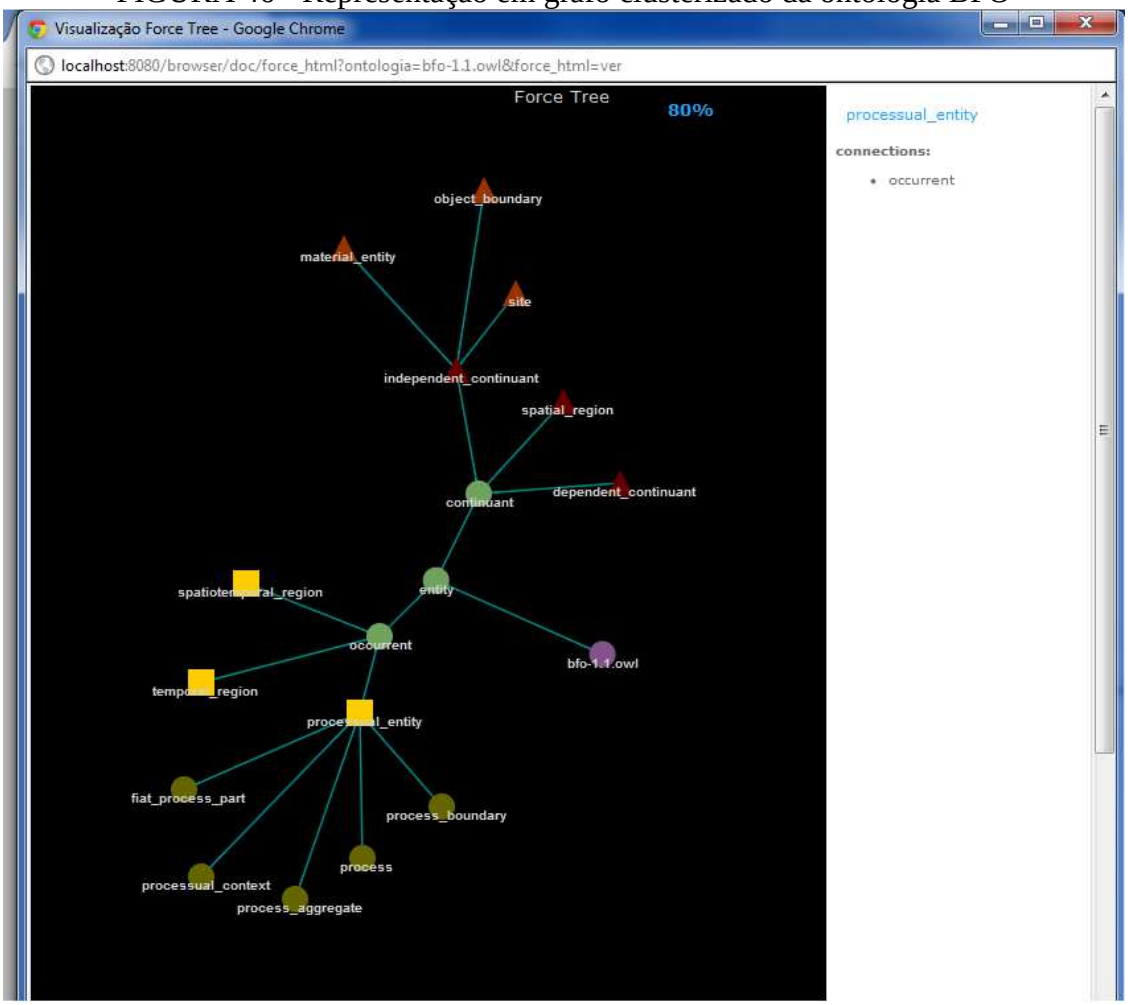


Fonte: Autoria própria

A FIG. 45 ilustra a visualização *Space Tree*. Ela mostra a classe *Entity* ao centro e posteriormente seus filhos. À medida que os dados se afastam da raiz as cores se diferenciam. Ao lado, apresentam-se informações sobre as conexões diretas da classe em evidencia. O *tooltip* nos fornece informações de propriedades e descrição da classe em evidencia. Esta essa tela ainda permite *zoom in* e *zoom out*, além de movimentação da árvore como um todo.

Force Tree (FIG. 46), diferente da *Space Tree*, tem livre movimentação dos elementos da árvore. Ela é dividida em agrupamentos de dados cada um com uma forma e cores diferentes. Todos os elementos da tela podem ser colocados em evidência arrastando-os para qualquer local da tela. As informações sobre a classe aparecem em *tooltips* e as ligações da entidade aparecem na parte direita da tela.

FIGURA 46 - Representação em grafo clusterizado da ontologia BFO



Fonte: Autoria própria

A FIG. 46 ilustra a visualização *Force Tree*. Ela mostra a classe *Entity* ao centro e posteriormente seus filhos. À medida que os dados se afastam da raiz as cores e formas se

diferenciam, cada classe pai forma um *cluster* de imagens que representam seus filhos. Ao lado, apresentam-se informações sobre as conexões diretas da classe em evidencia. O *tooltip* nos fornece informações de propriedades e descrição da classe em evidencia. Esta tela ainda permite *zoom in* e *zoom out*, além de movimentação da árvore como um todo e movimentação de cada elemento individualmente.

Este tipo de árvore emula uma visualização 3D, pois permite que as imagens se sobreponham, além de proporcionar perspectivas e sombra. Somado ao livre movimento, tem-se a impressão de uma imagem em três dimensões. As cores e imagens representativas de cada classe são fornecidas de forma aleatória pelo sistema interno de *parser*.

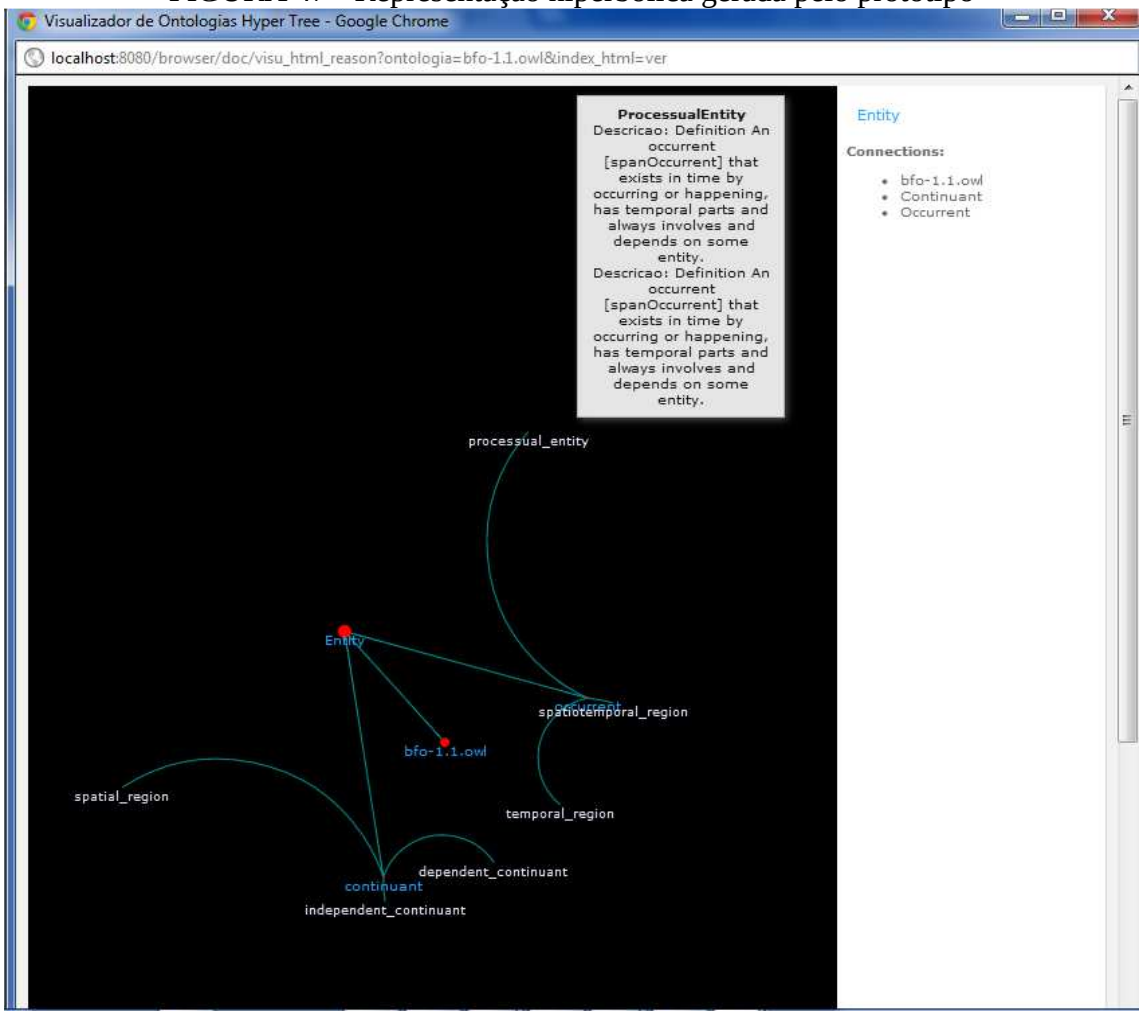
Arvore hiperbólica.

Aqui utilizamos a visualização hiperbólica. Para atender aos propósitos da ferramenta, modificamos a visualização da seguinte forma: as distâncias seguem uma pequena alteração de regra em seu comportamento. Elas diminuem levemente a medida que se afastam do nó raiz. Outra diferença é que o nó considerado raiz, para efeito de representação, é o nó que representa a entidade que se está avaliando. Quando se clica em uma entidade, ela toma o lugar central na tela e toda a cadeia de representação segue como se ela fosse a raiz da árvore naquele momento.

Essa tela segue os mesmos padrões de suporte para a apresentação dos dados contidos em cada classe da ontologia:

- *Tooltip*: apresenta informações de descrição e propriedades das entidades. Basta posicionar o mouse sobre o nome da entidade que as informações são mostradas;
- Barra lateral: mostra as conexões diretas da entidade em evidência, na barra lateral estão representados os filhos e a classe pai;
- A árvore: representa as ligações entre as classes, mostrando dessa forma as relações hierárquicas de forma visual. Também permite a navegação de forma agrupada, não sendo necessário analisar toda a ontologia de uma única vez, mas sim por partes.

FIGURA 47 - Representação hiperbólica gerada pelo protótipo



Fonte: Autoria própria

A FIG. 47 que representa a visualização hiperbólica foi retirada do protótipo criado. Ela, como nas outras ilustrações, mostra a classe *Entity* como centro da árvore e posteriormente seus filhos e filhos dos filhos. À medida que os dados se afastam da raiz as cores e formas se diferenciam assim como o estilo das arestas que os interligam. As classes são dispostas de forma não linear, sendo as distâncias entres elas alteradas de forma exponencial. Ao lado, são apresentadas informações sobre as conexões diretas da classe em evidência. O *tooltip* nos fornece informações de propriedades e descrição da classe em evidência.

6.2 Avaliação e Aplicação das Utilizações de *Matching*

Nesta seção será demonstrada a aplicação dos roteiros descritos no Capítulo 5 nas seções 5.1 e 5.3 em duas ontologias. No caso foram escolhidas CELL (*Cell Ontology*) que mapeia tipos de células e FAO (*Fungal Anatomy Ontology*) ontologia específica de células fungo. Essas ontologias foram escolhidas por estarem de acordo com o padrão BFO e por serem de projetos já estabelecidos dentro do desenvolvimento de ontologias. Desta forma, essas ontologias são à entrada de dados do sistema desenvolvido e o objeto do estudo de caso. A partir delas são gerados os dados e a análise da aplicação do método proposto.

A seção 6.2.1 apresenta o resultado da aplicação dos roteiros propostos nas seções 5.1.1, 5.1.2 e 5.3.1 do capítulo 5 entre as duas ontologias; já a seção 6.2.2 traz os resultados da avaliação feita em cima dos dados obtidos na fase de alinhamento, ou seja, a aplicação dos roteiros propostos nas seções 5.1.3 e 5.3.2 do capítulo 5.

6.2.1 Alinhamento

A seção 6.2.1.1 apresenta o resultado do alinhamento basedo em nome (*string*); já a seção 6.2.1.2 traz os resultados do alinhamento baseado em estrutura.

6.2.1.1 CELL X FAO alinhamento baseado em nome

Avaliação do alinhamento: aqui iniciou-se a primeira fase do processo envolvendo as ontologias CELL e FAO. Para cada parte do processo de alinhamento aplicamos as instruções de avaliação do capitulo 5 itens 5.1.1 e 5.3.1. Mostramos abaixo os resultados desse passo, apresentando parcialmente as tabelas e representações. Para maiores informações das tabelas, ver Anexo 1.

Primeira fase alinhamento baseado em *String*

Após o *upload* dos arquivos OWL, que são a entrada de dados, para o sistema, ativa-se o processo de alinhamento (através de um botão específico de ativação dentro do sistema). O sistema processa o primeiro passo de alinhamento especificado na seção 5.2.2.1 (*matching* baseado em *string*) e como primeiro resultado retornado temos a seguinte lista de palavras (TAB. 36), juntamente com seu nível de equivalência, representantes de entidades candidatas ao alinhamento final:

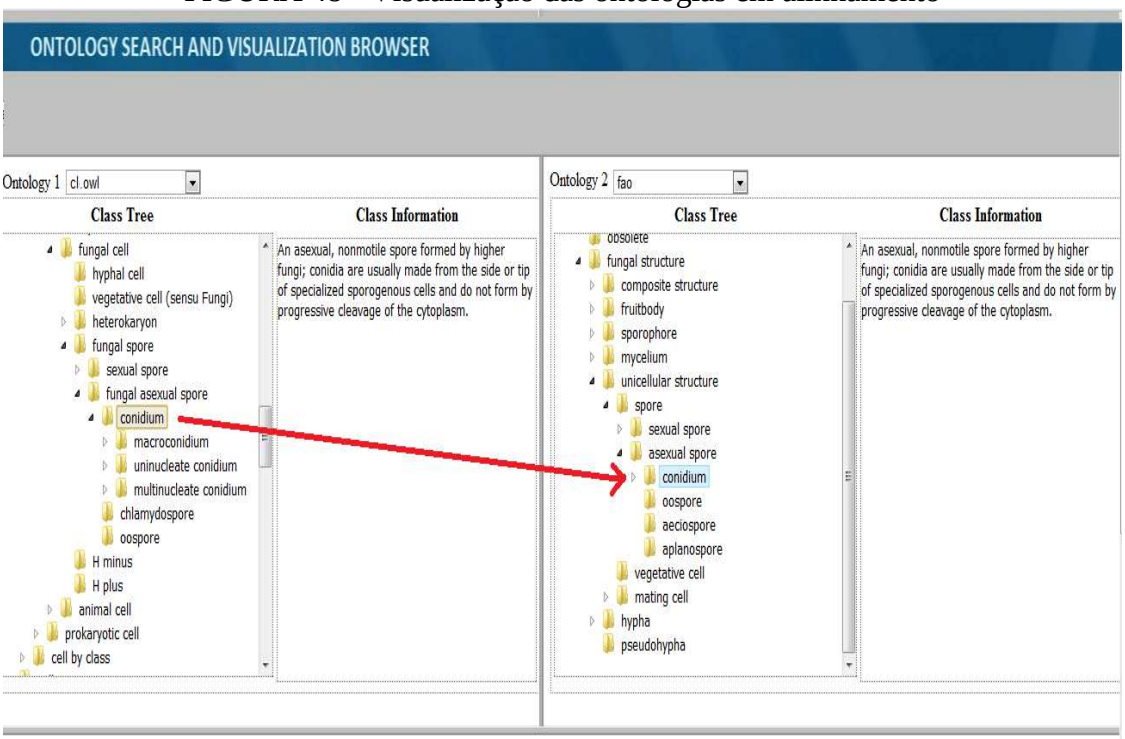
TABELA 36 –
Resultado da primeira iteração de *matching*

FAO	Equivalência	CELL
Spore	Ótimo	fungal spore
asexual spore	Ótimo	fungal asexual spore
+	+	+
+	+	+
+	+	+
multinucleate arthroconidium	Ótimo	uninucleate arthroconidium
uninucleate arthroconidium	Ótimo	multinucleate arthroconidium
multinucleate blastoconidium	Ótimo	uninucleate blastconidium
uninucleate blastconidium	Ótimo	multinucleate blastoconidium
multinucleate macroconidium	Ótimo	uninucleate macroconidium

Fonte: Autoria própria

Analisou-se as listas de entidades retornadas na TAB. 37 e, para cada par de entidades, foi realizada uma comparação dos nomes utilizando-se a visualização em árvore disponibilizada pelo sistema. Quantificaram-se as avaliações de acordo com as métricas fornecidas na seção 5.1 e 5.3 do cap 5. Para cada par de entidade foi verificada sua consistência e existência nas duas ontologias, além da paridade dos nomes (nível de similaridade).

FIGURA 48 - Visualização das ontologias em alinhamento



Fonte: Autoria própria

Com o exame da estrutura identada pode-se, além de comparar os nomes em si, ter uma impressão preliminar do resultado do alinhamento. A visualização da estrutura e das informações também permite iniciar a avaliação de completude dos dados modelados na ontologia. A análise se seguiu para todas os pares de entidades retornadas pelo sistema na primeira fase do alinhamento (já vistos na TAB. 36) e junto com a apreciação das estruturas identadas (FIG.48) alcançou-se a tabela auxiliar TAB. 37. Nesta tabela as marcações se as entidades são ou não candidatas ao alinhamento foram feitas, “sim” para candidatas e “não” para não candidatas, esta marcação é de acordo com o especificado na seção 5.1 do capítulo 5 deste trabalho :

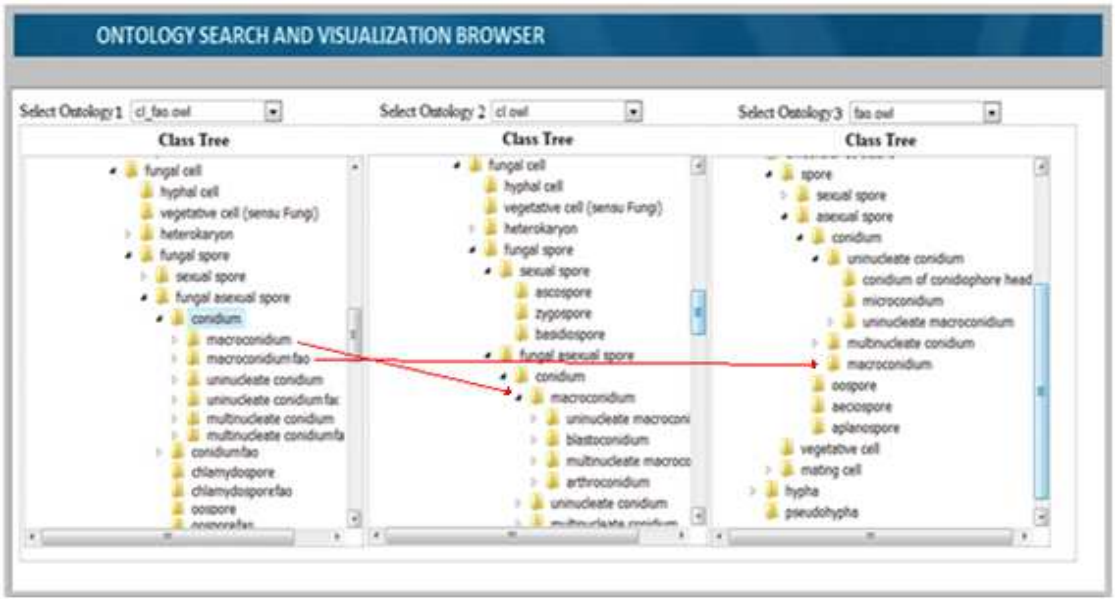
TABELA 37
Metrificação da primeira fase de alinhamento

Alinhamento *String* Primeira Fase			
FAO	Equivalência	CELL	Candidata
Spore	Ótimo	fungal spore	Sim
asexual spore + + +	Ótimo + + +	fungal asexual spore + + +	Sim + + +
multinucleate arthroconidium	Ótimo	uninucleate arthroconidium	Não
uninucleate arthroconidium	Ótimo	multinucleate arthroconidium	Não
multinucleate blastoconidium	Ótimo	uninucleate blastconidium	Não
uninucleate blastconidium	Ótimo	multinucleate blastoconidium	Não
multinucleate macroconidium	Ótimo	uninucleate macroconidium	Não

Fonte: Autoria própria

O resultado da TAB. 37 foi informado ao sistema através da tela apresentada na seção 5.3 (FIG.37) e gerou a primeira estrutura preliminar alinhada. A FIG. 49 mostra a visualização dessa estrutura preliminar alinhada, chamada de cl_fao.owl, a esquerda, a ontologia CELL ao meio e a ontologia FAO a direita.

FIGURA 49 - Visualização da primeira semi ontologia gerada



Fonte: Autoria própria

A avaliação da visualização na FIG. 49 permitiu aplicar as métricas dadas nas seções 5.1 e 5.3 do capítulo 5 à entidades candidatadas ao alinhamento e avaliação. Para a aplicação dessas métricas compararam-se as classes de cada uma das três estruturas

apresentadas (cl_fao *versus* cl.owl *versus* FAO), assim verificou-se a corretude do alinhamento inicial dado pelo sistema. Por exemplo, verificou-se se as entidades *macronidium* das duas ontologias base (CELL e FAO) estavam sob a mesma entidade pai (*conidium*). Assim, segue-se a comparação de todos os pares de nomes até o fim da lista dada na TAB. 37. A tabela demonstrativa da primeira avaliação preliminar do alinhamento (TAB. 38), que foi baseada na estrutura acima, segue da seguinte forma:

TABELA 38
Sumarização da primeira fase de avaliação do alinhamento

Classes/Subclasses				
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Avaliação Final
fungal spore	[fungal asexual spore/ asexual spore fao; sexual spore/ sexual spore fao]	5	5	5
sexual spore	[ascospore/ ascospore fao; Basidiospore/ Basidiospore fao; Zygospore/ Zygospore fao]	5	5	5
fungal asexual spore	[chlamydospore/ chlamydospore fao; conidium conidium fao; oospore/ oospore fao]	5	5	5
+	+	+	+	+
+	+	+	+	+
+	+	+	+	+

Fonte: Autoria própria

A TAB. 38 mostra as entidades pai que foram extraidas da ontologia CELL e as candidatas a alinhamento que foram retiradas da ontologia CELL e da ontologia FAO, as entidades da segunda ontologia tem seus nomes somados ao nome de sua ontologia na representação. Além disso a tabela 38 traz as notas dadas a cada par de entidades apresentadas na TAB. 37 em relação a corretude de alinhamento e estrutura.

A quantificação da primeira fase do processo de alinhamento se baseou, além da comparação direta dos nomes, no conhecimento que o avaliador tem sobre o domínio. Também isso pôde ser identificado através das percepções possíveis a partir das visualizações diversas dos dados.

As segunda e terceira etapas do alinhamento baseado em nome (*string*) tem metrificações mais sólidas (tem como alicerce algoritmos de substituição e de comparação exata) e menos abertas à interpretações. Não é necessario nenhum comando específico para

ativação dessas fases, elas são alcançadas pela continuação do processo de alinhamento que se inicia na fase 1. Seguimos com o resumo dos resultados das duas.

Segunda fase alinhamento baseado em *String*

A segunda etapa gera novos dados paras entradas iniciais (ontologias base) somados aos retornos dados ao sistema na primeira fase de avaliação baseada em *String* (retorno de candidatas FIG. 32). A apresentação dos resultados dessa fase se seguirá de acordo com os resultados que foram emitidos pelos algoritmos descritos na secção 5.2 do capítulo 5 deste trabalho.

Após a entrada de dados referente à segunda fase do alinhamento baseado em *string* no sistema, inicialmente, temos como saída (retorno dado pelo sistema a uma entrada de dados) a TAB. 39. Essa tabela traz a lista de entidades candidatas ao alinhamento e seus respectivos “valores” de equivalência, essa lista foi produzida pelos algoritmos utilizados no sistema referentes a segunda fase do alinhamento.

TABELA 39 –
Demonstração dos resultados da segunda fase de alinhamento

FAO	Equivalência	CELL
Spore	Médio	fungal spore
asexual spore	Bom	fungal asexual spore
+	+	+
+	+	+
+	+	+
multinucleate arthroconidium	Baixo	uninucleate arthroconidium
uninucleate arthroconidium	Baixo	multinucleate arthroconidium
multinucleate blastoconidium	Baixo	uninucleate blastconidium
uninucleate blastconidium	Baixo	multinucleate blastoconidium
multinucleate macroconidium	Baixo	uninucleate macroconidium

Fonte: Autoria própria

A TAB. 39 mostra o resultado da segunda iteração. Como as pontuações retornadas não tem mais grande frequência de valores altos, a metrificação pôde ser diretamente relacionada a elas. Então, os resultados seguintes podem ser considerados menos subjetivos. A tabela 40 mostra a pontuação de parte das entidades retornadas pelo sistema na segunda fase do alinhamento baseado em *string*.

TABELA 40
Avaliação das entidades candidatas

Alinhamento *String* Segunda Fase				
FAO	Equivalência	CELL	Pontuação	Candidata
Spore	Médio	fungal spore	3	Sim
asexual spore	Bom	fungal asexual spore	4	Sim
Conidium	Ótimo	Conidium	5	Sim
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
multinucleate arthroconidium	Baixo	uninucleate arthroconidium	2	Não
uninucleate arthroconidium	Baixo	multinucleate arthroconidium	2	Não
multinucleate blastoconidium	Baixo	uninucleate blastconidium	2	Não
uninucleate blastconidium	Baixo	multinucleate blastoconidium	2	Não
multinucleate macroconidium	Baixo	uninucleate macroconidium	2	Não

Fonte: Autoria própria

A partir da TAB. 40, uma semi ontologia similar às entidades apresentadas na FIG. 53 foi construída. Junto a análise das notas aplicadas, aqui também analisou-se as listas de entidades: para cada par de entidades uma comparação, utilizando-se a visualização em árvore, é feita entre os nomes das entidades apresentadas. Ao final, quantificaram-se as avaliações de acordo com as métricas dadas na seções 5.1 e 5.3 do cap 5 nas TAB. 40 e 41 e os resultados foram informados ao sistema através da tela para esse fim (FIG.32).

TABELA 41–
Pontuação da segunda fase de alinhamento

Tabela Auxiliar de Avaliação				
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Avaliação Final
fungal spore	[fungal asexual spore/ asexual spore fao; sexual spore/ sexual spore fao]	5	5	5
sexual spore	[ascospore/ ascospore fao; Basidiospore/ Basidiospore fao; Zygospore/ Zygospore fao]	5	5	5
fungal asexual spore	[chlamydospore/ chlamydospore fao; conidium conidium fao; oospore/ oospore fao]	5	5	5
+	+	+	+	+
+	+	+	+	+
+	+	+	+	+

Fonte: Autoria própria

A quantificação da segunda fase fase do processo de alinhamento se baseou, assim como na primeira necessita do conhecimento do avaliador.

Terceira fase alinhamento baseado em *String*

A terceira e última fase do alinhamento baseado em *string* segue a mesma linha da fase anterior. Aqui temos a menor subjetividade nas notas aplicadas ao alinhamento pelo sistema, pois só algoritmos de comparação exata são utilizados. Segue a amostra do resultado da aplicação desta fase (TAB. 42):

TABELA 42
Resultado preliminar da terceira fase de alinhamento baseado em nome

FAO	Equivalência	CELL
Spore	Baixo	fungal spore
asexual spore	Médio	fungal asexual spore
+	+	+
+	+	+
+	+	+
multinucleate arthroconidium	Excluir	uninucleate arthroconidium
uninucleate arthroconidium	Excluir	multinucleate arthroconidium
multinucleate blastoconidium	Excluir	uninucleate blastconidium
uninucleate blastconidium	Excluir	multinucleate blastoconidium
multinucleate macroconidium	Excluir	uninucleate macroconidium

Fonte: Autoria própria

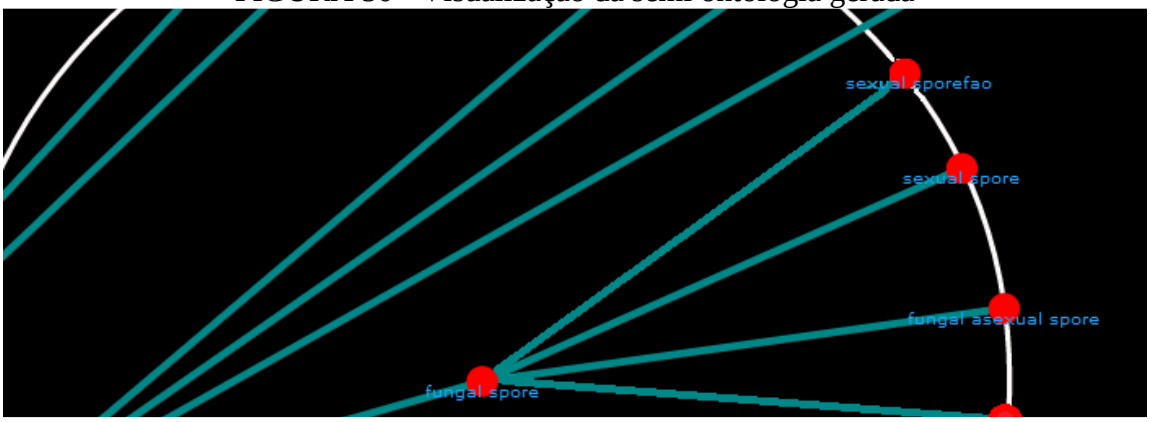
A TAB. 42 mostra a lista retornada pelo sistema como resultado da entrada de dados da terceira iteração de alinhamento baseado em *string*. A TAB. 43, a seguir, mostra a pontuação, data pelo sistema, de parte das entidades retornadas e a análise se são ou não candidatas ao alinhamento. Assim a marcação “sim” ou “não” foi feita como é mostrado.

TABELA 43
Avaliação das entidades candidatas fase estrutura

Alinhamento *String* Terceira Fase				
FAO	Equivalência	CELL	Pontuação	Candidata
Spore	Baixo	fungal spore	2	Não
asexual spore	Médio	fungal asexual spore	3	Sim
Conidium	Ótimo	Conidium	5	Sim
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
multinucleate arthroconidium	Excluir	uninucleate arthroconidium	1	Não
uninucleate arthroconidium	Excluir	multinucleate arthroconidium	1	Não
multinucleate blastoconidium	Excluir	uninucleate blastconidium	1	Não
uninucleate blastconidium	Excluir	multinucleate blastoconidium	1	Não
multinucleate macroconidium	Excluir	uninucleate macroconidium	1	Não

Os dados da TAB. 43 foram retornado aos sistema através da tela apresentada para esse fim (FIG. 32,) e, a partir destes dados, uma semi ontologia foi construída pelo sistema. Uma parte da visualização desta nova estrutura foi retirada e é demonstrada na figura 50. Aqui podemos ver a representação de *fungal spore* como classe pai e *sexual sporefao* (representante da ontologia FAO) e *sexual spore* (representante da ontologia CELL) como classes filhos:

FIGURA 50 - Visualização da semi ontologia gerada



Fonte: Autoria própria

Analisaram-se as listas de entidades (TAB.. 43), em cada par de entidades uma comparação utilizando-se a visualização foi feita. Ao final quantificaram-se as avaliações de acordo com as métricas dadas nas seções 5.1 e 5.3 do cap 5. A TAB. 44 foi gerada como resultado dessa análise.

TABELA 44
Tabela Auxiliar de Avaliação

Tabela Auxiliar de Avaliação		
Classes/Subclasses		Nota
fungal spore	[fungal asexual spore/ asexual spore fao; sexual spore/ sexual spore fao]	2
sexual spore	[ascospore/ ascospore fao; Basidiospore/ Basidiospore fao; Zygospore/ Zygospore fao]	5
fungal asexual spore	[chlamydospore/ chlamydospore fao; conidium conidium fao; oospore/ oospore fao]	3
+	+	
+	+	
+	+	

A avaliação do alinhamento final da fase baseada em *String* é feita com uma média das pontuações de cada par de classes analisadas em cada uma das três fases que

constituem esse passo de alinhamento. Sumarizando as notas aplicadas até o momento, o resultado ficou como na TAB. 45.

TABELA 45
Avaliação final da fase baseada em nome

Ontologias		Fase 1	Fase 2		Fase 3		Resultados	
FAO	CELL	Candidatas	Pont	Candidata	Pont	Candidata	Pont	Candidata
Spore	fungal spore	Sim	4	Sim	2	Não	3	Sim
asexual spore	fungal asexual spore	Sim	5	Sim	3	Sim	4	Sim
+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+
multinucleate arthroconidium	uninucleate arthroconidium	Não	3	Não	1	Não	2	Não
uninucleate arthroconidium	multinucleate arthroconidium	Não	3	Não	1	Não	2	Não
multinucleate blastoconidium	uninucleate blastconidium	Não	3	Não	1	Não	2	Não
uninucleate blastconidium	multinucleate blastoconidium	Não	3	Não	1	Não	2	Não

Fonte: Autoria própria

O fechamento preliminar de cada fase é utilizado para um fechamento final, o que leva a uma nota geral de avaliação TABELA 45 e 46. A seguir, apresenta-se a descrição dos resultados dos alinhamentos baseados em estrutura.

TABELA 46
Sumarização *String*

	Ontologia 1	Ontologia 2
Corretude Alinhamento	3.6	3.6
Corretude Estrutura	5	4
Avaliação Final	4.3	3.8

Fonte: Autoria própria

6.2.1.2 - CELL X FAO alinhamento baseado em estrutura

Aqui descreveremos os resultados da avaliação da segunda fase do alinhamento ente as ontologias escolhidas, o alinhamento baseado em estrutura. Para essa fase a ordem de passos avaliativos continua a ser aplicada de acordo com os algoritmos descritos na seção de

metodologia (5.2) deste trabalho. Mostramos abaixo um resumo dos resultados do processo de avaliação do alinhamento baseado em estruturas.

Ativa-se processo de alinhamento baseado em estrutura, isso pode ser feito diretamente no sistema ou pode acontecer como mais uma fase do processo total. Como resultado inicial dessa fase de alinhamento temos a seguinte lista (TAB. 47) de classes e subclasses candidatas ao alinhamento final:

TABELA 47
Lista inicial de candidatas da fase de estrutura

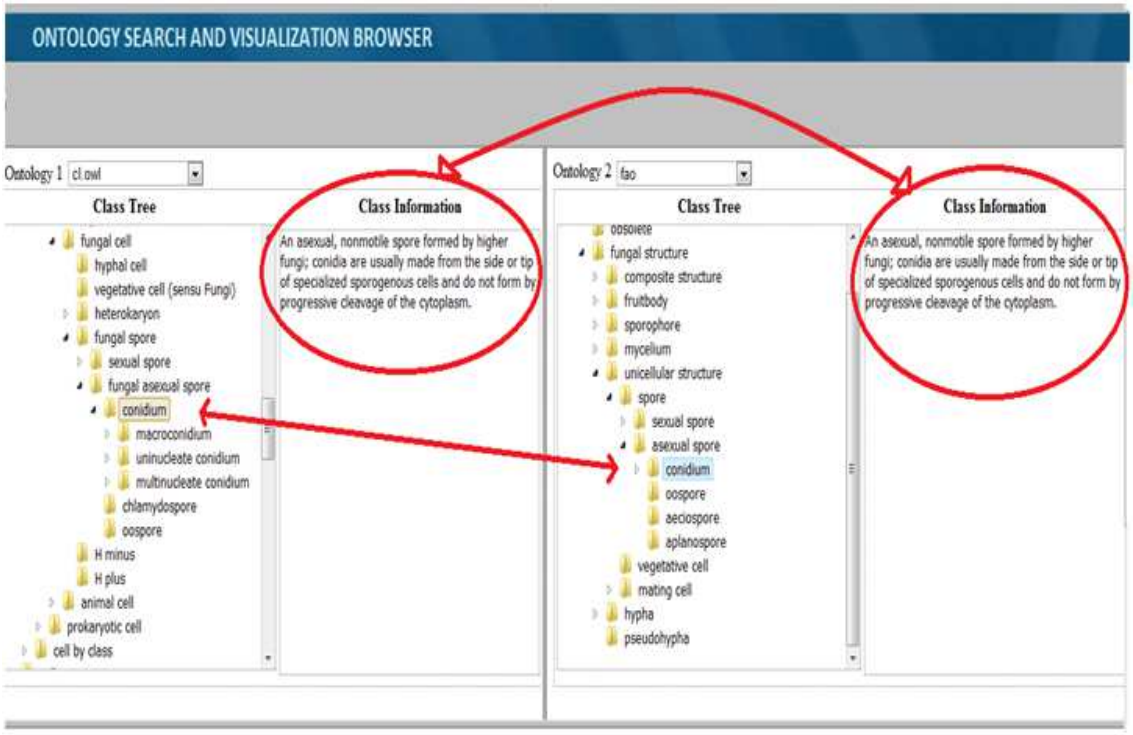
Cell		Equivalência	FAO	
Classe	SubClasse		Classe	SubClasse
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Boa	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Boa	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]
Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Boa	Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]
fungal cell	[H minus; H plus; fungal spore; heterokaryon; hyphal cell; vegetative cell (sensu Fungi)]	Boa	unicelluar structre	[mating cell; spore; vegetative cell]
Hyphal Cell	[]	Ruim	Hypha	[Hulle cell; aseptate hypha; conidiophore; hypha in mycelium; hyphal tip; oidium; septate hypha]

Fonte: Autoria própria

As listas de classes retornadas foram analisadas com o suporte das ferramentas de visualização. Para cada par de candidatas, uma busca na ferramenta de visualização é feita. Após alcançar cada conjunto de entidades apresentado verifica-se a descrição destas e, com isso, avaliaram-se as notas retornadas pelo sistema. A FIG. 51 mostra as duas ontologias (CELL e FAO) e a descrição das classes selecionadas. A avaliação das notas de cada par de entidades retornadas na TAB. 47 é feita através da averiguação das informações apresentadas

no campo *Class Information*. Quantificou-se as avaliações de acordo com as métricas dadas nas seções 5.1 e 5.3 do cap 5.

FIGURA 51- Figura de comparação de duas ontologias



Fonte: Autoria própria

Aqui a análise das estruturas apresentadas na FIG. 51 visou à comparação das descrições, desta forma dando continuidade aos procedimentos necessários para a realização da avaliação. Além disso, a visualização da estrutura e das informações também permitiu pontuar de forma mais confiável a avaliação de completude dos dados modelados na ontologia, isso em relação à fase anterior. A análise seguiu para todas as entidades retornadas e gerou a TAB. 48 (auxiliar), na qual as marcações “sim” ou “não” foram feitas de acordo com a seção 5.1 do capítulo 5:

TABELA 48
Tabela de avaliação da primeira fase de estrutura

Estrutura fase 1						
Cell		Equivalência	FAQ			
Classe	SubClasse		Classe	SubClasse	Nota	Candidata
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Boa	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	4	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Boa	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	4	Sim
Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Boa	Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	4	Sim
multinucleate conidium	[multinucleate macroconidium]	Boa	multinucleate conidium	[multinucleate macroconidium]	4	Sim

Fonte: Autoria própria

O resultado acima foi informado ao sistema (FIG. 37) que gerou a primeira estrutura preliminar alinhada. A FIG. 52 mostra o resultado do alinhamento, ontologia *cl_fao.owl* a esquerda, a ontologia CELL (*cl.owl*) ao meio e a ontologia FAO (*fao.owl*) a direita.

FIGURA 52- Visualização da comparação da semi ontologia gerada com as ontologias avaliadas

Fonte: Autoria própria

Para criação da tabela de avaliação (TAB. 49) da primeira etapa da fase de alinhamento baseado em estrutura, a comparação de cada conjunto de candidatas ao alinhamento foi feito através da inspeção visual. Para isso utilizaram-se não só as representações apresentadas na FIG. 52, mas também as informações representadas na FIG. 51. A tabela de avaliação gerada pela análise da estrutura acima somada às tabelas auxiliares foi a TAB.49 a seguir:

TABELA 49
Sumarização Avaliação X Inspeção Visual

Tabela Auxiliar de Avaliação

Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude completude	Avaliação Final
multinucleate conidium	[multinucleate macroconidium/ multinucleate macroconidium fao]	5	4	3	4
uninucleate macroconidium	[uninucleate arthroconidium / uninucleate arthroconidium fao; uninucleate blastoconidium/ uninucleate blastoconidium fao]	5	4	4	4
fungal asexual spore	[chlamydospore/ chlamydospore fao; conidium conidium fao; oospore/ oospore fao]	5	5	5	5

+	+	+	+	+	+
+	+	+	+	+	+
+	+	+	+	+	+

Fonte: Autoria própria

Para a confirmação das notas dadas pelo sistema durante a primeira etapa da fase de alinhamento baseado em estrutura foi analisado a semi ontologia criada em relação às ontologias base. E além da comparação direta das descrições, o conhecimento que o avaliador tem sobre o domínio e o desenho das ontologias na tela foram cruciais para o entendimento e categorização da avaliação.

A segunda etapa do alinhamento baseado em estrutura, como na segunda fase do alinhamento baseado em *String*, tem metrificações mais sólidas e menos abertas à interpretações. Aqui resumiremos os resultados retornados em forma de lista de candidatas ao alinhamento (TAB.. 50).

TABELA 50 –
Resultado da segunda iteração alinhamento estrutura

FAO		Equivalência	CELL	
Classe	Subclasse		Classe	Subclasse
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]
Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Ótima	Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate macroconidium]
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Ótima	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]
Conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Ótima	Conidium	[macroconidium; multinucleate conidium; uninucleate conidium]
fungal asexual spore	[chlamydospore; conidium; oospore]	Média	fungal asexual spore	[aeciospore; aplanospore; conidium; oospore]
fungal spore	[fungal asexual spore; sexual spore]	Média	fungal spore	[asexual spore; sexual spore]

Fonte: Autoria própria

Para a avaliação dos resultados retornados na TAB. 50, como na primeira etapa da fase de alinhamento estrutural, é feita a comparação de cada conjunto de candidatas ao alinhamento através da inspeção visual. A TAB. 51 abaixo mostra a pontuação de parte das entidades retornadas.

TABELA 51
Pontuação de alinhamento dado pelo usuário

Estrutura fase 2						
Cell		Equivalência	FAQ			
Classe	SubClasse		Classe	SubClasse	Nota	Candidata
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	5	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	5	Sim
Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Ótima	Macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	5	Sim
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate macroconidium]	5	Sim
fungal asexual spore	[chlamydospore; conidium; oospore]	Média	fungal asexual spore	[aeciospore; aplanospore; conidium; oospore]	4	Sim
fungal spore	[fungal asexual spore; sexual spore]	Média	fungal spore	[asexual spore; sexual spore]	4	Sim

Fonte: Autoria própria

Os dados da TAB.51 foram informados ao sistema e uma semiontologia similar à FIG. 52 foi construída. Aqui também analisou-se as listas de entidades, para cada par de entidades uma comparação utilizando a visualização em árvore para análise dos nomes. Ao final quantificaram-se as avaliações (TAB. 52) de acordo com as métricas dadas nas seções 5.1 e 5.3 do cap 5.

TABELA 52
Pontuação segunda fase - Usuário

Tabela Auxiliar de Avaliação

Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude completude	Avaliação Final
fungal spore	[fungal asexual spore/ asexual spore fao; sexual spore/ sexual spore fao]	5	4	3	4
multinucleate conidium	[multinucleate macroconidium / multinucleate macroconidium fao]	5	4	4	5
fungal asexual spore	[chlamydospore/ chlamydospore fao; conidium conidium fao; oospore/ oospore fao]	5	5	5	4
+	+	+	+	+	+
+	+	+	+	+	+
+	+	+	+	+	+

Fonte: Autoria própria

A sumarização da avaliação final dessa fase de alinhamento, que foi feita com análise das estruturas somada a aplicação de média simples nas notas retornadas pelo sistema em cada uma das fases, resultou no seguinte formato (TAB. 53):

TABELA 53
Tabela de Avaliação Final

Classes/Subclasses		Fase 1		Fase 2		Resultados	
CELL X FAO		Pontuação	Candidatas	Pontuação	Candidata	Pontuação	Candidata
	[fungal asexual spore/ asexual spore fao; sexual spore/ sexual spore fao]	4	Sim	4	Sim	4,0	Sim
fungal spore							
multinucleate conidium	[multinucleate macroconidium / multinucleate macroconidium fao]	4	Sim	5	Sim	4,5	Sim
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	5	Sim	4	Sim	4,5	Sim

Fonte: Autoria própria

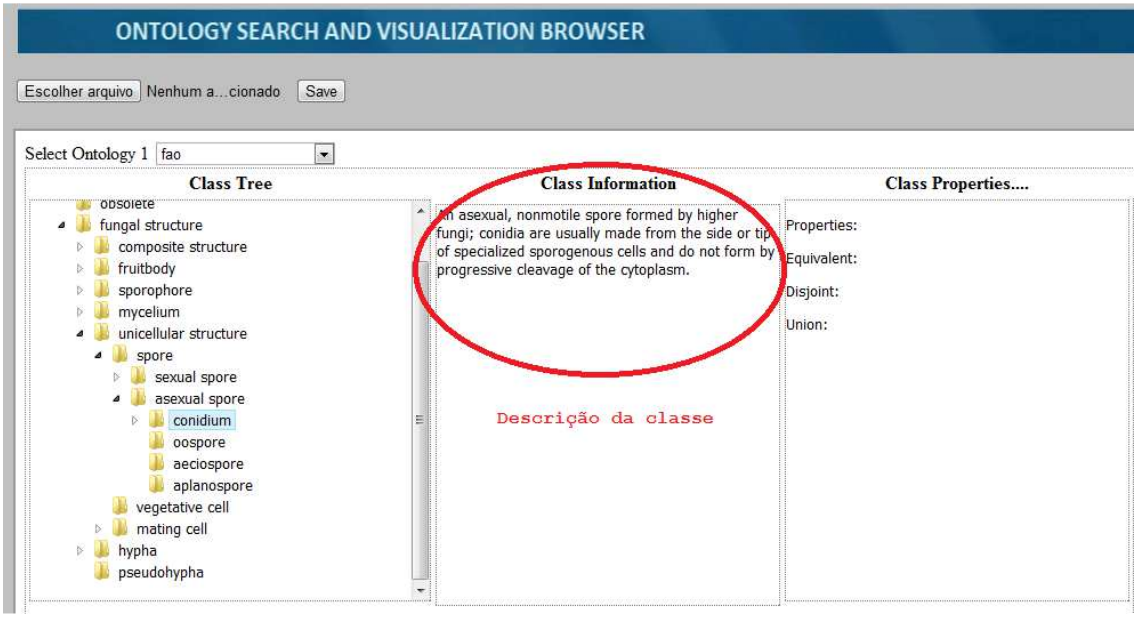
O fechamento preliminar de cada fase é utilizado para um fechamento final que nos levou a uma nota geral de avaliação. A seguir, na seção 6.2.2, são apresentados os resultados da avaliação baseados nos dados apresentados anteriormente.

6.2.2 - Avaliação

As métricas da avaliação final aplicadas foram baseadas nas notas dados ao longo do processo de avaliação dos alinhamentos somados a nota dada a completude das informações de cada um dos pares de entidades candidatas ao alinhamento final. Por exemplo, para entidade *conidium*: além de utilizar as notas de alinhamento referentes a ela nos processos de alinhamento baseados em nome e baseados em estrutura, averiguamos a completude das descrições (FIG. 53) e o nível de detalhamento das propriedades (FIG. 54). As notas para cada uma desses parâmetros (descrições e propriedades) foram dadas para a sumarização final da avaliação.

Essa avaliação de completude foi feita ao longo de todo o processo, então para cada passo dado uma tabela auxiliar (TAB. 54) de verificação das informações modeladas nas ontologias foi criada. O procedimento adotado foi esse, pois cada fase de alinhamento pode gerar pares candidatos ao alinhamento diferentes.

FIGURA 53 - Suporte avaliação de descrições

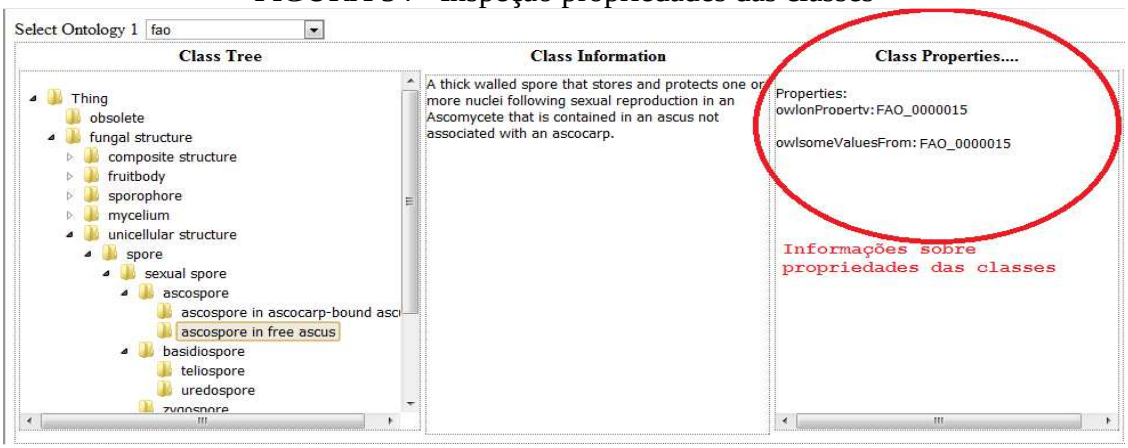


Fonte: Autoria própria

A FIG. 53 mostra o detalhamento das informações da classe *conidium* da ontologia FAO. Ela é utilizada dentro da avaliação de completude de informações de cada uma das ontologias. Utilizando esse suporte, as tabelas auxiliares de avaliação são preenchidas. Para utilização desse recurso, o avaliador utiliza a navegação dentro da ontologia que o sistema proporciona, somado ao seu conhecimento sobre o domínio.

A FIG. 54 mostra, além da descrição da classe, as propriedades que cada uma comporta. Neste caso específico podemos constatar que a classe *ascospore in free ascus* herda suas propriedades e valores de outra classe especificada nas propriedades. Isso é usado para avaliar o nível de detalhamento que cada ontologia traz para suas classes.

FIGURA 54 - Inspeção propriedades das classes



Fonte: Autoria própria

O resultado da aplicação das métricas de avaliação, relativas à completude e propriedades, à fase de alinhamento baseada baseada em estrutura, são demonstrados na TAB. 54:

TABELA 54
Avaliação de completude dos dados das classes - estrutura

Tabela Auxiliar de Avaliação

CELL/FAO		Nota
Basidiospore	Basidiospore	5
Zygospore	Zygospore	5
Hypa multinucleate	Hypa multinucleate	3
arthroconidium uninucleate	arthroconidium	4
arthroconidium multinucleate	uninucleate arthroconidium multinucleate	5
blastoconidium uninucleate	blastoconidium	4
blastconidium multinucleate	uninucleate blastconidium multinucleate	5
macroconidium multinucleate	macroconidium multinucleate	5
blastoconidium uninucleate	blastoconidium uninucleate	4
macroconidium	macroconidium	4
+	+	+
+	+	+
+	+	+

Fonte: Autoria própria

Após todas as etapas de avaliação das ontologias CELL e FAO completas, as

notas são compiladas em uma tabela de avaliação final, com cada par de classe e uma geral. E desta, uma nota geral foi calculada, fazendo-se a média simples entre as notas de cada fase do processo. Para cálculo da nota final dessa tabela auxiliar temos a soma de cada um dos fatores (Corretude de alinhamento, Corretude de Descricao e Corretude de Estrutura) dividida por 3, ou seja, a média simples das notas dadas a cada um dos fatores considerados. Por exemplo: o alinhamento entre *multinucleate macroconidium* e *multinucleate macroconidium fao* recebeu nota 4, a corretude da estrutura recebeu nota 4 e as corretudes das descrições nota 5, a avaliação final desse par de entiades foi dado por (4+4+5)/3. Todas as quantificações foram feitas entre 1 e 5 (TAB. 55).

TABELA 55
Sumarização das Pontuações

Tabela de Avaliação – Par de Entidades					
Classes/Subclasses		Corretude Alinhamento	Corretude Estrutura	Corretude Descrição	Avaliação Final
multinucleate conidium	[multinucleate macroconidium/ multinucleate macroconidium fao]	4	4	5	4.3
uninucleate macroconidium	[uninucleate arthroconidium / uninucleate arthroconidium fao; uninucleate blastoconidium/ uninucleate blastoconidium fao]	5	4	5	4.6
+	+	+	+	+	+
+	+	+	+	+	+
+	+	+	+	+	+

Fonte: Autoria própria

A tabela final de avaliação (TAB. 56) foi obtida com a média simples entre as notas dadas para todos os pares de entidades de cada um dos parâmetros relacionados acima (alinhamento, descrição e estrutura). Para Corretude de alinhamento somamos todas as notas dadas aos conjuntos de entidades e dividimos pelo total de conjuntos por exemplo: colocamos a nota dos parâmetros como sendo A, sua média como sendo M e o número de conjuntos avaliados como sendo C, entao temos que o resultado final é:

$$\mathbf{M} = \frac{\mathbf{1}}{C} \sum_{n=0}^{C} \mathbf{A}_n$$

para cada um dos parêmetros aplicados.

TABELA 56 –
Compilação final das notas do processo

Compilação Final		
[redacted]	CELL	FAO
Corretude Alinhamento	4,5	
Corretude Estrutura	4,2	5
Corretude Descrição	4,3	4,6
Avaliação Final	4,3	4,7

Fonte: Autoria própria

Após o resumo das notas dadas às avaliações das ontologias em tratamento, alguns dados relevantes de alinhamento são compilados a seguir. Esses dados são: total de classes das duas ontologias, classes não consideradas para o alinhamento, total de pares de classes alinhadas e, por fim, total de pares de classes descartadas do alinhamento.

- Total de termos ontologia CELL: 2003;
- Total de termos ontologia FAO: 81;
- Classes não consideradas para o alinhamento ontologia CELL: 1906;
- Classes não consideradas para o alinhamento ontologia FAO: 0;
- Total de classes alinhadas 78;
- Total de classes descartadas: 19;

Aqui temos que aproximadamente 80% das classes indicadas para o alinhamento foram efetivamente alinhadas. Também temos um grande número de classes da ontologia CELL desconsideradas para o alinhamento; isso se deve ao fato da ontologia CELL ter grande quantidades de termos que não têm concordância em nenhum nível com a ontologia FAO, pois são descrições de outros tipos de celular que não fungos.

Um fato que deve ser marcado é a alta porcentagem de classes alinhadas em relação ao número de classes da ontologia FAO (~96,5%), já era esperado um número grande de não concordâncias por parte da CELL, pois essa modela outros tipos de células que nao fungos. No entanto, a grande quantidade de classes provindas da ontologia FAO alinhadas pode ser um indicativo de qualidade dos algoritmos aplicados ao alinhamento semi automático.

No próximo capítulo apresentamos, além das conclusões finais, uma discussão sobre os dados e o processo aplicado.

7 – Considerações finais.

As complexas atividades relacionadas à produção e disseminação de conhecimento geram a necessidade do tratamento da informação produzida, especialmente àquela referente aos processos internos de uma instituição. Também surge a necessidade do controle do que é produzido pela organização. Por exemplo, em uma empresa de grande porte de geração de energia, além do controle das demandas de geração e transmissão de energia existe ainda a necessidade de controle dos processos internos para manutenção da competitividade. Isso inclui na reflexão sobre como tratar e disseminar informação.

De fato, tanto as atividades e as relações de produção, quanto o contexto e as relações sociais, carecem de informações acessíveis, pressupondo a necessidade de organização das informações. Ferramentas de classificação e organização da informação, em última instância, atendem a necessidades dos usuários independentemente do quão complexo pode ser essa ferramenta em termos computacionais. O que importa na verdade é o sucesso ou insucesso do usuário em sua busca. Um aumento no volume de dados sem contextualização e tratamento leva à sobrecarga de informação, quando o usuário não consegue mais dar sentido ao conjunto de dados recuperado.

Mesmo a complexidade das ferramentas, sejam elas computacionais ou de outra natureza, não sendo o principal objetivo do tratamento de informação, e sim o meio, é necessário que essas sejam produzidas da melhor maneira possível. Um dos grandes problemas encontrados no que diz respeito à produção e manutenção de ferramentas, principalmente computacionais, se situa na manutenção e modelagem das suas bases de dados.

Desta forma, temos como um dos grandes filões a serem tratados dentro do escopo de controle e processamento de informações, a modelagem e armazenamento. A partir destes pontos temos toda uma cadeia de dependências computacionais e informacionais que são atendidas pelas bases de dados. Isso torna a modelagem dos dados que será utilizado pelos sistemas um ponto crítico a ser levado em conta.

Outro ponto que deve ser levado em pauta sobre os processos de tratamento e utilização de informação é a capacidade de disseminação dessas. A sua comunicação não só com usuários, mas também entre os sistemas que se encarregam do seu tratamento, deve ter

especial atenção. Isso se deve a necessidade de interoperabilidade entre esses e compreensão dos dados por humanos.

O quesito interoperabilidade pode ter grande ajuda em um trabalho mais intenso de modelagem das informações, pois essas, que têm recebido grande atenção em pesquisas sobre modelagem de dados, são utilizadas na organização da informação, integração de fontes heterogêneas, dentre outras. Além disso, a necessidade de ontologias pode ser vista como consequência desse crescente volume de informação, praticamente exigindo a consideração de melhorias nas ferramentas da tecnologia da informação.

Então podemos encontrar grande suporte a interoperabilidade de dados nas ontologias, pois essas permitem correta modelagem das estruturas e informações dentro dos sistemas. Fazendo com que, dessa forma, os sistemas que necessitam dos mesmos tipos de dados possam se comunicar sem grandes problemas.

O requisito para utilização de uma ontologia na modelagem e interoperabilidade dos dados é sua correta modelagem. Como, a construção de uma ontologia é complexa e quase artesanal (ALMEIDA, 2009), formas de facilitar essa construção são importantes. Como, dentro do processo de construção está contido o processo de avaliação (GOMÈZ PERES, 2001) trabalhar no sentido de auxilio à avaliação pode facilitar o processo como um todo.

O processo de avaliação de ontologias esbarra em barreiras de complexidade, precisão e tamanho. Como uma ontologia pode ser utilizada para a modelagem de dados complexos a análise dessa modelagem também se torna complexa. Para uma grande precisão na avaliação a interferência humana é de grande importância, no entanto, essa interferência torna o processo lento. Desta forma, temos um impasse dentro da necessidade de avaliação. Fazer avaliação de forma automática que é mais rápida, porém menos precisa, ou fazer uma avaliação de forma artesanal.

Entre os tipos de processo de avaliação tivemos especial atenção com os baseados em alinhamento e análise humana. O primeiro pelo caráter automático e a segunda pela precisão na avaliação que proporciona.

Na avaliação realizada por agentes humanos temos grande variedade de pessoas e habilidades envolvidas, e nem sempre todos os envolvidos tem os conhecimentos necessários para leitura de linguagens de marcação. Desse fato nasce mais uma necessidade dentro do contexto de modelagem e avaliação de ontologias. Uma forma de facilitar o entendimento do que já foi modelado dentro das ontologias.

Para o suporte ao entendimento dos modelos já implementados, a utilização de maneiras de transformá-los em metáforas visuais é uma saída de ampla utilização. Essas não somente utilizadas pra ontologias, mas para várias áreas do conhecimento.

Já a avaliação baseada em alinhamento de ontologias foi considerada pelo seu caráter, na maioria das vezes, automatizado. Quando temos o trabalho de alinhamento entre ontologias, não só a avaliação dos dados e descrições pode ser feita, mas também a análise de sua capacidade de intercomunicação entre ontologias e outros sistemas.

Esses fatores levam também a compreensão de que uma forma intermediária pode ser utilizada para o processo de construção e avaliação de ontologias. Uma forma que dê suporte a atividade humana no processo, mas que traga informações “mastigadas” para subsídio do processo de análise de uma ontologia em questão.

Para uma forma intermediária efetiva, mais de um tipo de tratamento e alinhamento de ontologias pode ser utilizado. Dessa forma, podem-se concatenar as vantagens dessas formas para a criação de um método semiautomático passível de ser usado não somente por especialistas em ontologias, mas também por outros participantes envolvidos no processo.

O método desenvolvido nesse trabalho leva em conta os três fatores citados acima: avaliação automática, avaliação artesanal e compreensão dos dados. Para cada um dos pontos foi criada uma estratégia que possibilite a avaliação; para automática o alinhamento, para artesanal a completude e para compreensão e suporte a visualização de informação.

O método colocado por esse trabalho visa à junção dos três pontos colocados a um sistema de métricas simples. Os dois pontos iniciais sendo partes do desenvolvimento junto com as métricas e o terceiro ponto o suporte a processo como um todo.

Para que o processo de avaliação retornasse um resultado mais palpável para o avaliador, optou-se por colocar métricas em suas fases. Dessa forma, temos no final um valor simples representante do quão “boa” a ontologia em avaliação é, além de uma nota para, o que pode ser considerado, sua interoperabilidade com uma ontologia pai.

Métricas simples foram desenvolvidas, pois para utilizá-las é necessário conjugar os resultados automáticos com análise subjetiva dos processos. Então, desta forma, temos valores aplicados aos processos de fácil interpretação e análise. Com isso temos intenção de permitir que avaliação possa ser feita por especialistas de qualquer área de domínio, na qual as ontologias foram implementadas, e não somente por um engenheiro de ontologias ou especialistas em informação.

A utilização das métricas em duas frentes distintas faz com que elas sejam mais confiáveis, pois a análise do avaliador é feita em cima de resultados automáticos de comparação. A redução da quantidade de dados facilita a investigação deles pelo avaliador, podendo, dessa forma, reduzir a quantidade de erros de julgamento que poderiam existir. Também pode ser utilizada como redutor de erros essa análise prévia que os algoritmos dos sistemas fazem nas ontologias participantes do processo.

Ao final do processo, temos valores sumarizados de cada parte do processo. Como cada etapa do processo é valorada, as avaliações podem ser feitas de forma particionadas ou uma avaliação baseada nos valores finais ou mesmo em uma única média final de notas é possível através dessas métricas simplificadas.

O alinhamento de ontologias foi escolhido como participante do método, pois permite que as métricas desenvolvidas sejam aplicadas automaticamente às ontologias envolvidas. Isso feito par a par das entidades candidatas ao alinhamento permitiu análise detalhada de um número representativo dos dados modelados. E através da aplicação dos algoritmos têm-se métricas prévias para auxilio das avaliações subjetivas.

Além do citado, temos como vantagem na escolha de métodos de alinhamento como base avaliativa o reconhecimento de interoperabilidade entre as ontologias envolvidas. No caso das ontologias escolhidas, ontologias mais especializadas devem ser passíveis de mapeamento dentro da ontologia mais genérica. Nesse caso, deve se possível mapear a ontologia específica de fungos junto aos dados de uma ontologia de representação de tipos de células.

Outro ponto que levou à escolha e à avaliação comparativa, ou seja, temos uma ontologia que está em processo de avaliação, então escolhemos uma ontologia base para aplicação do processo e, desta forma, comparação dos dados da ontologia avaliada com a ontologia base. Para isso, a ontologia de base para comparação foi escolhida de um projeto grande e consolidado dentro da área de desenvolvimento de ontologias. Desta forma conseguimos certa confiança nos dados de comparação.

Entre os resultados dos processos de alinhamento conseguimos definir pontos de convergência e divergência, a possibilidade de compatibilidade e análise das informações modeladas.

Todos os processos têm como suporte técnicas de visualização de informação para suporte às avaliações subjetivas. Isso se deve à necessidade de suporte ao entendimento dos modelos por pessoas não treinadas no desenvolvimento de ontologias.

Para o processo de suporte transformou-se as ontologias em alguns tipos de árvores que são desenhadas como tradução dos dados modelados dentro das ontologias em grafos e apresentados pelo sistema criado para o alinhamento e visualização de ontologias.

A união das técnicas de avaliação subjetiva e objetiva foi possibilitada pelo acompanhamento visual dos processos automáticos de alinhamento pelos avaliadores e "[...] tendo em vista que toda técnica de visualização é destinada a apoiar a realização de tarefas de análise de dados, é abordada a questão de caracterização dessas tarefas e dos mecanismos de interação necessários para suportá-las" (LUZZARDI, 2002). Utilizamos técnicas de visualização como viabilizadoras das avaliações cruzadas.

O método visou cobrir partes das avaliações de ontologias tais como:

- Garantir a modelagem correta de um domínio, não somente em relação às descrições e propriedades;
- Avaliação em termos hierárquicos, garantia dos corretos modelos hierárquicos, ou seja, felino tem que estar abaixo de mamífero em uma ontologia de modelagem de mamíferos;
- Interoperabilidade de ontologias, verificação se uma ontologia "filho" consegue ser corretamente mapeada em uma ontologia base;
- Corretude.
- Entre as bases de avaliação descritas por Brank, Grobelnik e Mladeni`c (2006):
- Aquelas baseadas na comparação da ontologia a um "*Gold Standard*" (MAEDCHE; STAAB, 2002);
- Aquelas baseadas no uso da ontologia em uma aplicação e a avaliação dos resultados (PORZEL; MALAKA, 2004);
- Aquelas envolvendo comparações com uma fonte de dados, uma coleção de documentos, por exemplo, sobre o domínio a ser coberto pela ontologia (BREWSTER et al, 2004);
- Aquelas onde avaliação é feitas por pessoas que tentam averiguar quão bem a ontologia atinge um conjunto pré-definido de critérios, padrões, requisitos (LOZANO-TELIO; GÓMEZ PÉREZ, 2004).

Cobrimos, em algum nível, os cinco pontos citados acima. Isso foi possível através da mescla dos processos de visualização, alinhamento e avaliação.

Isso permitiu a sumarização quantitativa da avaliação semiautomática que é colocada como função das métricas, pois permitiu o resultado geral de avaliação que é representado por um número absoluto no final de cada ponto do processo de avaliação total.

Todo o processo teve como suporte as ferramentas de tratamento de informação criadas. A necessidade de criação de ferramentas se baseou na possibilidade de criar sistemas que cruzassem as técnicas de visualização com os tipos de alinhamentos desejados para a construção do processo como um todo. Esse processo pode ser aplicado utilizando-se várias ferramentas diferentes somadas a outros suportes (como editores de texto e editores de ontologias), mas essa produção de sistemas permitiu diminuir a necessidade de utilização de ferramentas externas. No entanto, a ferramenta cobre boa parte dessas ferramentas externas, pois ela permite retorno da avaliação subjetiva, importa entidades candidatas em uma única ontologias e permite quatro tipos de visualização diferentes, que são suporte às análises subjetivas.

Temos por fim a compilação do processo para que o objetivo de um método único fosse cumprido. Isso foi feito através da criação de roteiros que, além de utilizarem as ferramentas criadas, guiam todas as fases de avaliação. Com a utilização de roteiros guias é possível fazer as inferências subjetivas necessárias sobre os resultados automáticos dados pelo sistema de suporte criado.

Então, dessa forma, temos como resultado final o método unificado que mescla avaliação subjetiva com avaliação semiautomática de alinhamentos de ontologias. Os processos descritos, principalmente no que diz respeito às métricas, podem ser modificados para atender outras demandas. Podem-se modificar de forma simples os algoritmos criados para que eles contemplem outros tipos de avaliação além de hierárquicas baseadas em descrições e *matching* baseados em nomes.

Por fim, pode-se considerar que a essência do método de avaliação proposto é a comparação *gold standard* e avaliações feitas por pessoas. No entanto, ao invés da comparação direta entre duas ontologias analisando diretamente suas descrições ou outro fator qualquer, pegamos como base a capacidade de uma ontologia “menor” se encaixar em uma “maior”. Isso quer dizer, que utilizamos a capacidade de comunicação entre as ontologias para avaliar sua construção.

Assumindo que uma ontologia mais especializada construída sob o mesmo padrão de uma ontologia mais genérica deve ter a capacidade de comunicação com esta. Comunicação tal que além de baseada em atributos, pode ser avaliada por descrições, nome e estrutura. Pode-se dizer que a constatação dessa interoperabilidade, através da análise de suas

descrições e estruturas, mostra qualidade na construção em relação aos parâmetros adotados para sua implementação. Assim, evidenciando que ela foi bem feita dentro dos padrões esperados.

Desta forma temos uma vantagem mais explicita no método, pois esse, além de ter passos para avaliação dos dados das ontologias em si, nos traz formas de avaliar sua intercomunicabilidade. E através disso, sua capacidade de disseminação e comunicação com padrões consolidados.

Também pode se inferir que essa avaliação vai além das formas *gold standard* verificadas, pois transforma a comparação com *gold standard* em uma forma de voltada a tarefas. Essa tarefa neste caso é: uma ontologia deve se ligar a outra, se isso for alcançado de forma satisfatória, mais um parâmetro de qualidade citado na literatura foi alcançado.

Assim, tem-se uma série de padrões de qualidade que podem ser verificados através do método proposto neste trabalho. Corretude das informações e atributos através da comparação direta através das validações visuais, corretude das estruturas através dos alinhamentos, interoperabilidade entre duas ontologias através de seu alinhamento e de uma tarefa específica aplicada.

Um fator que é importante frisar é a aplicabilidade do método em quaisquer pares de ontologias que atendam aos pré-requisitcos colocados no capítulo 5 deste trabalho. Desde que dentro do mesmo padrão de descrição e modelagem, duas ontologias podem ser carregadas dentro dos sistemas e os processos propostos podem ser utilizados. Isso pode ser feito quando se assume que duas ontologias, que visam à descrição de um mesmo domínio e estão sob um mesmo padrão ontológico, devem se comunicar, além de ter estruturas e descrições próximas para classes semelhantes.

No caso dos dados apresentados neste trabalho, temos essas características de comparação bem evidenciadas. Durante o processo de avaliação, tomou-se como base a ontologia CELL, aqui considerada a ontologia mais genérica, no que diz respeito às comparações *gold standard*. As avaliações mostraram a consistência de nome e descrições entre as duas ontologias. Em termos gerais, as descrições contidas na ontologia FAO, considerada a ontologia mais específica, foram consistentes com as descrições das célula de fungos contindas na ontologia CELL.

As relações estruturais também foram avaliadas através do alinhamento e da criação da semi ontologia alinhada, desta forma, por exemplo, foi possível verificar se duas classes semelhantes estavam sob a mesma classe pai. Essas avaliações mostram consistência na modelagem de relações entre as duas ontologias.

Os resultados em relação ao número de classes alinhadas e o número de classes totais das duas ontologias avaliadas caminhou dentro do esperado. A ontologia mais genérica teve menor quantidade de classes alinhadas, isso já era esperado, pois entre os dados de entrada escolhidos, a ontologia CELL e a ontologia FAO, a ontologia CELL abrange grande quantidade de descrições de outros tipos de céluas que não fungos.

Um fato que deve ser marcado é a alta porcentagem de classes alinhadas em relação ao número de classes da ontologia FAO (~96,5%). A grande quantidade de classes provindas da ontologia FAO alinhadas é considerado um indicativo de qualidade da ontologia e dos algoritmos aplicados ao alinhamento semi automático.

Isso é considerado como indicativo de qualidade dos algoritmos, tanto computacionais como processos, porque, como o estudo de caso foi feito em cima de duas ontologias de projetos consolidado de mesma base, seu bom alinhamento e qualidade de descrições era esperado. A necessidade de se escolher ontologias de qualidade estava no objetivo de avaliação e validação do método proposto. No entando, o fato de duas ontologias consolidadas terem sido escolhidas não retira a qualidade dos resultados avaliativos em si, mas os corrobora.

8 – Bibliografia

ALANI, H. A Touchgraph Visualization Tab for Protégé 2000. 2 nd INTERNATIONAL CONFERENCE ON KNOWLEDGE CAPTURE, 2., 2003, Florida, USA., [**Proceedings…**]. Flórida, USA, 2003.

ALBUQUERQUE, A. C. F.; CAMPOS DOS SANTOS, J. L.; MAGALHÃES NETTO, J. F. "Modeling Complex Domain Ontology Based on the Unified Foundational Ontology". EXTENDED PROCEEDINGS OF THE 4TH. LATIN AMERICAN CONFERENCE ON COMPUTER HUMAN INTERACTION IN CONJUNCTION WITH THE 7TH. LATIN AMERICAN WEB CONGRESS, 4., 7., 2009, Ensenada, B. C., [**Proceedings...**]. Ensenada, B. C.: CLIHC; LAWEB, 2009.

ALMEIDA, M.B. A proposal to evaluate ontology content. **Applied Ontology**, Birmingham, v. 4, n. 3-4, p. 245-265, 2009.

AUMÜLLER, D., DO, H.H., MASSMANN, S., RAHM, E: Schema and Ontology Matching with COMA++. Proc. SIGMOD, 2005, [Software Demonstration], Baltimore, June, 2005.

BATES, M. The invisible substrate of information science. **Journal of American Society of Information Science and Technology**, v. 50, n. 12, 1999.

BICER , V.; LALECI; GOKCE, B., DOGAC, A.; KABAK, Y. Artemis message exchange framework: semantic interoperability of exchanged messages in the healthcare domain, SIGMOD Rec., 34, 3., Sept., 2005.

BORKO, H. Information science: what is it? **American Documentation**, v. 19, n.1, p. 3-5, 1968.

BRANK, J.; GROBELNIK, M.; MLADENI´c, D. A survey of ontology evaluation techniques. PROCEEDINGS OF THE CONFERENCE ON DATA MINING AND DATA WAREHOUSES, Ljubljana, Sloveni, 2006, [**Proceedings ….**]. Ljubljana, Sloveni, 2006. Disponível em: http://lorcancoyle.org/files/Ye2007Ontology-.pdf>. Acesso em: 01out.2010.

BRASILESCOLA. 2010. Disponível em: < http://www.brasilescola.com>. Acesso em 14 out. 2010.

BREWSTER, C. et al. Data driven ontology evaluation. INTERNATIONAL CONFERENCE ON LANGUAGE RESOURCES AND EVALUATION, Lisboa, 2004. Disponível em: <http://http://eprints.aktors.org/337/02/BrewsterLREC-final.pdf>. Acesso em: 22 fev. 2010.

BUSSLER, C., FENSEL, D., MAEDCHE, A. Conceptual Architecture for Semantic Web Enabled Web Services. SIGMOD Rec., v. 31, n. 4, dez. 2002.

CARVALHO, E. S.; MARCOS, A. F. Visualização de informação. **Centro de Computação Gráfica**, 2009. Disponível em:<http://hdl.handle.net/1822/8863>. Acesso em: 22 out.2010.

CASTANO, S., FERRARA, A., MONTANELLI, S.; ZUCCHELLI , D. HELIOS: A General Framework for Ontology-based Knowledge Sharing and Evolution in P2P Systems. 14th INTERNATIONAL WORKSHOP ON DATABASE AND EXPERT SYSTEMS APPLICATIONS, 14., 2003, [**Proceedings…**]. [s.l], 2003.

CERVO, A. L.; BERVIAN, P. A. **Metodologia científica**. São Paulo: Makron Books, 1996.

CHEN, B., TAN, H; LAMBRIX, P. "Structure-Based Filtering for Ontology Alignment," Enabling Technologies. 15th IEEE INTERNATIONAL WORKSHOPS ON ENABLING TECHNOLOGIES: INFRASTRUCTURE FOR COLLABORATIVE ENTERPRISES, 15, 2006, (WETICE'06), 2006. pp. 364-369

CHERNOFF, H. The Use of faces to Represent Points in K-Dimensional Space Graphically. Journal Amer. Statistical Associatio, New Yorke, USA, v. 68, p. 361- 368, 1973

CHISEL COMPUTER HUMAN INTERATION &SOFTWARE ENGINEERING LAB. Shrimp User Manual. [s.l], 2010. http://www.thechiselgroup.org/shrimp

CORCHO, Ó. et al. Odeval: A tool for evaluating rdf(s), daml+oil and owl concept taxonomies. In: AIAI. [S.l.: s.n.], 2004. p. 369–382.

DIAS, M. P.; CARVALHO, F. J. A visualização da informação e a sua contribuição para a ciência da informação. International Conference on Language Resources and Evaluation, Brasill, 2007. Disponível em: <http://www.dgz.org.br/out07/Art 02.htm>.

DOAN, J; MADHAVAN, P. DOMINGOS e HALEVY, A. Ontology matching: A machine learning approach. em S. Staab and R. Studer, editores, Handbook on Ontologies in Information Systems. Springer-Velag, 2003.

DUVAL, R. Approche cognitive des problèmes de géométrie en termes de congruence. Annales de Didactique et de Sciences Cognitives. IREM de Strabourg, 1988.

EDEN. 2010. Disponível em: <http://eden.dei.uc.pt> . Acesso em 14 out. 2010.

EHRIG e STAAB . Ontology Mapping – An Integrated Approach The Semantic Web: Research and Applications, 2004 – Springe

EHRIG, M. e SURE, Y. Ontology mapping - an integrated approach. In Christoph Bussler, John Davis, Dieter Fensel, and Rudi Studer, editors, Proceedings of the 1st' ESWS, volume 3053 of Lecture Notes in Computer Science, pages 76–91, Heraklion, Greece, 2004. Springer Verlag.

ELHADAD, D. G. M.; NETZER, Y. Automatic evaluation of search ontologies in the entertainment domain using text classification. In: . Applied Semantic Technologies: Using Semantics in Intelligent Information Processing. [S.l.]: Taylor and Francis, 2010.

EUZENAT, J. e SHVAIKO, P. (1998). "Ontology Matching". Springer.

EUZENAT, J. e SHVAIKO, P. A Survey of Schema-based Matching Approaches Journal on Data Semantics, 2005.

EUZENAT, J. e SHVAIKO, P. Ten Challenges for Ontology Matching In Proceedings of ODBASE, 2008.

FALBO, R. A.; MENEZES, C., e ROCHA, A. R. C., 1998, "Integração do Conhecimento Sobre Processos de Software em um Ambiente de Desenvolvimento"; In: proceedings of IX Conferência Internacional de Tecnologia de Software, Paraná, Brasil, Junho, 1998

FAROOQ, A.; ARSHAD, M. J.; SHAH, A. A Layered approach for Similarity Measurement between Ontologies Journal of American Science, 2010.

FELICISSIMO, Carolina; et all. Geração de ontologias subsidiada pela engenharia de requisitos. In: Workshop on Requirements Engineering. 6., 2003, Piracicaba, Brasil. Nov. 2003. p. 255-269.

FELLBAUM, C. *WordNet: an electronic lexical database*. The MIT Press, Cambridge (MA US), 1998.

FERNáNDEZ, M.; CANTADOR, I.; CASTELLS, P. A tool for collaborative ontology reuse and evaluation. 4th International Workshop on Evaluation of Ontologies for the Web, Edinburgh, UK, 2006. Disponível em: <http://km.aifb.uni-karlsruhe.de/ws- /eon2006/ eon2006fernandezetal.pdf>.

GAL, A; MODICA, G; JAMIL, HM (2003) Automatic ontology matching using application semantics.

GANGEMI, A. et al. A metaontology-based framework for ontology evaluation and selection. 4th International Workshop on Evaluation of Ontologies for the Web, Edinburgh, UK, 2006. Disponível em: <http://km.aifb.unikarlsruhe. de/ws/eon2006-/eon2006gangemietal.pdf>.

GNOME BRASIL. 2007. Disponível em: <http://br.gnome.org/>. Acesso em: 08/04/2007

GóMEZ-PéREZ, A. A. evaluation of taxonomic knowledge in ontologies and knowledge bases. Workshop On Knowledge Acquisition, Modeling And Management, Alberta, Alberta, Canada, 1999. Disponível em: <http://sern.ucalgary.ca/KSI/KAW/KAW99-/review papers/Gomez- Perez1/index.html>. Acesso em: 10/03/2009.

GóMEZ-PéREZ, A. Evaluation of ontologies. international. Journal of Intelligent Systems, Alberta, Canada, 2001. Acesso em: 10/03/2009.

GOOGLEMAPS. Belo Horizonte-Minas Gerais. GOOGLE, 2012. Disponível em: < http://maps.google.com.br/maps?hl=pt-PT&tab=wl>. Acesso em: 20 out. 2010.

GRéGIO, A. R. A.; FILHO, B. P. de C.; MONTES, R. S. A. Técnicas de visualização de dados aplicadas à segurança da informação. IX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionai, Campinas, São Paulo, 2009. Disponível em: <http://www.ppgia.pucpr.br/˜maziero/lib/exe/fetch.php/ceseg:2009-sbsegmc5.pdf>. Acesso em: 15/10/2010.

GRUBER, T R. What is an ontology? [S. l. : s. n.], 1996. Disponível em : <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html >
GRUBER, T. R. A translation approach to portable ontology specifications. 1993. Disponível em: <http://www-ksl.stanford.edu/KSL Abstracts/KSL-92-71.html>. Acesso em: 10/03/2009.

GRUNINGER, M.; FOX, M. S. Methodolgy for the design and evaluation of ontologies. Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knoweldge Sharing, Toronto, Canada, 1995.

GUARINO, N. Formal Ontology in Information Systems. Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998. Amsterdam, IOS Press, pp. 3-15.

GUARINO, N.; WELTY, C. Evaluating ontological decisions with ontoclean. 2006. Disponível em: <http://citeseer.ist.psu.edu/guarino02evaluating.html>. Acesso em: 01/04/2010.

GUIZZARDI, G. “Ontological Foundations for Structural Conceptual Models”. PhD Thesis (CUM LAUDE), University of Twente, The Netherlands. Published as the same name book in Telematica Institut Fundamental Research. Series No. 15, ISBN 90-75176-81-3 ISSN 1388-1795; No. 015; CTIT PhD-thesis, ISSN 1381-3617; No. 05-74. Holanda, 2005

GUIZZARDI, G. “The Role of Foundational Ontology for Conceptual Modeling and Domain Ontology Representation”. Proceedings of 7th DB&IS, Vilnius, IEEE Press 2006

HEALEY, C. G. Building a Perceptual Visualisation Architecture. Behaviour & Information Technology. 2000. 349-366 p.

HEIST, G., SCHREIBER, A. T. & WIELINGA, B. J. (1997), ‘Using Explicit Ontologies in KBS Development’, Internacional Journal of Human-Computer Studies 45, 183–292

HOLSAPPLE, C.; JOSHI, K. (2002a): Knowledge Management: A Threefold Framework. Information Society, 18 (1): 47-64.

INSELBERG, Alfred e DIMSDALE, Bernard. Parallel coordinates: a tool for visualizing multi-dimensional geometry. *Proceedings of the 1st conference on Visualization '90* (VIS '90), Arie Kaufman (Ed.). IEEE Computer Society Press, Los Alamitos, 1990, CA, USA.

JAVASCRIPT INFOVIS TOOLKIT:CREATE INTERACTIVE DATA VISUALIZATIONS FOR THE WEB. 2010. Disponível em: http://thejit.org/. Acesso em: 05/06/2010.

KALFOGLOU, Y.; SCHORLEMMER , M. Ontology Mapping: The State of the Art The Knowledge Engineering Review Journal, 2003.

KANG, J. e NAUGHTON, J. On schema matching with opaque column names and data values. In Proc. 22nd International Conference on Management of Data (SIGMOD), pages 205–216, San Diego (CA US), 2003.

KATIFORI, A. et al. Ontology visualization methods — a survey. ACM Comput. Surv., v. 50, n. 5, 2007.

KEIM, D. A. Visualization Techniques for mining Large Database: A Comparation. Transactions On Knouledge and Data Engineering, New York, USA v.8, n.6, 1966

KEIM, Daniel A., Krigel ,Hans-Peter, "VisDB: Database Exploration Using Multidimensional Visualization," IEEE Computer Graphics and Applications, p. 40-49, September/October, 1994

KINER, T. G.; CALONEGO, J.; BUK, C. V. Uso de realidade aumentada em ambientes virtuais de visualização de dados. 7th Symposium on Virtual Reality, Sao Paulo, 2004.

LAMPING, J.; RAO, R.; PIROLLI, P. A focus context technique based on hyperbolic geometry for visualizing large hierarchies. CA,USA, 1995. Disponível em: <http://www.acm.org/sigchi/chi95/Electronic/documnts/papers/jl bdy.htm>. Acesso em: 01/04/2010.

LOZANO-TELLO, A.; GóMES-PéREZ, A. Ontometric: A method to choose the appropriate ontology. Journal Of Database Management, 2004. Disponível em: <http://www.accessmylibrary.com/coms2/summary 0286-20574535 ITM>. Acesso em: 10/03/2009.

LUZZARDI, P. Critérios de Avaliação de Técnicas de Visualização de Informações Hierárquicas. Tese de doutorado — UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, Rio Grande do Sul, Brasil, 2003. Acesso em: 01 jul. 2010.

MAEDCHE, A.; STAAB, S. Measuring similarity between ontologies. Proc. Of the European Conference on Knowledge Acquisition and Management, Madri, Espanha, 2002. Disponível em: <http://www.unikoblenz.de/˜staab/Research/Publications/Publications.htm>. Acesso em: 22/02/2010.

MASCARDI V, LOCORO A e ROSSO, P. Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation- IEEE, 2010 - Engineers, Inc., 345 E. 47 th St. NY

MAYNARD, D. e ANANIADOU , S. Term extraction using a similarity-based approach. In Didier Bourigault, Christian Jacquemin, and Marie-Claude Lhomme, editors, *Recent advances in computational terminology*, pages 261–278. John Benjamins, Amsterdam (NL), 2001.

MILLER, G. WordNet: A lexical database for english. Communications of the ACM, 38(11):39–41, 1995.

NASCIMENTO, H. A. D. do; FERREIRA, C. B. R. Visualização de informação –uma abordagem prática. XXV Congresso da Sociedade Brasileira de Computação, Rio Grande do Sul, Brasil, 2005. Disponível em: <http://www.unikoblenz.de/˜staab-/Research/Publications/Publications.htm>. Acesso em:22/02/2010.

NOY , N. F. e MUSEN, M. PROMPT: Algorithm and tool for automated ontology merging and alignment In Proceedings of AAAI, 2000.

NOY, F. N.; GUINNESS, D. L. Ontology development 101: a guide to c r e a t e y our f i r s t ont o l o g y . Di sponí v e l em: <ht tp: / / k s l . s t anf o rd. edu/people /dlm/papers/ontology-tutorial-noy-mcguinness.doc>

NOY, N. F. Semantic integration: A survey of ontology-based approaches. ACM SIGMOD Record, 33(4):65–70, 2004.

NOY, N. F.; MUSEN, M. A. Anchor-PROMPT: Using Non-Local Context for Semantic Matching In Proceedings of the Workshop on Ontologies and Information Sharing at IJCAI, 2001

NOY, N. F.; MUSEN, M. Ontology versioning in an ontology management framework. IEEE Intelligent Systems, 19(4):6–13, 2004.

NOY, N. F.; MUSEN, M. PROMPT: Algorithm and tool for automated ontology merging and alignment. In Proc. 17th National Conference of Artificial Intelligence (AAAI), pages 450–455, Austin (TX US), 2000. Noy, N. F.; Musen, M.. Ontology evolution: Not the same as schema evolution. Knowledge and Information Systems, 6(4):428–440, 2004.

NOY, N. F.; MUSEN, M. SMART: Automated support for ontology merging and alignment. In Proc. 12th Workshop on Knowledge Acquisition, Modeling and Management (KAW), Banff (CA), 1999.

NOY, N. F.; MUSEN, M. The PROMPT suite: interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies, 59(6):983–1024, 2003.

ONTO Viz. Protégé. 2010. Disponível em: < http://protegewiki.stanford.edu/wiki/OntoViz>

ONTOSPHERE3D. E-Life Research Group. 2010. Disponível em: <http://ontosphere3d.sourceforge.net/>. Acesso em 14 out. 2010.

PALOPOLI, L., PONTIERI, L., TERRACINA, G. e URSINO, D. Intensional and extensional integration and abstraction of heterogeneous databases. Data and Knowledge Engineering, 35(3):201–237, 2000.

PILLAT, Raquel Mainardi. Coordenação dinâmica de visualizações de dados multidimensionais. Dissertação de Mestrado — Universidade Federal do Rio Grande do Sul, Rio Grande do Sul, Brasil, 2006. Acesso em: 05/04/2010.

POHLHEIM, H..“Visualization of evolutionary algorithms-set of standard techniques and multidimensional visualization. 1999.

PORZEL, R.; MALAKA, R. A task-based approach for ontology evaluation. Workshop on Ontology Learning and Population in 16th European Conference on Artificial Intelligence,

Valencia, Espanha, 2004. Disponível em: <http://olp.dfki.de/ecai04/finalporzel.pdf>. Acesso em: 22/05/2010.

RAHM, E e BERNSTEIN, P. A survey of approaches to automatic schema matching. The VLDB Journal, 10(4):334–350, 2001.

RAUNICH, S. e RAHM, E. ATOM: Automatic Target-driven Ontology Merging In Proceedings of ICDE, 2011

ROBBI, C. Sistema para visualização de informações Cartograficas para planejamento urbano. Tese de doutorado — INSTITUTO NACIONAL DE PESQUISAS ESPACIAI-INPE, São José dos Campo, 2000. Disponível em: <http://mtc-m05.sid.inpe.br-/col/sid.inpe.br/deise/2000/11.06.09.31/doc/TeseClaudia.pdf>. Acesso em: 01 jul. 2010.

ROBERTSON, G. G.; MACKINLAY, J. D.; CARD, S. K. Cone trees: animated 3dvisualizations of hierarchical information. Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology, Louisiana, United States, 1991. Disponível em: <http://portal.acm.org/citation.cfm?id=108883&retn=1>. Acesso em: 01/02/2010.

RODDICK, J.F.. A survey of schema versioning issues for database systems. Information and Software Technology, 37(7), 383-393, 1996.

SABOU, M. et al. Ontology selection: Ontology evaluation on the real semantic web. 4th International Workshop on Evaluation of Ontologies for theWeb at the 15th International World Wide Web, Edinburgh, UK, 2006. Disponível em: <http://km.aifb.uni-karlsruhe.de/ws/eon2006/eon2006sabouetal.pdf>.

SANTANNA, G. Ontologias e Web Service. Universidade Federal da Bahia, 2008. Disponível em: http://wiki.dcc.ufba.br/TISA/OntologiasWebServices
SAP Community Network. 2010. Disponível em: <http://scn.sap.com/community/pi-and-soa-middleware>. Acesso em: 05 set. 2010.

SASSO, C. M. D.; CHUBACHI, O. M.; LUZZARDI, P. Introdução à visualização de informações. Revista de Informática Teórica e Aplicada, Rio Grande do Sul, Brasil, p.143–158, 2001.

SETZER, V. W. Dado, informação, conhecimento e competência. DATAGRAMAZERO-Revista de Ciência da Informação, n. 0, 1999.

SHNEIDERMAN, B. Treemaps for space-constrained visualization of hierarchies. [S.L.], Dec. 26th, 1998. Disponível em: < http://www.cs.umd.edu/hcil/treemap-history/>. Acesso em 14 out. 2010.

SHNEIDERMAN, B. Treemaps for space-constrained visualization of hierarchies. Maryland,USA, 1998. Disponível em: <http://www.cs.umd.edu/hcil/treemap-history/>. Acesso em: 01/04/2010.

SHVAIKO, P., EUZENAT, J.: Ontology matching: state of the art and future challenges IEEE Transactions on Knowledge and Data Engineering, 2012.
SIGCHI. The Association For Computing Machinery. [S.L.], 2010. Disponível em: http://www.sigchi.org

SMITH B. Ontology. In L. Floridi, editor, Blackwell companion to philosophy, information and computers, Oxford, 2003.

SOWA, J. F. (2006), Ontology. Disponível em: http://www.jfsowa.com/ontology/index.htm. Acessado: 13/10/2010.

SPRINGERIMAGES. Tela TGVizTab . 2010http://www.springerimages.com/)
STANFORD.... Árvores identadas. http://vis.stanford.edu/protovis/ex/indent.html Acesso em 18/09/2010

STEFANER, M. A star for your christimas tree. **Blog Well Formed Data**. [s.l.], Dec. 17th 2007. Disponível em: http://well-formed-data.net. Acesso em 14 out. 2010.

STOLTE, C.; TANG, D. e HANRAHAN, P. Polaris: a system for query, analysis, and visualization of multidimensional databases. Commun. ACM, v. 51, n. 11, p. 75–84, 2008. Acesso em: 15/04/2010.

TARTIR, S. et al. Ontoqa: Metric-based ontology quality analysis. Proc. Of Workshop on Knowledge Acquisition from Distributed,Autonomous, Semantically Heterogeneous Data and Knowledge Sources., 2006. Disponível em: <http://km.aifb.uni-karlsruhe-.de/ws/eon2006/ eon2006fernandezetal.pdf>.

THE GENE ONTOLOGY. http://www.geneontology.org/. Acesso em 12 jul. 2010.
THE XML C PARSER AND TOOLKIT OF GNOME. Disponível em: <http://www.xmlsoft.org/> Acesso em 08 abr. 2010.

TOMOGRAFIA Computadorizada. **Wikipédia**. [s. l], 2010. Disponível em: <

http://pt.wikipedia.org/wiki/Tomografia_computadorizada>. Acesso em 14 out. 2010.

TOWNNSEND, M. Discrete Mathematic: Applied combinatorics and graph theory. [S.l.]: Benjamin/Cummings, 1987.

VAISHNAVI, V.; KUECHLER,W. Design research in information systems. IS WorldNet, 2004. Disponível em: <http://home.aisnet.org/displaycommon.cfm?an=1&subarticlenbr=279>. Acesso em: 01/10/2011.

VALIATI, E. R. A. Avaliação de usabilidade de técnicas de visualização de informações multidimensionais Universidade Federal do Rio Grande do Sul. Instituto de Informática. Programa de Pós-Graduação em Computação. 2008

VALTCHEV, P. (1999). *Construction automatique de taxonomies pour laide la reprsentation de connaissances par objets*. Ph.D. Dissertation, Universite Grenoble.

VALTCHEV, P., e EUZENAT, J. Eds., Dissimilarity measure for collections of objects and values, ser. Lecture Notes in *Computer Science*. London, UK: Springer, 1997, vol. 1280.

VIM The Editor. Disponível em: <http://www.vim.org/>. Acesso em: 22 mar.2010

W3C a. 2003. Disponível em: <http://www.w3.org/ >. Acesso em: 03/04/2009.

W3C a. 2004. Disponível em: <http://www.w3.org/TR/owl-features/>. Acesso em: 03/04/2009.

W3C. 2005

W3C. 2011. www.w3.org . Acesso em 13 fev. 2011.

WANG,Y., LIU, W. e BELL , D.: A Structure-based Similarity Spreading Approach for Ontology Matching In Proceedings of SUM, 2010.

WETHERELL, C.; SHANNON, A. Tidy drawings of trees. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, v. 2, n. 5, 1979. Acesso em: 15/04/2010.

YELLEN, J.; GROSS, J. L. Handbook of graph theory: Graph theory and its application. [S.l.]: CRC Press, 2004.

YU, J.; THOM, J. A.; TAM, A. Ontology evaluation using wikipedia categories for browsing. Proceedings of the sixteenth ACM conference on Conference on information and knowledge

management, New York, NY, USAl, 2007. Disponível em: <http://doi.acm.org/10.1145/1321440.1321474>. Acesso em: 22/02/2010.

ZIVIANI, N. Projeto de Algoritmos com implementação em C++ e Java. THOMSON Learning. 2004.

Anexo 1 – Resultados: Tabelas e Algoritmos

Fase Alinhamento String

Alinhamento String Primeira Fase			
FAO	Equivalência	CELL	Condidata
Spore	Ótimo	fungal spore	Sim
asexual spore	Ótimo	fungal asexual spore	Sim
Conidium	Ótimo	conidium	Sim
Macroconidium	Ótimo	macroconidium	Sim
Arthroconidium	Ótimo	arthroconidium	Sim
multinucleate arthroconidium	Ótimo	multinucleate arthroconidium	Sim
uninucleate arthroconidium	Ótimo	uninucleate arthroconidium	Sim
Blastoconidium	Ótimo	blastoconidium	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	Sim
uninucleate blastconidium	Ótimo	uninucleate blastconidium	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	Sim
uninucleate macroconidium	Ótimo	uninucleate macroconidium	Sim
multinucleate conidium	Ótimo	multinucleate conidium	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	Sim
uninucleate conidium	Ótimo	uninucleate conidium	Sim
conidium of conidiophore head	Ótimo	conidium of conidiophore head	Sim
Microconidium	Ótimo	microconidium	Sim
Oospore	Ótimo	oospore	Sim
sexual spore	Ótimo	sexual spore	Sim
Ascospore	Ótimo	ascospore	Sim
Basidiospore	Ótimo	basidiospore	Sim
Zygospore	Ótimo	zygospore	Sim
Hypa	Ótimo	hyphal cell	Sim
multinucleate arthroconidium	Ótimo	uninucleate arthroconidium	Não
uninucleate arthroconidium	Ótimo	multinucleate arthroconidium	Não
multinucleate blastoconidium	Ótimo	uninucleate blastconidium	Não
uninucleate blastconidium	Ótimo	multinucleate blastoconidium	Não
multinucleate macroconidium	Ótimo	uninucleate macroconidium	Não
multinucleate blastoconidium	Ótimo	multinucleate macroconidium	Não
uninucleate macroconidium	Ótimo	multinucleate macroconidium	Não
multinucleate arthroconidium	Ótimo	multinucleate arthroconidium	Não
uninucleate conidium	Ótimo	uninucleate conidium	Sim

conidium of conidiophore head	Ótimo	conidium of conidiophore head	Sim

Alinhamento String Segunda Fase				
FAO	Equivalência	CELL	Pontuação	Condidata
Spore	Médio	fungal spore	3	Sim
asexual spore	Bom	fungal asexual spore	4	Sim
Conidium	Ótimo	conidium	5	Sim
Macroconidium	Ótimo	macroconidium	5	Sim
Arthroconidium	Ótimo	arthroconidium	5	Sim
multinucleate arthroconidium	Ótimo	multinucleate arthroconidium	5	Sim
uninucleate arthroconidium	Ótimo	uninucleate arthroconidium	5	Sim
Blastoconidium	Ótimo	blastoconidium	5	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	5	Sim
uninucleate blastconidium	Ótimo	uninucleate blastconidium	5	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	5	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	5	Sim
uninucleate macroconidium	Ótimo	uninucleate macroconidium	5	Sim
multinucleate conidium	Ótimo	multinucleate conidium	5	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	5	Sim
uninucleate conidium	Ótimo	uninucleate conidium	5	Sim
conidium of conidiophore head	Ótimo	conidium of conidiophore head	5	Sim
Microconidium	Ótimo	microconidium	5	Sim
Oospore	Ótimo	oospore	5	Sim
sexual spore	Ótimo	sexual spore	5	Sim
Ascospore	Ótimo	ascospore	5	Sim
Basidiospore	Ótimo	basidiospore	5	Sim
Zygospore	Ótimo	zygospore	5	Sim
Hypa	Médio	hyphal cell	3	Sim
multinucleate arthroconidium	Baixo	uninucleate arthroconidium	2	Não
uninucleate arthroconidium	Baixo	multinucleate arthroconidium	2	Não
multinucleate blastoconidium	Baixo	uninucleate blastconidium	2	Não
uninucleate blastconidium	Baixo	multinucleate blastoconidium	2	Não
multinucleate macroconidium	Baixo	uninucleate macroconidium	2	Não
multinucleate blastoconidium	Baixo	multinucleate macroconidium	2	Não
uninucleate macroconidium	Baixo	multinucleate macroconidium	2	Não
multinucleate arthroconidium	Baixo	multinucleate arthroconidium	2	Não
uninucleate conidium	Ótimo	uninucleate conidium	5	Sim

conidium of conidiophore head	Ótimo	conidium of conidiophore head	5	Sim

Alinhamento String Terceira Fase				
FAO	Equivalência	CELL	Pontuação	Condidata
Spore	Baixo	fungal spore	2	Não
asexual spore	Médio	fungal asexual spore	3	Sim
Conidium	Ótimo	conidium	5	Sim
Macroconidium	Ótimo	macroconidium	5	Sim
Arthroconidium	Ótimo	arthroconidium	5	Sim
multinucleate arthroconidium	Ótimo	multinucleate arthroconidium	5	Sim
uninucleate arthroconidium	Ótimo	uninucleate arthroconidium	5	Sim
Blastoconidium	Ótimo	blastoconidium	5	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	5	Sim
uninucleate blastconidium	Ótimo	uninucleate blastconidium	5	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	5	Sim
multinucleate blastoconidium	Ótimo	multinucleate blastoconidium	5	Sim
uninucleate macroconidium	Ótimo	uninucleate macroconidium	5	Sim
multinucleate conidium	Ótimo	multinucleate conidium	5	Sim
multinucleate macroconidium	Ótimo	multinucleate macroconidium	5	Sim
uninucleate conidium	Ótimo	uninucleate conidium	5	Sim
conidium of conidiophore head	Ótimo	conidium of conidiophore head	5	Sim
Microconidium	Ótimo	microconidium	5	Sim
Oospore	Ótimo	oospore	5	Sim
sexual spore	Ótimo	sexual spore	5	Sim
Ascospore	Ótimo	ascospore	5	Sim
Basidiospore	Ótimo	basidiospore	5	Sim
Zygospore	Ótimo	zygospore	5	Sim
Hypa	Baixo	hyphal cell	2	Não
multinucleate arthroconidium	Excluir	uninucleate arthroconidium	1	Não
uninucleate arthroconidium	Excluir	multinucleate arthroconidium	1	Não
multinucleate blastoconidium	Excluir	uninucleate blastconidium	1	Não
uninucleate blastconidium	Excluir	multinucleate blastoconidium	1	Não
multinucleate macroconidium	Excluir	uninucleate macroconidium	1	Não
multinucleate blastoconidium	Excluir	multinucleate macroconidium	1	Não
uninucleate macroconidium	Excluir	multinucleate macroconidium	1	Não
multinucleate arthroconidium	Excluir	multinucleate arthroconidium	1	Não
uninucleate conidium	Ótimo	uninucleate conidium	5	Sim

conidium of conidiophore head	Ótimo	conidium of conidiophore head	5	Sim

Fase Alinhamento estrutura

Estrutura fase 1					
Cell		Equivalência	FAO		
Classe	SubClasse		Classe	SubClasse	Candidata
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Boa	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Boa	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim
macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Boa	macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Boa	multinucleate conidium	[multinucleate macroconidium]	Sim
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Boa	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim
conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Boa	conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Sim
fungal asexual spore	[chlamydospore; conidium; oospore]	Média	asexual spore	[aeciospore; aplanospore; conidium; oospore]	Sim
fungal spore	[fungal asexual spore; sexual spore]	Media	spore	[asexual spore; sexual spore]	Sim
sexual spore	[ascospore; basidiospore; zygospore]	Boa	sexual spore	[ascospore; basidiospore; zygospore]	Sim
arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Boa	arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Sim

blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Boa	blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Sim
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Boa	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Boa	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Boa	multinucleate conidium	[multinucleate macroconidium]	Sim
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Boa	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim
fungal cell	[H minus; H plus; fungal spore; heterokaryon; hyphal cell; vegetative cell (sensu Fungi)]	Boa	unicelluar structre	[mating cell; spore; vegetative cell]	Sim
Hyphal Cell	[]	Ruim	hypha	[Hulle cell; aseptate hypha; conidiophore; hypha in mycelium; hyphal tip; oidium; septate hypha]	Não

Estrutura fase 2 - Comparação nomes					
Cell		Equivalência	FAQ		Candidata
Classe	SubClasse		Classe	SubClasse	
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim

macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Ótima	macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate macroconidium]	Sim
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Ótima	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim
conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Ótima	conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Sim
fungal asexual spore	[chlamydospore; conidium; oospore]	Média	asexual spore	[aeciospore; aplanospore; conidium; oospore]	Sim
fungal spore	[fungal asexual spore; sexual spore]	Media	spore	[asexual spore; sexual spore]	Sim
sexual spore	[ascospore; basidiospore; zygospore]	Ótima	sexual spore	[ascospore; basidiospore; zygospore]	Sim
arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Ótima	arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Sim
blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Ótima	blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Sim
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate macroconidium]	Sim
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Ótima	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim

fungal cell	[H minus; H plus; fungal spore; heterokaryon; hyphal cell; vegetative cell (sensu Fungi)]	Excluir	unicelluar structre	[mating cell; spore; vegetative cell]	Não
Hyphal Cell	[]	Média	hypha	[Hulle cell; aseptate hypha; conidiophore; hypha in mycelium; hyphal tip; oidium; septate hypha]	Sim

Estrutura fase 3 - Nomes subclasses					
Cell		Equivalência	FAQ		Candidata
Classe	SubClasse		Classe	SubClasse	
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim
macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Ótima	macroconidium	[arthroconidium; blastoconidium; multinucleate macroconidium; uninucleate macroconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate macroconidium]	Sim

uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Ótima	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim
conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Ótima	conidium	[macroconidium; multinucleate conidium; uninucleate conidium]	Sim
fungal asexual spore	[chlamydospore; conidium; oospore]	Média	asexual spore	[aeciospore; aplanospore; conidium; oospore]	Sim
fungal spore	[fungal asexual spore; sexual spore]	Boa	spore	[asexual spore; sexual spore]	Sim
sexual spore	[ascospore; basidiospore; zygospore]	Ótima	sexual spore	[ascospore; basidiospore; zygospore]	Sim
arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Ótima	arthroconidium	[multinucleate arthroconidium; uninucleate arthroconidium]	Sim
blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Ótima	blastoconidium	[multinucleate blastoconidium; uninucleate blastconidium]	Sim
multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Ótima	multinucleate macroconidium	[multinucleate arthroconidium; multinucleate blastoconidium]	Sim
uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Ótima	uninucleate macroconidium	[uninucleate arthroconidium; uninucleate blastoconidium]	Sim
multinucleate conidium	[multinucleate macroconidium]	Ótima	multinucleate conidium	[multinucleate	Sim

				macroconidium]	
uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Ótima	uninucleate conidium	[conidium of conidiophore head; microconidium; uninucleate macroconidium]	Sim
fungal cell	[H minus; H plus; fungal spore; heterokaryon; hyphal cell; vegetative cell (sensu Fungi)]	Média	unicelluar structre	[mating cell; spore; vegetative cell]	Sim
Hyphal Cell	[]	Excluir	hypha	[Hulle cell; aseptate hypha; conidiophore; hypha in mycelium; hyphal tip; oidium; septate hypha]	Não

Fase Avaliação Alinhamentos

Completude Fase Baseado em Nome

FAO	Descrição	Propriedades	Equivalência
spore	5	3	Bom
asexual spore	4	4	Bom
conidium	5	3	Bom
macroconidium	5	3	Bom
arthroconidium	5	4	Bom
multinucleate arthroconidium	5	4	Bom
uninucleate arthroconidium	4	3	Bom
blastoconidium	4	4	Bom
multinucleate blastoconidium	4	4	Bom
uninucleate blastconidium	4	4	Bom
multinucleate macroconidium	4	4	Bom
. . .	. . .	. . .	. . .
multinucleate blastoconidium	4	5	Ótimo
uninucleate macroconidium	5	5	Ótimo
multinucleate conidium	4	5	Ótimo
multinucleate macroconidium	4	5	Ótimo
uninucleate conidium	5	5	Ótimo
conidium of conidiophore head	4	5	Ótimo
microconidium	4	5	Ótimo
oospore	4	5	Ótimo
sexual spore	4	5	Ótimo
ascospore	4	5	Ótimo
basidiospore	3	4	Ótimo
zygospore	3	4	Ótimo
Hypa	0	4	Ótimo
multinucleate arthroconidium	4	5	Ótimo
uninucleate arthroconidium	5	5	Ótimo
multinucleate blastoconidium	4	5	Ótimo
uninucleate blastconidium	4	5	Ótimo
multinucleate macroconidium	5	5	Ótimo
multinucleate blastoconidium	5	4	Ótimo
uninucleate macroconidium	4	4	Ótimo
multinucleate arthroconidium	5	4	Ótimo
uninucleate conidium	4	4	Ótimo
conidium of conidiophore head	5	5	Ótimo

Completude Fase Baseado em estrutura

FAO	Descrição	Propriedades	Avaliação
basidiospore	5	5	Ótimo
zygospore	5	5	Ótimo
Hypa	3	0	Baixo
multinucleate arthroconidium	4	3	Médio
uninucleate arthroconidium	5	3	Médio
multinucleate blastoconidium	4	3	Médio
uninucleate blastconidium	5	4	Bom
multinucleate macroconidium	5	3	Médio
multinucleate blastoconidium	4	4	Bom
uninucleate macroconidium	4	4	Bom
multinucleate arthroconidium	4	4	Bom
. . .	. . .	. . .	. . .

Fase Final

Fase 1		Fase 2		Resultados	
Pontuação	Candidatas	Pontuação	Candidata	Pontuação	Candidata
4	Sim	4	Sim	4,0	Sim
4	Sim	5	Sim	4,5	Sim
5	Sim	4	Sim	4,5	Sim
4	Sim	4	Sim	4,0	Sim
4	Sim	5	Sim	4,5	Sim
5	Sim	4	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
5	Sim	5	Sim	5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
3	Sim	4	Sim	3,5	Sim
3	Sim	2	Sim	2,5	Sim
5	Sim	2	Sim	3,5	Sim

5	Sim	5	Sim	5	Sim
4	Sim	4	Sim	4	Sim
3	Sim	3	Sim	3	Sim
5	Sim	4	Sim	4,5	Sim
4	Sim	3	Sim	3,5	Sim
5	Sim	4	Sim	4,5	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
3	Sim	4	Sim	3,5	Sim
3	Sim	4	Sim	3,5	Sim
3	Sim	4	Sim	3,5	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
4	Sim	4	Sim	4	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
5	Sim	4	Sim	4,5	Sim
54	Sim	4	Sim	29	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
4	Sim	5	Sim	4,5	Sim
3	Sim	3	Sim	3	Sim
5	Sim	5	Sim	5	Sim
3	Sim	3	Sim	3	Sim
5	Sim	5	Sim	5	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
5	Sim	5	Sim	5	Sim
5	Sim	5	Sim	5	Sim
4	Sim	4	Sim	4	Sim
4	Sim	4	Sim	4	Sim
5	Sim	5	Sim	5	Sim
4	Sim	5	Sim	4,5	Sim
5	Sim	5	Sim	5	Sim

4	Sim	4	Sim	4	Sim

Compilação Final

	CELL	FAO
Corretude Alinhamento	3,9	4,5
Corretude Estrutura	4,2	5
Corretude Descrição	4,3	4,6
Avaliação Final	4,133333	4,7

ALGORITMOS

Parser

Demonstrar-se-á , em forma de algoritmo (aqui descrito em alto nível), os resultados do desenvolvimento do *parser* descrito na seção 5.1.1 do capítulo de metodologia. O *parser* foi implementado com sua divisão em macrofunções distintas que são chamadas dentro de um programa principal e funções suporte que são utilizadas dentro das funções principais. Na função principal está contida a criação dos dicionários, a equiparação destes e a chamada da construção da biblioteca de geração de semiontologias e visualização. As funções são descritas a seguir.

O sistema principal é chamado a partir da interface web todas as vezes que uma ontologia é carregada. Assim que a ontologia entra no sistema ela se torna uma estrutura "json" passível de ser visualizada. Para essa transformação os algoritmos descritos abaixo são utilizados.

- **Principal:** é a função principal do programa. Aqui, são concentradas todas as funções que auxiliam o processo do sistema. As funções são descritas a seguir de acordo com sua ordem de chamada dentro da função principal.

```
principal(onto:arquivo):
        cria listaClasses
        cria owl
        cria dic1, dic2; dic3;dic4
        substituir = [[#;?CHARP];[ :; ?DOT]]
        ler onto linha a linha
                owl += limpaText(owl;substituir)
        listaClasses = Libxml2.crialistasubclasses(owl)
        para cada classe contida em listaClasses
                procura pela palavra("rdfssubClassOf") no texto de classe
                    se nao existir
                        dic1[buscaNomeClasse(classe)] é criado e recebe uma lista
                vazia ([])
```

para cada nome classe contida nomes do dic1
dic1 [classe] é aumentado do resultado de buscaSubClasse(owl, classe)
excluir calsse de listaClasses

para cada classe contida em listaClasses:
procura pela palavra("rdfssubClassOf") no texto de classe
se existir entao
dic2[buscaNomeClasse(classe)] é criado e recebe uma lista vazia ([])

para cada nome classe contida nomes do dic2
se resultado buscaSuperClasse(owl, classe) esta em dici1
dic2 [classe] é aumentado do resultado de buscaSuperClasse(owl, classe)
excluir calsse de listaClasses
se não
exclui classe de dic2

para cada classe contida em listaClasses:
dic3[buscaNomeClasse(classe)] é criado e recebe uma lista vazia ([])

para cada nome classe contida nomes do dic3
dic3 [classe] é aumentado do resultado de buscaSuperClasse(owl, classe)

dic4 recebe o resultado de mesclaDicionario(dic1;dic2;dic3)
retorna geraJson(dic4)

- **LimpaTexto:** função com objetivo de tornar o texto legível para a biblioteca utilizada.

limpaTexto(owl:*string*; substituir:lista *string*)
para cada componente comp de substituir
owl = owl.substituir(comp[0], com[1])

Função chamada dentro de um laço com a lista de caracteres que quebram a execução da biblioteca Libxml2. Para cada iteração da lista o caractere que quebra a execução, que está contido dentro do parâmetro “substituir”, é substituído por um caractere que é aceito pela biblioteca. O parâmetro “substituir” é uma lista de itens compostos pelo caractere que param a execução e pelos caracteres que os substituirão no texto, cada item da lista “substituir” tem o seguinte formato: [x:y](aqui x será substituído por y). Já o parâmetro “owl” é a sentença que será alterada pela função de acordo com os parâmetros dados pela lista “substituir”. Essa sentença é somente um pedaço da ontologia sendo alinhada. Por exemplo:

owl = “<owl:Class rdf:about="http://purl.obolibrary.org/obo/IAO_0000027#">”

substituir = [[“:”;”?DOT”]; [“#”:”?CHARP”]]

Chamada da função ficaria da seguinte forma:

limpaTexto(owl; substituir)

para cada componente comp de substituir
owl = owl.substituir(comp[0], com[1])

No final teríamos como resultado para duas iterações:

owl = <owl?DOT Class

rdf?DOTabout="http?DOT//purl.obolibrary.org/obo/IAO_0000027?CHARP">

- **buscaNomeClasse:** Função suporte que retorna nome das sub e superclasses do sistema.

buscaNomeClasse(owl:node; classe:node; flag:int)
Se flag for 1
Retornar superclasse
Se flag for 2
Retornar subclasse

Função que retorna nome da subclasse ou superclasse de acordo com a ordem dos parâmetros passados.

- Se buscarmos superclasses, o primeiro parâmetro “owl” deve vir da iteração em cima de todas as classes das ontologias; o segundo parâmetro “classe” deve ser a classe pesquisada e o parâmetro “flag” deve ser setado em 1;
- Se buscarmos subclasses, o primeiro parâmetro “owl” deve ser a classe pesquisada, o segundo parâmetro “classe” deve vir da iteração em cima de todas as classes das ontologias e o parâmetro “flag” deve ser setado em 2.

- **BuscaSubClasse:** função que retorna lista de subclasses para uma classe específica

buscaSubClasse(owl:*string*; classe:node)
cria listaNome
cria listaObjeto
cria listaGeral
Enquanto contador <> tamanho(owl)
Nome = buscaNomeClasse(owl[contador]; classe;2)
Se Nome <> vazio
listaNome.adiciona(nome)
listaObejto.adiona(owl[contador])

listaGeral recebe [listaNome;listaObjeto]
Retorna listaGeral

Função que gera uma lista de nomes e objetos subclasses para cada classe consultada em uma iteração. Ela utiliza a função “buscaNomeClasse” como suporte de criação da lista de objetos e nomes.

- **BuscaSuperClasse:** Função que retorna lista de superclasses para uma classe específica

buscaSuperClasse(owl:nodeList; classe:node)
cria listaNome
cria listaObjeto
cria listaGeral
Enquanto contador <> tamanho(owl)
Nome = buscaNomeClasse(owl[contador]; classe;1)
Se Nome <> vazio
listaNome.adiciona(nome)
listaObejto.adiona(owl[contador])
listaGeral recebe [listaNome;listaObjeto]

Função que gera uma lista de nomes e objetos superclasses para cada classe consultada em uma iteração. Ela utiliza a função “buscaNomeClasse” como suporte de criação da lista de objetos e nomes.

- **mesclaDicionario:** faz as comparações necessárias para se ter um estrutura de dicionários única de acordo com o descrito na metodologia seção 5.2

mesclaDicionario(dic1:dicionario;dic2:dicionario;dic3:dicionario)
retorna dic1Xdic2Xdic3

- **geraJson:** resultado da iterações são transformadas no dicionário “json” para visualização

geraJson(dic:dicionario;owl:nodelist;onto:*string*)
cria json vazio
para cada classe i dentdo dos nomes de dicionario
json acrescenta '{'
Nome = buscaNomeClasse
NameSpace = buscaNSClasse
Propriedades = buscaProperties
Descricao = buscaDescricaoClasse
Recursive para subclasses
‘}’

- **retorna json**

- **buscaDescricaoClasse:** Função suporte que retorna nome das sub e superclasses do sistema.

buscaDescricaoClasse(owl:nodelist; classe:*string*)
Retornar descricao

Função que retorna descrição da subclasse ou superclasse ou propriedade de acordo com a ordem dos parâmetros passados.

- **buscaNSClasse:** Função suporte que retorna namespace das sub e superclasses do sistema.

buscaNSClasse(owl:nodelist; classe:*string*)
Retornar namespace

Função que retorna namespace da subclasse ou superclasse ou propriedade de acordo com a ordem dos parâmetros passados.

- **buscaNomeProp:** Função suporte que retorna nome das propriedades de uma classe.

buscaNomeProp(classe:node)

Enquanto no esta contido na tag onPropertie
nomePropre.adiciona(no.label)
Retornar nomePropre

Aproveita a estrutura de nos geradas para fazer iteração nos nós filhos da classe procurando por nomes de propriedades.

- **buscaProp:** função suporte que retorna descrição das propriedades de uma classe.

buscaProp(classe:node;owl:nodeList)
Enquanto no esta contido na tag onPropertie
nomePropre.adiciona(buscaDescricaoClasse(owl;no.label))
Retornar nomePropre

Aproveita a estrutura de nós gerada para fazer iteração nos nós filhos da classe procurando por descrições de propriedades.

- **BuscaProperties:** Função que retorna lista de propriedades e descrições para uma classe específica

buscaProperties(classe:node;owl:nodelist)
cria listaProp
listaProp.adiona(buscaNomeProp(classe);buscaProp(classe;owl))
Retorna listaProp

Função que gera uma lista de nomes e descrições (caso existam) das propriedades para cada classe consultada em uma iteração. Ela utiliza a função “buscaNomeProp” e “buscaProp” como suporte de criação da lista de descrições e nomes.

Baseado em nome

Aqui apresentamos em alto nível as funções que constituem o algoritmo de *matching* baseado em nome.

Primeira fase

A primeira fase baseia-se na comparação direta e simples, somente fazendo comparação da existência de alguma semelhança entres dois nomes de classes candidatos ao alinhamento.

```
comparaNomeLivre(dic1:dicionario;dic2:dicionario)
    lista =[]
    para cada nome d1 na lista de chaves de dic1
        para cada nome d2 na lista de chaves de dic2
            se lista de caracteres ou aproximações de d2 existir em d1
                lista.adiciona([d1;d2];otimo)
    retorna lista
```

Um laço direto é feito nas classes das duas estruturas e a comparação através de uma função da biblioteca de expressão regular é feita. Para cada classe fixada do dicionário 1 todas as classes do dicionário 2 são analisadas. O resultado é retornado em uma lista de tuplas.

Segunda fase

A segunda fase baseia-se na comparação do resultado da primeira fase mais a resposta que o usuário informa ao sistema. Faz comparação da existência de semelhança real entres dois nomes de classes candidatos ao alinhamento aceitando apenas algumas diferenças entre letras e espaços nas palavras.

```
comparaNomeMedio(lista:list;retorno:list)
    lista2 =[ ]
    para cada dupla existente em lista
        verifica se existe dupla em retorno
            verifica retorno na segunda posição para sim ou nao
```

```
                aplica comparação substituições junto ao retorno (sim/não) para os
                pares lista
                        sem substituições
                                niveldado = ótimo
                        poucas substituições
                                niveldado = bom
                        medias substituições
                                niveldado = médio
                        varias substituições
                                niveldado = baixo
                        todas substituições
                                niveldado = excluir
                lista2.adiciona(dupla,níveldado)
        se não existe em retorno
           aplica comparação substituições para os pares lista
                                        sem substituições
                                                niveldado = ótimo
                                        poucas substituições
                                                niveldado = bom
                                        medias substituições
                                                niveldado = médio
                                        varias substituições
                                                niveldado = baixo
                                        todas substituições
                                                niveldado = excluir
                                        lista2.adiciona(dupla,níveldado)
        retorna lista2
```

Um laço direto é feito na lista de retorno da fase anterior junto a lista de retorno externo do usuário. As estruturas são comparadas através de uma função da biblioteca de expressão regular. Para cada classe fixada do dicionário 1 todas as classes do dicionário 2 são analisadas. O resultado é retornado em uma lista de tuplas depois de se ter feito a comparação de nomes em relação ao retorno externo.

Terceira fase

A terceira fase baseia-se na comparação do resultado da segunda fase mais a resposta que o usuário informa ao sistema. Faz comparação da existência de semelhança real entres dois nomes de classes candidatos ao alinhamento não aceitando diferenças de nenhum estilo (entre letras e espaços nas palavras).

```
comparaNomeSsevero(lsita:list;retorno:list)
        lista3 =[ ]
        para cada dupla existente em lista
        verifica se existe dupla em retorno
```

```
verifica retorno na segunda posição para sim ou não
        sim
                aplica comparação exata para os pares lista
                        iguais
                                niveldado = ótimo
                        poucas diferencas
                                niveldado = bom
                        medias diferencas
                                niveldado = médio
                        varias diferencas
                                niveldado = baixo
                        todas diferencas
                                niveldado = excluir
                        lista3.adiciona(dupla,níveldado)
                        não
                                continua
    se não existe em retorno
        aplica comparação exata para os pares lista
                iguais
                        niveldado = ótimo
                poucas diferencas
                        niveldado = bom
                medias diferencas
                        niveldado = médio
                varias diferencas
                        niveldado = baixo
                todas diferencas
                        niveldado = excluir
        lista3.adiciona(dupla,níveldado)
    retorna lista3
```

Um laço direto é feito na lista de retorno da fase anterior junto à lista de retorno externo do usuário. As estruturas são comparadas através de uma função da biblioteca de expressão regular. Para cada classe fixada do dicionário 1 todas as classes do dicionário 2 são analisadas. O resultado é retornado em uma lista de tuplas depois de se ter feito a comparação de nomes em relação ao retorno externo.

Baseado em ESTRUTURA

Primeira fase

A primeira fase baseia-se na comparação das descrições das superclasses como mostra o algoritmo abaixo. Faz comparação da existência de semelhança real entres duas descrições de superclasses candidatos ao alinhamento calculando a porcentagem da semelhança entre elas.

comparaDescricaoLivre(dic1:dicionario;dic2:dicionario)

```
        lista =[ ]
        para cada nome d1 na lista de chaves de dic1
        para cada nome d2 na lista de chaves de dic2
descri1 = buscaDescricaoClasse(buscaSuperClasse(nome d1))        descri2        =
buscaDescricaoClasse(buscaSuperClasse(nome d2))
          porcent = comparaDescricao(descri1;descr2)
          verifica porcent
                maior ou igual 9
                      lista.adiciona([d1;d2];otimo)
                maior ou igual a 7 menor que 9
                      lista.adiciona([d1;d2];bom)
              maior ou igual 5 menor 7
                      lista.adiciona([d1;d2];medio)
              maior ou igual 3 menor 5
                      lista.adiciona([d1;d2];baixa)
            maior ou igual 1 menor 3
                      lista.adiciona([d1;d2];ruim)
            menor 1 maior 0
                      lista.adiciona([d1;d2];excluir)
           igual  ou menor 0
                    continua
        retorna lista
```

geraSemiOntologia(lista)

Um laço direto é feito nos dicionários de retorno da fase de parser. As estruturas são comparadas através de uma função da biblioteca de expressão regular. Para cada classe fixada do dicionário 1 todas as classes do dicionário 2 são analisadas.

Ao final a função “geraSemiOntologia” é chamada para criação da meta ontologia configurada.

Segunda fase

A segunda fase baseia-se na comparação do resultado da primeira fase mais a resposta que o usuário informa ao sistema. Faz comparação da existência de semelhança real entres dois nomes de classes candidatos ao alinhamento aceitando diferenças (entre letras e espaços nas palavras) e comparando essas diferenças com nível de similaridade já retornado da fase anterior.

```
compraNomeMedio(lista:list;retorno:list)
              lista2 =[]
              para cada dupla existente em lista
              verifica se existe dupla em retorno
              verifica retorno na segunda posição para sim ou não/    sim
                        nivelado = comparaNomeDescri(dupla)
```

```
                    se nivelado diferente de excluir
                            lista2.adiciona(dupla,níveldado)
                if nivelado excluir e retorno sim
                            lista2.adiciona(dupla,níveldado)
            se não existe em retorno
                    nivelado = comparaNomeDescri(dupla)
                    se nivelado diferente de excluir
                        lista2.adiciona(dupla,níveldado)
        retorna lista2
```

geraSemiOntologia(lista)

Ao final a função “geraSemiOntologia” é chamada para criação da meta ontologia configurada.