Pós-Graduação em Ciência da Computação

Adriel Almeida Café

UMA ARQUITETURA COM IMPLEMENTAÇÃO PARA INTEGRAÇÃO SEMÂNTICA DE ONTOLOGIAS E BANCOS DE DADOS

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE
2015

Universidade Federal de Pernambuco
Centro de Informática
Pós-graduação em Ciência da Computação

Adriel Almeida Café

UMA ARQUITETURA COM IMPLEMENTAÇÃO PARA INTEGRAÇÃO SEMÂNTICA DE ONTOLOGIAS E BANCOS DE DADOS

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: *Frederico Luiz Gonçalves de Freitas*

RECIFE
2015

Agradecimentos

- Ao meu Senhor, pois, *"ao Rei dos séculos, imortal, invisível, ao único Deus sábio, seja honra e glória para todo o sempre" (1 Timóteo 1:17)*;
- À minha amada, Gabriela Matos, pelo grande incentivo, ajuda e por cuidar de mim, mesmo quando distante fisicamente;
- À minha família que sempre me apoiou a correr atrás dos meus sonhos;
- Ao meu orientador, Prof. Fred Freitas, por me conduzir nessa incrível jornada;
- Ao meu amigo, Filipe Santana, pois sem ele não teria chegado tão longe;
- Por fim, aos meus amigos que me acompanharam nesse grande desafio.

“Veni, vidi, vici”

—JÚLIO CÉSAR

Resumo

Integração da informação é uma área de pesquisa ativa a qual tenta-se criar mecanismos para extrair ou mesclar fontes de dados. Integração semântica é uma subárea da integração da informação. Nela são utilizadas tecnologias da Web Semântica para tornar o processo de integração o mais inteligente e automático possível.

Para realizar uma integração semântica, são utilizadas ontologias que representam formalmente o vocabulário utilizado para descrever o conteúdo de fontes de dados. Por conta disso, elas são frequentemente utilizadas para enriquecer o processo de integração. Bancos de dados relacionais são utilizados para armazenar e recuperar grandes quantidades de dados de forma rápida e eficiente. Devido à aplicação da estratégia relacional em diversos sistemas com propósitos diferentes, quando há a necessidade de fazê-los trocar dados (dentro de um mesmo domínio), utilizar uma estratégia de integração semântica é recomendado pela praticidade e versatilidade.

Este trabalho tem como objetivo propor uma arquitetura para integração semântica de fontes de dados heterogêneas dentro de um mesmo domínio. Esta arquitetura propõe a realização de uma integração virtual, mediada por consultas, entre ontologias e bancos de dados relacionais.

Como prova de conceito foi desenvolvido o Gryphon, um framework Java de código aberto que implementa a arquitetura proposta. O Gryphon Framework (semi) automatiza as principais atividades relacionadas à integração semântica: alinhamento de ontologias, mapeamento de banco de dados e reescrita de consultas.

Para verificar a capacidade e a praticidade de realizar integração semântica com a arquitetura proposta e o Gryphon Framework, foram realizados dois experimentos. No primeiro experimento, foram criadas consultas por um biólogo especialista do domínio biológico para ontologias formais ricamente axiomatizadas (Gene Ontology, Protein Ontology, Chemical Entities of Biological Interest e BioTopLite2). Essas ontologias foram utilizadas para integrar e consultar o banco de dados UniProt/SwissProt. No segundo experimento, foram integradas duas ontologias (SIOC e rNews) e dois bancos de dados (Joomla e WordPress) do domínio de notícias utilizando a ontologia News como camada semântica.

Palavras-chave: Integração da Informação, Integração de Dados, Alinhamento de Ontologias, Mapeamento de Banco de Dados, Reescrita de Consultas.

Abstract

Integration of information is an active research area that attempts to create mechanisms to merge data sources, or retrieve data from distributed sources. Semantic integration as a subarea, frequently uses semantic web technologies to make the integration process the most intelligent and automatic as possible.

To perform a semantic integration, ontologies are used to formally represent the vocabulary used to describe the content of data sources. Ontologies are often used to enrich the process of integration. Relational databases are used to store and retrieve large amounts of data quickly and efficiently. Due to the application of relational strategy in various systems with different purposes, when there is a need to make them exchange data (within the same domain), a semantic integration strategy is recommended by its practicality and versatility.

This work aims at proposing an architecture for semantic integration of heterogeneous data sources within the same domain. With the architecture, it is proposed to conduct a virtual integration, mediated by queries, between ontologies and relational databases.

As proof of concept, the Gryphon was developed. It is an open source Java framework which implements the proposed architecture. The Gryphon Framework (semi) automates key activities related to semantic integration: alignment of ontologies, database mapping and query rewrite.

To verify the ability and practicality of performing semantic integration with the proposed architecture under Gryphon Framework, two experiments were conducted. In the first experiment queries were created by an expert in the biological domain to retrieve data integrated with the support of formal and richly axiomatized ontologies (Gene Ontology, Protein Ontology, Chemical Entities of Biological Interest and BioTopLite2). These ontologies were used to integrate and query the UniProt/SwissProt database. In the second experiment were integrated two ontologies (SIOC and rNews) and two databases (Joomla and WordPress) from news domain using the News ontology as semantic layer.

Keywords: Information Integration, Data Integration, Ontology Alignment, Database Mapping, Query Rewriting

Lista de Figuras

Lista de Quadros

Lista de Acrônimos

R2RML RDB to RDF mapping language 40

RDBMS Relational DataBase Management System 37

RDFS Resource Description Framework Schema 26

RDF Resource Description Framework 21

RO Relation Ontology 61

SIOC Semantically-Interlinked Online Communities 70

SO Sequence Ontology 61

SPARQL-DL SPARQL Protocol and RDF Query Language - Description Logics 46

SPARQL SPARQL Protocol and RDF Query Language 21

SQL Structured Query Language 27

SQWRL Semantic Query-Enhanced Web Rule Language 93

Turtle Terse RDF Triple Language 24

UniProt Universal Protein Resource 61

URI Uniform Resource Identifier 25

W3C World Wide Web Consortium 23

XML eXtensible Markup Language 21

Sumário

1

Introdução

A web semântica surgiu em 2001, em um artigo publicado por BERNERS-LEE; HENDLER; LASSILA (2001). Seu intuito é permitir que as máquinas possam compreender o que são os dados e como eles são organizados. Até então, a web somente salva e recupera informações para os usuários.

A recuperação inteligente da informação é uma das principais características da web semântica (USCHOLD, 2004). Ao utilizar os padrões e as ferramentas criadas para esse propósito, tarefas exaustivas de interpretação, combinação e aplicação de filtros são automatizadas, tornando essas atividades cada vez mais simples.

Integração da informação é uma área de pesquisa ativa que tem por objetivo a criação de mecanismos para extrair dados ou mesclar fontes de dados (UZDANAVICIUTE; BUTLERIS, 2011). Integração semântica, é uma subárea da integração da informação, na qual são utilizados métodos criados para manipular o conteúdo da web semântica, tornando o processo de integração o mais inteligente e automático possível (NOY, 2004).

Integração semântica é o processo que se utiliza de uma representação conceitual sobre os dados e seus relacionamentos, de forma a reduzir ou eliminar possíveis heterogeneidades. Uma das formas de representar o conteúdo incluído em uma base de dados se dá por meio do uso de ontologias (CRUZ; XIAO, 2005). Nesse contexto, são utilizadas para descrever como as entidades do domínio se comportam em relação umas às outras. Com isso, é disponibilizado um vocabulário de alto nível com descrições que incorporam o esquema de dados em uma interpretação do mundo real (CALVANESE, 2002).

A literatura descreve as ontologias como um artefato primordial para o enriquecimento do processo de integração (ALASOUD; HAARSLEV; SHIRI, 2009; BERGAMASCHI et al., 2001; GAGNON, 2007; NOY, 2004; UZDANAVICIUTE; BUTLERIS, 2011). Bancos de dados relacionais são utilizados para armazenar e recuperar grandes quantidades de dados de forma rápida e eficiente (ASTROVA, 2004). Entretanto, o emprego da estratégia relacional em diversos sistemas com propósitos diferentes resulta em diferenças na representação de determinadas entidades como, por exemplo, um sistema financeiro possui semelhanças com um sistema de gestão escolar, como conceitos de pessoas e estoque, mas somente essas semelhanças não

enriquecem devidamente o processo de integração. Quando há a necessidade de fazer sistemas dessa natureza trocar informações (dentro de um mesmo domínio), utilizar uma estratégia de integração semântica é recomendado pela praticidade e versatilidade oferecidas pelas ontologias (AUER; IVES, 2007).

No entanto, realizar integração semântica não é apenas unir bancos de dados por meio de uma ontologia. Sobre um mesmo domínio, bancos de dados podem ser complementares. O mesmo acontece para ontologias, *e.g.*, um banco de dados incluindo o registro de automóveis e outro as rotas que esses automóveis realizam. Como ontologias têm escopo definido sobre o conteúdo representado, *i.e.* propósito, realizar a integração de apenas um banco de dados pode requerer o uso de ontologias diferentes, por exemplo, para representar os automóveis e para descrever como estes podem se locomover em uma cidade.

Assim, ao integrar fontes de dados, utilizando uma estratégia de integração semântica, é necessário resolver o problema de heterogeneidade dos dados, além de possíveis problemas de heterogeneidade entre as próprias ontologias. Para isso, a literatura descreve soluções (separadamente) que permitem:

- Alinhar ontologias (EHRIG; STAAB; SURE, 2005; LI et al., 2009; NOY; MUSEN, 2000; DAVID et al., 2011);
- Descrever entidades de bancos de dados como classes das ontologias (JESúS BARRASA ÓSCAR CORCHO, 2004; CULLOT; GHAWI; YéTONGNON, 2007; BIZER, 2004); e
- Consultar bancos de dados utilizando o vocabulário das ontologias, traduzindo automaticamente consultas à ontologia para consultas aos bancos (BIZER, 2004; KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013).

Nesse contexto, este trabalho tem como objetivo propor uma arquitetura para integração semântica de fontes de dados heterogêneas dentro de um mesmo domínio. Esta arquitetura propõe a realização de uma integração virtual, mediada por consultas, entre ontologias e bancos de dados relacionais simultaneamente. Dessa forma, as fontes são mantidas com sua constituição original, permitindo evoluções futuras.

Como prova de conceito foi desenvolvido o Gryphon[1], um *framework* escrito na linguagem Java[2], de código aberto, que implementa a arquitetura proposta. O Gryphon (semi) automatiza as principais atividades relacionadas à integração semântica, *e.g.*, alinhamento de ontologias, mapeamento de banco de dados e reescrita de consultas.

Para verificar a capacidade e a praticidade de realizar integração semântica com a arquitetura proposta e o Gryphon Framework, foram realizados dois experimentos. No primeiro experimento, foram criadas consultas por um especialista do domínio biológico para as seguintes

[1] https://github.com/adrielcafe/GryphonFramework
[2] https://oracle.com/java

ontologias formais do mesmo domínio: Gene Ontology (ASHBURNER et al., 2000), Protein Ontology (NATALE et al., 2011), Chemical Entities of Biological Interest (DEGTYARENKO et al., 2008) e BioTopLite2 (SCHULZ; BOEKER, 2013). Essas ontologias são utilizadas para permitir a integração do banco de dados UniProt/SwissProt (CONSORTIUM, 2008), que descreve características de proteínas. No segundo experimento, foram integradas duas ontologias (SIOC[3] e rNews[4]) e dois bancos de dados (do Joomla[5] e do WordPress[6]) do domínio de notícias utilizando a ontologia News[7] como camada semântica. Este experimento verificou todo o potencial da arquitetura proposta e do Gryphon Framework.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo desta proposta é propor e avaliar uma arquitetura para integração semântica entre ontologias e bancos de dados relacionais simultaneamente. Esta arquitetura deverá descrever detalhadamente todo o processo de integração virtual, mediada por consultas, de forma que seja possível utilizar uma camada semântica para diminuir os problemas de heterogeneidade de dados.

1.1.2 Objetivos Específicos

- Criar uma arquitetura que seja capaz de realizar integração semântica, mantendo as fontes de dados com suas características originais, utilizando ontologias como vocabulário de consulta;
- Implementar a arquitetura proposta, por meio de um *framework*, a fim de comprovar a eficácia da mesma;
- Identificar as principais estratégias para possibilitar a realização de tarefas de integração de bancos de dados e ontologias, incluindo subtarefas como alinhamento, mapeamento e reescrita de consultas;
- Identificar soluções existentes que realizem alguma subtarefa de integração para utilizá-los na implementação do *framework*;
- Definir uma estratégia de reuso das soluções existentes para tarefas de integração, de forma que estas possam ser utilizadas colaborativamente; e

[3] http://rdfs.org/sioc/spec/
[4] http://dev.iptc.org/files/rNews/rnews_1.0_draft3_rdfxml.owl
[5] http://joomla.org
[6] http://wordpress.org
[7] http://ebiquity.umbc.edu/ontology/news.owl

- Realizar um estudo de caso real utilizando dados biológicos.

1.2 Justificativa

Ao criar uma aplicação específica, a coleta de dados é realizada de forma direcionada à aplicação. Dessa forma, bancos de dados são criados para resolver, na maioria das vezes, problemas específicos. Entretanto, a Internet sempre teve como objetivo facilitar a troca de informações. Uma das formas de simplificar essas trocas é permitir que a informação seja localizada em qualquer lugar da Internet, acessível e entendível, tanto para os humanos quanto para as máquinas (PASSIN, 2004).

Nesse contexto, a recuperação inteligente da informação ganha grande notoriedade (BUSSLER et al., 2004). Como meio para recuperar informação de maneira inteligente, há a integração semântica.

A integração semântica, ao utilizar ontologias para representar uma determinada porção do domínio na qual os dados estão inseridos, minimiza ou elimina a heterogeneidade presente nos dados (SHADBOLT; BERNERS-LEE; HALL, 2006). Apesar de ser uma área já explorada, a procura e o desenvolvimento de abordagens e soluções de integração está crescendo (NOY, 2004; BERGAMASCHI et al., 2001; UZDANAVICIUTE; BUTLERIS, 2011; GAGNON, 2007; ALASOUD; HAARSLEV; SHIRI, 2009).

Um exemplo de domínio que vem sempre buscando soluções de integração é o domínio biomédico. Neste, há uma quantidade crescente de bancos de dados que incluem características diversas sobre os seres vivos, *e.g.*, o UniProt (CONSORTIUM, 2008) e o Ensembl (HUBBARD, 2002). Assim, estratégias que possibilitem a comunicação de diversas fontes podem permitir, além da recuperação de dados integrados, o estudo de características compartilhadas entre diversos seres vivos e os humanos, como realizar a comparação que o açúcar apresenta no metabolismo de diversos organismos diferentes.

Para tratar a integração semântica, existem abordagens que utilizam arranjos diferentes na composição das ontologias que são utilizadas como vocabulário para consultar os dados. Pode ser utilizada uma ontologia global (camada semântica), para integrar bancos de dados locais (CALVANESE; GIACOMO; LENZERINI, 2001; CALVANESE, 2002). Outra possibilidade é utilizar uma ou mais ontologias específicas, se comunicando, para descrever o conteúdo dos bancos de dados (KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013). Entretanto, esses trabalhos não descrevem a possibilidade de realizar integração num cenário mais amplo, onde há uma ontologia global utilizada como meio de aglutinar ontologias locais. Tais ontologias locais, por sua vez, representam o conteúdo específico de um ou mais bancos utilizados na integração. Cenário esse, frequentemente encontrado no domínio biológico, na qual são utilizadas ontologias genéricas, como a Basic Formal Ontology (SPEAR, 2006) ou a BioTopLite2 (SCHULZ; BOEKER, 2013) para organizar ontologias específicas de domínio, como a Gene Ontology (ASHBURNER et al., 2000), Protein Ontology (NATALE et al., 2011) e

Chemical Entities of Biological Interest (DEGTYARENKO et al., 2008). As ontologias mais genéricas fornecem os axiomas necessários para descrever adequadamente os dados dos bancos, que são descritos em termos das ontologias mais específicas. É nesse exemplo de cenário heterogêneo que este trabalho se insere.

Além da aplicação no domínio biológico, a integração semântica pode ser utilizada para, *e.g.*:

- Permitir a avaliação da saúde pública em tempo real (JI, 2014);
- Integrar modelos de simulação na área de petróleo e gás (SOMA et al., 2008); e
- Resolver problemas de heterogeneidade em especificações de *software* (RHODE, 2013).

1.3 Contribuições

Espera-se que este trabalho contribua das seguintes formas:

- Com uma nova estratégia para integração semântica capaz de extrair objetos (de ontologias) e dados (de bancos de dados) simultaneamente;
- Com uma ferramenta de código aberto que utilize esta nova estratégia para simplificar o processo de integração em qualquer aplicação;
- Com uma demonstração e incentivo ao reuso de *software* existente para realizar subtarefas de integração; e
- Por fim, com uma revisão da literatura que:
  - Apresente o estado da arte das soluções para integração semântica;
  - Exponha os problemas relacionados ao tema.

1.4 Organização do Trabalho

Os demais capítulos que compõem este trabalho estão organizados conforme a descrição a seguir.

O Capítulo 2, Referencial Teórico, tem como objetivo abordar os principais conceitos relacionados ao trabalho em questão. São tratados os conceitos necessários para a sua compreensão, relacionados à web semântica e integração semântica, bem como seus problemas e soluções.

O Capítulo 3, Trabalhos Relacionados, inicia uma discussão sobre as abordagens existentes na área da integração semântica, apontando suas principais caracteristicas.

O Capítulo 4, Arquitetura Proposta, apresenta a arquitetura desenvolvida neste trabalho. Neste capítulo, são descritos os componentes da arquitetura, o processo de integração e os cenários de integração. Além disso, são descritos os requisitos necessários para uma implementação bem-sucedida.

No Capítulo 5, Implementação, é apresentado o Gryphon Framework, solução que implementa a arquitetura proposta. Seu funcionamento e forma de usar são explicados detalhadamente. Também são descritas as soluções existentes que foram utilizadas no desenvolvimento do *framework*.

No Capítulo 6, Experimentos, são descritos dois experimentos que foram realizados neste trabalho a fim de solucionar problemas reais de integração de dados e comprovar a eficácia desta arquitetura e sua implementação.

Por fim, no Capítulo 7, Conclusões, são discutidas as conclusões do trabalho, limitações, contribuições e uma breve análise comparativa com os trabalhos relacionados.

2

Referencial Teórico

2.1 Web Semântica

Um dos principais objetivos da Internet é facilitar a troca de informações. Desde a sua criação, foram desenvolvidas tecnologias voltadas para o usuário, permitindo que o mesmo fosse capaz de executar determinadas tarefas mais facilmente.

Segundo BREITMAN (2005), a Internet se desenvolveu mais rapidamente como um meio para troca de documentos entre pessoas, ao invés de um meio que fomentasse a troca de informações processadas automaticamente. Como consequência, as máquinas nem sempre são capazes de entender (no sentido de interpretar) o conteúdo disponível na Internet da mesma forma que os humanos interpretam.

BREITMAN (2005) e PASSIN (2004) apontam o acesso e entendimento da informação por parte das máquinas como uma deficiência da web. As máquinas, em sua maioria, não são capazes de processar semanticamente as informações que têm acesso, consequentemente dependendo dos usuários. Segundo BERNERS-LEE; HENDLER; LASSILA (2001), ao invés de pensar na informação para os humanos, o ideal é pensar na máquina, tornando as máquinas mais inteligentes. Como resultado, é possível criar máquinas que auxiliem os humanos a tomarem decisões, facilitando a vida das pessoas de uma maneira transparente.

A ideia sobre uma web que permite o processamento do conteúdo que está sendo veiculado, *i.e.* web semântica, surgiu em 2001. No trabalho de BERNERS-LEE; HENDLER; LASSILA (2001), há uma discussão de como a web semântica pode se tornar uma extensão da web. A extensão pode permitir a criação de dados digitais com mais recursos (busca semântica, integração de dados, recomendação de produtos) e significado definido. Estes incrementos permitirão que humanos e computadores trabalhem de forma otimizada (tarefas podem ser automatizadas ou as máquinas podem fazer sugestões ao usuário). Dessa forma, as máquinas passariam a compreender (no sentido de processar) os dados que, até então, elas somente salvam e recuperam conforme as demandas dos usuários.

Nesse sentido, o objetivo maior da web semântica pode ser compreendido como um meio de tornar a informação, localizada em qualquer lugar da Internet, acessível e entendível, tanto

para os humanos quanto para as máquinas. PASSIN (2004) afirma que a web semântica é mais uma visão do que uma tecnologia, no sentido de aproximar o ser humano da máquina, por meio de uma comunicação mais natural e transparente.

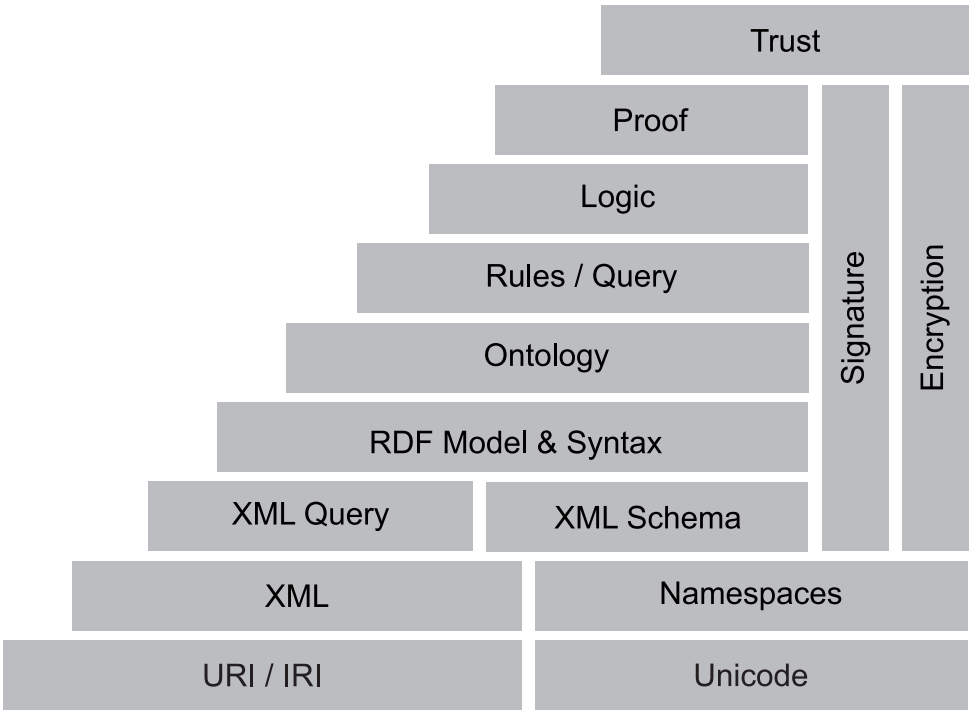
Segundo os autores citados, outra forma de estender a web atual é organizar seu conteúdo utilizando agentes inteligentes, permitindo que computadores e humanos trabalhem em cooperação. Nesse sentido, a web semântica tem como finalidade conseguir atribuir um significado ao conteúdo publicado não só na Internet, mas em qualquer meio eletrônico.

A recuperação inteligente da informação é uma das principais características da web semântica. Ao utilizar os padrões e as ferramentas criadas com tecnologias da web semântica, trabalhos exaustivos como interpretação, combinação e filtragem são automatizados tornando essas atividades cada vez mais simples.

Isso tudo é possível porque a web semântica, como o próprio nome diz, está ligada à semântica da informação. Segundo ALLEMANG; HENDLER (2011), a semântica é utilizada com uma noção da natureza, do significado. Para ANTONIOU; HARMELEN (2008), ela se refere à informação sobre o conteúdo de documentos disponíveis na web. Esse tipo de informação também é conhecido como metadado, onde dados são utilizados para descrever outros dados.

ANTONIOU; HARMELEN (2008) descrevem a organização da web semântica em camadas. Nelas, os recursos disponíveis para aplicações da web semântica são feitos em etapas, onde cada etapa é disposta como uma camada. A seguir, são descritas as principais camadas da web semântica, ilustradas na Figura 2.1.

Figura 2.1: Camadas da Web Semântica.



Fonte: ANTONIOU; HARMELEN (2008)

- **Camada de Apresentação (XML)**. Ela permite escrever documentos estruturados com facilidade. eXtensible Markup Language (XML) é frequentemente empregada no transporte de dados pela web (USCHOLD, 2004);
- **Camada de Dados (RDF *Model & Syntax*)**. O Resource Description Framework (RDF) é um modelo de dados básico que descreve assertivas sobre recursos na web (AUER; IVES, 2007). Está situado logo acima da camada XML, e apresenta a sintaxe conforme XML. Apesar disso, não é dependente da mesma, visto que existem outras representações em que é possível escrever RDF (ver Seção 2.1.3);
- **Camada de Representação (*Ontology*)**. Nessa camada, encontram-se linguagens para descrever ontologias (ver Seção 2.1.2). Estas linguagens expandem o esquema RDF para permitir uma representação mais complexas dos objetos (PASSIN, 2004). Essa camada é caracterizada pela precisão semântica, uma vez que as linguagens são bastante ricas em termos de expressividade (ALLEMANG; HENDLER, 2011);
- **Camada da Lógica (*Logic*)**. Esta camada apresenta todo o processo dedutivo. Nele são definidos os formalismos que permitem aos computadores realizar a inferência de informações, por meio de regras e restrições (PASSIN, 2004);
- **Camada da Prova (*Proof*)**. Nesta camada, encontram-se todos os mecanismos de avaliação da veracidade de uma informação. Além disso, são verificadas a consistência de dados oriundos da web semântica (PASSIN, 2004);
- **Camada da Confiança (*Trust*)**. A última camada está relacionada às assinaturas digitais e todo processo de certificação para a credibilidade das fontes de informações acessadas na web (PASSIN, 2004).

Dentre as soluções presentes na web semântica, destacam-se as ontologias e as tecnologias relacionadas às linguagens de representação do conhecimento, como Ontology Web Language (OWL) (HORROCKS et al., 2005), a linguagem de grafos RDF e a linguagem de consulta SPARQL Protocol and RDF Query Language (SPARQL) (KOLLIA; GLIMM; HORROCKS, 2011).

As ontologias serão abordadas na Seção 2.1.1. Na Seção 2.1.2 será descrito o formalismo de representação mais adotado, o OWL. Em seguida, na Seção 2.1.3, será descrito o formalismo utilizado para descrever indivíduos, *i.e.* instâncias, das classes e relações incluídas em ontologias: o RDF. Por último, na Seção 2.1.4, será descrita a linguagem de consulta para grafos em RDF, o SPARQL.

2.1.1 Ontologias

A ideia por trás do que se entende por ontologia foi descrito por Aristóteles, no século IV A.C., em *Organon*, caracterizado como um conjunto de escritos filosóficos, e que incluem a

discussão sobre a natureza e a estrutura da realidade (SPEAR, 2006).

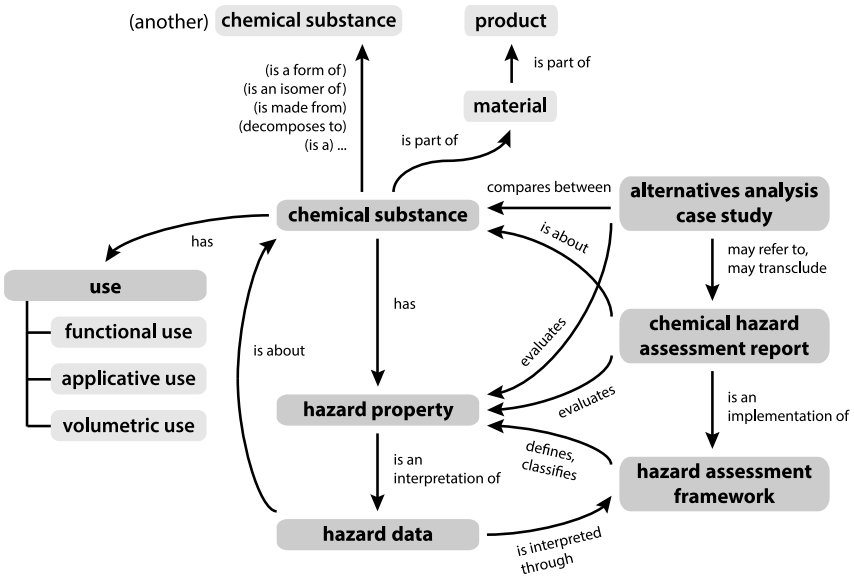
Em *Organon*, Aristóteles tentou criar a primeira forma de lógica, categorizando taxonomicamente objetos do mundo (FREITAS, 2003). Em seu trabalho, ele utilizou a lógica para descrever as coisas como elas são compreendidas pela mente humana. Entretanto, ele percebeu que a lógica por si só não seria suficiente para representar tudo.

Apesar de ter base filosófica, as ontologias vêm sendo tema de pesquisas em computação, pela demanda em categorizar, estruturar e representar entidades e conceitos de um certo domínio (KISHORE; SHARMAN, 2004). Ontologia pode ser definida de duas formas, como uma ciência e como um artefato representacional. SMITH (2003) define como a ciência do que existe, dos tipos e estruturas de objetos, eventos, propriedades, processos e relações em todas as áreas da realidade. Como um artefato, ontologias são descritas por meio de conceitos comuns, como composição, tempo, espaço, processos, além de vários outros (GUIZZARDI; HALPIN, 2008).

Outra definição, de cunho computacional, é dada por GRUBER (1993), o qual descreve que as ontologias são "uma especificação formal e explícita de uma conceitualização compartilhada". Quando o mesmo se refere à "Conceitualização", está tratando de um modelo abstrato, restrito ao seu domínio. Uma especificação "Explícita" significa que os conceitos e relacionamentos do modelo abstrato e suas restrições são dadas por termos explícitos (GRUNINGER; LEE, 2002). "Formal", por sua vez significa ser baseada em um formalismo de representação. Por fim, o termo "Compartilhada" se refere a ser desenvolvida com base em conceitos definidos de forma consensual entre membros de um domínio ou na literatura.

As ontologias são utilizadas na Inteligência Artificial (IA) para descrever teorias sobre como é estruturado o conhecimento, sendo auxiliado por mecanismos de raciocínio automatizado (SMITH et al., 2007). Um exemplo de ontologia pode ser descrito na Figura 2.2.

Figura 2.2: Exemplo de Ontologia.



Fonte: <http://kaios.net/research/organizing/>

A Figura 2.2 ilustra uma ontologia do domínio da avaliação de perigos químicos. Esta ontologia descreve classes (*product*, *material*, *chemical substance*) e as relações entre esses objetos (*hazard property*, *hazard data*, *use*). Neste exemplo, a classe *chemical substance* está ligada a classe *material* pela propriedade **"is part of"** (é parte de). Interpretando e exemplificando esta relação é possível afirmar que um conjunto de substâncias químicas chamadas "Amianto" são necessárias para criar o material "Fibrocimento" (utilizado em construção civil). A classe *material* também utiliza a propriedade **"is part of"** para descrever que é composta de tipos provenientes da classe *product*. Assim, por inferência, é possível descrever que "Fibrocimento" é **"is part of"** "Telha", um *product*. A classe *chemical substance* também se relaciona com a classe *hazard property* através da propriedade *has* (possui). A interpretação desta relação seria: a substância química "Amianto" possui uma *hazard property* do tipo "Câncer de Pulmão".

Para representar ontologias, se faz necessário empregar formalismos de representação que permitam a publicação de ontologias em um formato processável (BREITMAN, 2005). A Seção 2.1.2 irá abordar este assunto.

2.1.2 OWL

Para que as ontologias sejam criadas e representadas formalmente, é necessário que estas sejam descritas em uma linguagem formal. A linguagem OWL foi criada para cumprir este propósito. OWL é um formalismo de representação, processável por computadores, e que se utiliza da linguagem formal Description Logics (DL) para descrever classes, propriedades e axiomas (HORROCKS et al., 2005).

O formato OWL destaca-se por possuir semântica definida, com base sólida na DL HORROCKS et al. (2005). A OWL é a linguagem recomendada para a representação de ontologias pelo World Wide Web Consortium (W3C). Para JEAN-MARY; SHIRONOSHITA; KABUKA (2009), o formato OWL habilita a expressividade enquanto mantém completude computacional (para todas as computações se garante tempo finito).

A linguagem OWL é composta por diversos elementos: *namespace*, cabeçalhos, classes, indivíduos e propriedades (BREITMAN, 2005).

- *Namespaces* são utilizados para declarar quais vocabulários serão utilizados pela ontologia (BREITMAN, 2005). O vocabulário pode ser uma outra ontologia ou um grafo. Este tipo de definição evita erros de ambiguidade (SHVAIKO; EUZENAT, 2012).

- Com os cabeçalhos, é possível descrever a ontologia, importar outras ontologias ou controlar a versão da mesma (BREITMAN, 2005). Ou seja, é onde os metadados da ontologia são descritos.

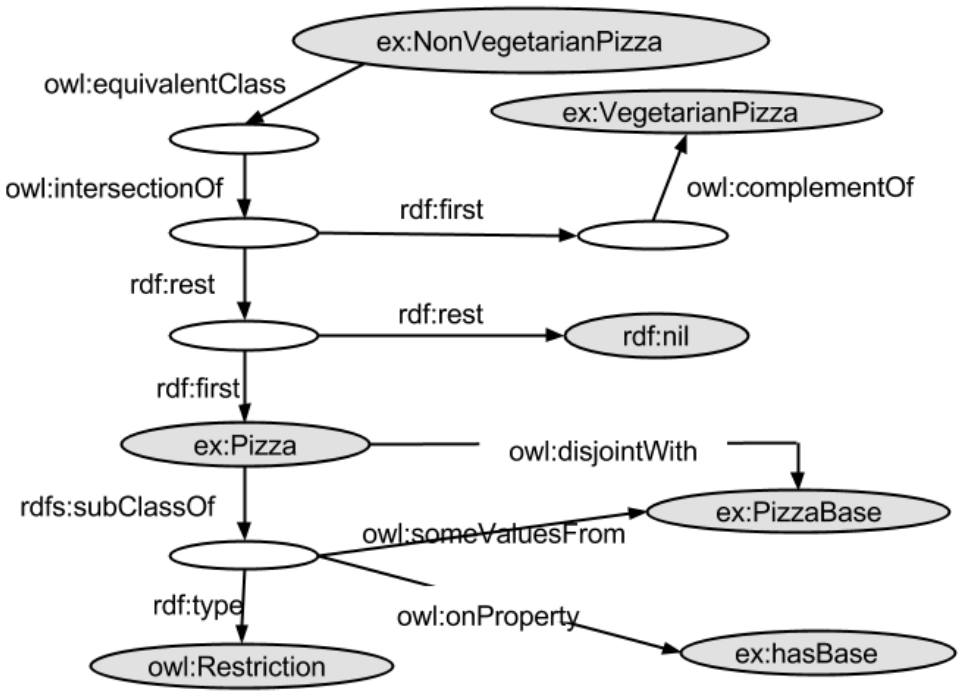
- Classes são utilizadas para descrever os conceitos de um domínio (BREITMAN, 2005). Cada indivíduo em OWL pertence a uma classe, ou seja, indivíduos são ins-

tâncias (objetos) das classes. Os indivíduos podem se relacionar a outros indivíduos por meio de propriedades.

- Propriedades servem para descrever relacionamentos entre classes (BREITMAN, 2005). Existem propriedades do tipo *object* que relacionam duas classes. Também existem propriedades do tipo *datatype* que relacionam indivíduos a literais, como *xsd:string*, *xsd:int* e *xsd:boolean*.

O exemplo da Figura 2.3 é de uma ontologia descrita com recursos de OWL. Neste exemplo, deduz-se que uma *ex:Pizza* precisa ter (*owl:onProperty*) pelo menos uma (*owl:someValuesFrom*) *ex:PizzaBase*, isto é uma restrição imposta ao utilizar a classe *owl:Restriction*. A propriedade *owl:disjointWith* impede que haja *ex:PizzaBase* repetidas em *ex:Pizza*.

Figura 2.3: Exemplo de uma Ontologia em OWL.



Fonte: <http://www.cs.wustl.edu/ jain/cse570-13/ftp/semantic/>

2.1.3 RDF

O RDF é uma linguagem declarativa para troca de dados na web. Ele pode ser considerado um elemento chave da web semântica. O RDF utiliza a sintaxe de XML para representar metadados e relacionamentos entre recursos da web (BREITMAN, 2005). Mas, não se limita a essa sintaxe. Também é possível escrever um documento RDF utilizando as sintaxes Terse RDF Triple Language (Turtle), Notation 3 (N3) e N-Triples.

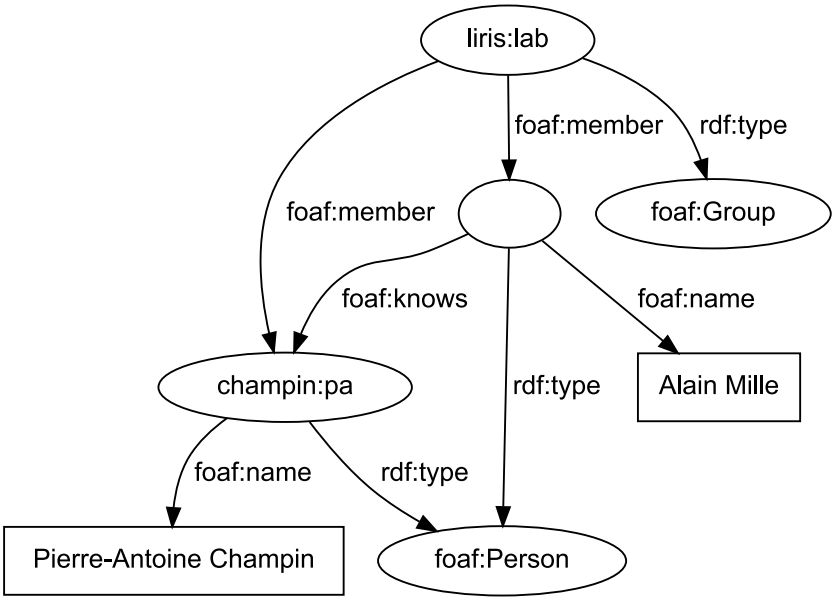
Segundo BREITMAN (2005), um dos objetivos de RDF é tornar a web mais acessível às máquinas, em termos de entendimento dos dados. O RDF acrescenta metadados nestes recursos, de modo a possibilitar a interpretação dos mesmos pelos computadores.

Alguns exemplos de utilização do RDF:

- Descrever páginas web para facilitar a recuperação por parte dos mecanismos de busca;
- Descrever conteúdos multimídia como imagem, áudio e vídeo;
- Descrever propriedades de produtos para compra *online*, como preço e disponibilidade;
- Descrever propriedades de notícias, como título e palavras-chave; e
- Descrever o relacionamento entre pessoas.

Com o RDF, é possível visualizar a estruturação de um conjunto de dados como um grafo. Na Figura 2.4, tem-se um exemplo de grafo utilizando recursos de RDF. Neste grafo, é possível deduzir que *champin:pa* é um indivíduo do tipo (*rdf:type*) pessoa (*foaf:Person*) e seu nome (*foaf:name*) é "Pierre-Antoine Champin". Esta pessoa é membro (*foaf:member*) do grupo (*foaf:Group*) *liris:lab*. Também é possível afirmar que essa pessoa conhece (*foaf:knows*) "Alain Mille", membro do mesmo grupo.

Figura 2.4: Exemplo de Grafo em RDF.



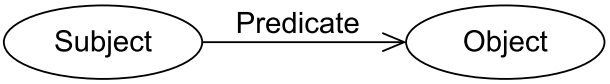
Fonte: <http://liris.cnrs.fr/ pchampin/2012/lod/rdf.html>

Para identificar recursos, RDF utiliza o Uniform Resource Identifier (URI), bem como liga os membros de uma tripla (*sujeiro-predicado-objeto*). Dessa forma, o RDF permite que dados estruturados e semiestruturados possam ser mesclados e compartilhados em diferentes aplicações, contribuindo para a interoperabilidade na troca de dados (ALLEMANG; HENDLER, 2011).

Conforme a Figura 2.5 ilustra, as triplas são formadas por três componentes (BREITMAN, 2005):

- **Sujeito**. Corresponde a qualquer identidade ou objeto. Ele é identificado por meio de um endereço único, o URI;
- **Predicado**. São as características (propriedades) do sujeito. Um sujeito pode ter uma ou mais propriedades. Também são identificados através de uma URI; e
- **Objeto**. É um objeto que pertence a um ou mais recursos. Tipos primitivos como cadeia de caracteres (*string*), números (*integer*, *float*) ou valores lógicos (*boolean*) podem ser utilizados. Também é possível referenciar instâncias por meio da sua URI.

Figura 2.5: Componentes de uma Tripla.

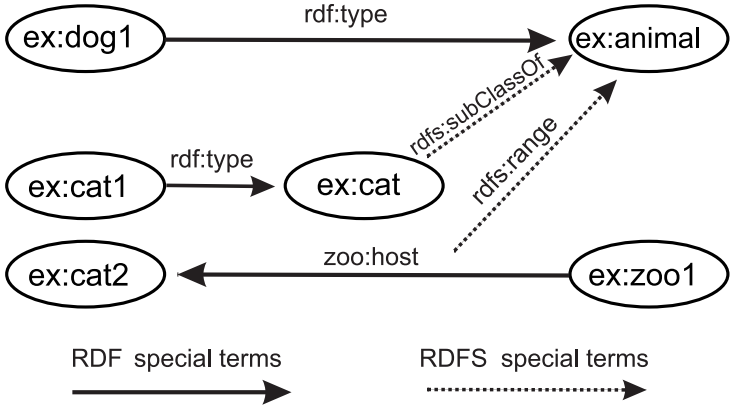


Fonte: <http://w3.org/TR/rdf11-concepts/>

RDF fornece um número limitado de elementos predefinidos, impossibilitando a criação de um vocabulário muito rico. A extensão do RDF, chamada de Resource Description Framework Schema (RDFS), permite a criação, por exemplo, de subclasses e subpropriedades. Com o RDFS, é possível criar heranças de classes e propriedade e descrevê-las mais detalhadamente, *e.g.*, informando o domínio (*domain*), o alcance da propriedade (*range*) ou criando uma etiqueta (*label*).

A figura Figura 2.6 ilustra os novos recursos introduzidos no RDFS. O *rdfs:subClassOf* permite a criação de heranças, *e.g.*, *ex:cat* é subclasse da classe *ex:animal*. A propriedade *rdfs:range* restringe o valor de outras propriedades, *e.g.*, a propriedade *zoo:host* apenas permite valores do tipo *ex:animal*.

Figura 2.6: Exemplo de Grafo em RDFS.



Fonte: <http://wikiwand.com/en/RDF$_S$*chema* >

Mesmo com os novos recursos do RDFS, este ainda não permite o uso de raciocínio até a camada de prova descrita pela web semântica (DOU; MCDERMOTT; QI, 2005). Devido a isso, a OWL é utilizada para este propósito.

2.1.4 SPARQL

Realizar consultas em ontologias ou grafos é muito importante no contexto da web semântica, pois, é por este mecanismo que usuários e aplicações podem interagir e extrair informações nestas fontes de dados.

SPARQL é uma linguagem de consulta que busca correspondências em grafos RDF. Foi desenvolvida para ser facilmente entendida por usuários que conhecem Structured Query Language (SQL), a linguagem mais utilizada de acesso a banco de dados relacionais (ALLEMANG; HENDLER, 2011). Uma consulta SPARQL é dividida, basicamente, em quatro partes (KOLLIA; GLIMM; HORROCKS, 2011):

- **Definição do Vocabulário**. No início da consulta, são definidos os prefixos. Cada prefixo representa uma ontologia ou grafo. Prefixos são formados por um *namespace* e um URI;
- **Definição da Cláusula *SELECT***. Assim como na SQL, é na cláusula *SELECT* onde as variáveis são declaradas, *e.g.*, *?x* e *?Result*. As variáveis utilizadas no *SELECT* servem para dizer quais variáveis deverão retornar no resultado da consulta. Caso não seja necessário remover alguma variável do resultado, utiliza-se "*" para retornar todas as variáveis;
- **Definição da Cláusula *WHERE***. Esta cláusula é composta por uma ou mais triplas (sujeito, predicado e objeto). Este conjunto de triplas é chamado de Basic Graph Pattern (BGP). SPARQL tentará encontrar correspondências (*matching*) entre o BGP e a ontologia ou grafo RDF que estiver sendo pesquisada (SHVAIKO; EUZENAT, 2012); e
- **Definição dos Modificadores de Resultado**. É possível ordenar (*ORDER BY*), limitar (*LIMIT*) e paginar (*OFFSET*) o resultado da consulta.

A Figura 2.7 ilustra uma simples consulta SPARQL. Esta consulta possui apenas uma tripla, ela faz uma busca por instâncias que estejam relacionadas à propriedade *foaf:name*. Apenas o nome (*?name*) é retornado. O modificador de resultado *LIMIT* foi utilizado para limitar o número de resultados em dez.

Figura 2.7: Exemplo de uma Consulta em SPARQL.

```
PREFIX foaf:  <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
   ?person foaf:name ?name .
} LIMIT 10
```

Fonte: O Autor

2.2 Integração Semântica

Nesta seção, será apresentado brevemente o estado da arte relacionado à integração de dados, especificamente sobre os métodos baseados em ontologias (integração semântica).

Integração de dados é o problema de combinar dados de diferentes fontes e prover para o usuário uma visão unificada destes dados (LENZERINI, 2002). Para CRUZ; XIAO (2005) a integração de dados fornece a capacidade de manipular dados de forma transparente entre múltiplas fontes de dados heterogêneas.

LENZERINI (2002) afirma que criar sistemas de integração de dados é um desafio importante para aplicações de todo o mundo. Sistemas de integração são relevantes em diversos contextos, *e.g.*, integração de informações corporativas, integração de dados biomédicos, sistemas de informação geográfica, e aplicações de comércio eletrônico (*e-commerce*) (CRUZ; XIAO, 2005).

Uma forma de realizar integração de dados é por meio da integração semântica. Este é o processo de utilização de uma representação conceitual dos dados e seus relacionamentos, permitindo eliminar possíveis heterogeneidades (CRUZ; XIAO, 2005). Segundo CRUZ; XIAO (2005), esta representação conceitual se dá por meio de ontologias, devido à sua capacidade de representação citada anteriormente.

De acordo com ZIEGLER; DITTRICH (2004), o objetivo da integração semântica é agrupar, combinar ou completar os dados de diferentes fontes, tendo em vista o uso de uma representação semântica explícita, evitando que dados semanticamente incompatíveis sejam utilizados. Ou seja, a integração semântica precisa assegurar que (apenas) dados relacionados ao domínio sejam utilizados.

Para LINKOVá (2007), as ontologias e bancos de dados estão estreitamente relacionados. A principal diferença é o propósito. Ontologias são utilizadas para descrever o significado dos termos usados em um domínio, já os bancos de dados apresentam um modelo, geralmente, específico voltado para a solução (LINKOVá, 2007).

O que distingue a integração semântica da integração de dados é a forma como os dados heterogêneos são tratados. Para KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV (2013), devido às possibilidades de representação que as ontologias possuem, a descrição do

conteúdo é mais rica do que em uma integração de dados convencional. Esta descrição mais rica em detalhes, *i.e.* axiomatizada, aumenta as chances de uma integração ser mais bem-sucedida (são extraídos dados mais relevantes e precisos).

Baseado nesse contexto, as próximas subseções são dedicadas a abordar os tipos de problemas que essa área de pesquisa se propõe a resolver e as soluções existentes.

2.2.1 Problemas Relacionados

Em um cenário real, diferentes fontes de dados são criadas, organizadas e mantidas por diferentes organizações, com propósitos diferentes. Não deve-se esperar que a mesma informação esteja presente da mesma forma em diferentes fontes de dados. Pelo contrário, as informações serão representadas de diferentes formas e níveis de abstração, ocasionando a heterogeneidade dos dados (CALVANESE, 2002).

Um dos grandes esforços para os novos sistemas de informação baseados em ontologias está em resolver problemas em fontes de dados heterogêneas (SMITH, 2003). Os pesquisadores aplicam ontologias tanto para apoiar a interoperabilidade semântica, integrando fontes de dados com vocabulários diferentes, como para visualização dos dados a partir de uma perspectiva diferente (UZDANAVICIUTE; BUTLERIS, 2011).

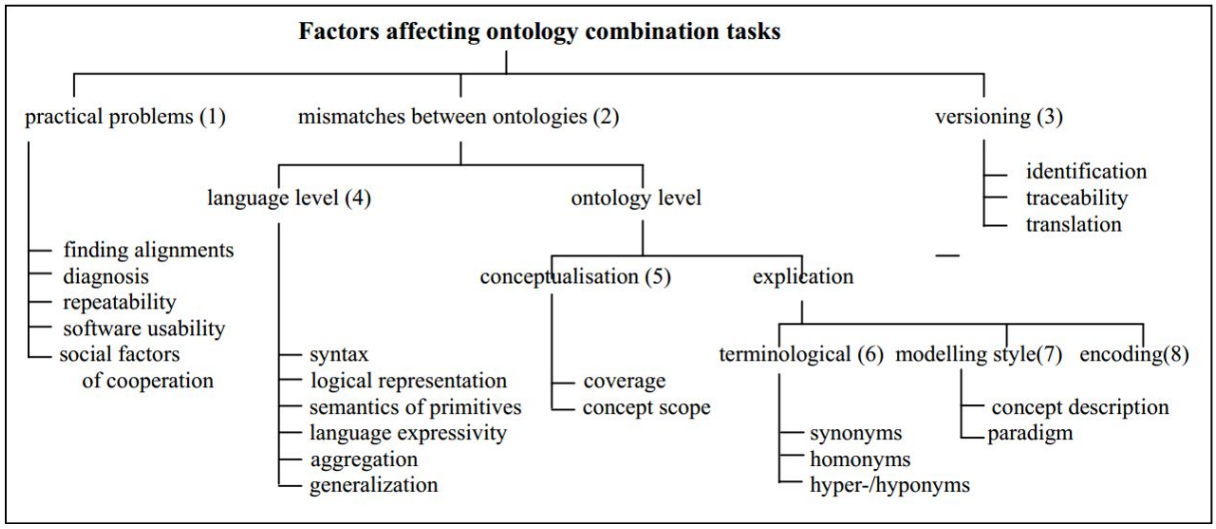
Segundo CRUZ; XIAO (2005), as fontes de dados podem ser heterogêneas em relação a:

- **Sintaxe**. É causada pelo uso de diferentes modelos ou linguagens (como OWL ou RDF);
- **Estrutura**. Ocorre devido as diferenças no esquema de dados (diferentes classes usadas para representar o mesmo domínio); e
- **Semântica**. Acontece quando há diferentes significados ou interpretações do dado em vários contextos (duas classes com o mesmo nome, mas que representam diferentes conceitos).

CRUZ; XIAO (2005) afirmam que, para alcançar a interoperabilidade dos dados, estes problemas precisam ser eliminados.

Diversos outros problemas relacionados à integração semântica são descritos na Figura 2.8 por KLEIN (2001).

Figura 2.8: Fatores que afetam a integração semântica.



Fonte: KLEIN (2001)

Dentre os problemas apresentados na Figura 2.8, os mais relevantes para o contexto de integração semântica são os problemas relacionados a:

- **Nível de linguagem**. Problemas relacionados à sintaxe podem ocorrer, entretanto, problemas maiores podem ocorrer, *e.g.*, construtores ou conceitos disponíveis em uma linguagem (como *owl:disjointWith* presente na linguagem OWL) podem não existir em outras linguagens (não é possível expressar a propriedade *owl:disjointWith* com a linguagem RDF); e

- **Nível de ontologia**. Mesmo quando as fontes possuem a mesma linguagem é possível ocorrer problemas. Quando as fontes utilizam o mesmo termo para descrever conceitos diferentes, *e.g.*, o termo "Empregado" pode ser descrito de formas diferentes em um contexto semelhante. Também ocorre o inverso, utilizar diferentes termos para descrever o mesmo conceito, *e.g.*, os termos "Estudante" e "Graduando" podem significar o mesmo em certos contextos. Outros possíveis problemas são descritos por KLEIN (2001), como o uso de diferentes convenções de modelagem, diferentes níveis de granularidade ou escopo.

No Quadro 2.1, duas ontologias são comparadas por NOY (2004). As ontologias AKT Reference Ontology[1] e eBiquity Person Ontology[2] descrevem o domínio de congressos e eventos acadêmicos. Elas estão no formato OWL. Nelas existem conceitos que representam pessoas, projetos e publicações acadêmicas.

[1] http://swl.slis.indiana.edu/repository/owl/aktportal.owl
[2] http://ebiquity.umbc.edu/ontology/person.owl

Quadro 2.1: Comparação entre duas ontologias para detectar problemas de heterogeneidade dos dados.

	AKT Reference Ontology	**eBiquity Person Ontology**
Diferentes nomes para o mesmo conceitos	*PhD-Student*	*PhDStudent*
Mesmo nome para diferentes conceitos	*Project* Apenas o projeto atual	*Project* Projetos passados e propostas de projeto
Escopo	Inclui *journals* e publicações compostas	Inclui estudantes e palestrantes
Diferentes convenções de modelagem	*Journal* é uma classe	*journal* é uma propriedade
Granularidade	*Professor-In-Academia*	*adjunct, affiliated, associate, principal*

Fonte: NOY (2004)

Como visto no Quadro 2.1, mesmo que duas ontologias simples descrevam o mesmo domínio, estas não estão livres de problemas de heterogeneidade. Por terem sido criadas em OWL, não existem problemas em nível de linguagem, entretanto, diversos problemas em nível de ontologia foram facilmente reconhecidos (NOY, 2004). Por exemplo, as duas ontologias possuem o conceito de *Journal*, mas em uma ontologia esse conceito é representado por uma classe, já na outra ontologia ele é representado como uma propriedade. Em outro exemplo, a primeira ontologia representa os professores em uma única classe, *Professor-In-Academia*, já a segunda ontologia utiliza quatro propriedades para representá-los: *adjunct*, *affiliated*, *associate* e *principal*.

Com base nos problemas apontados por CRUZ; XIAO (2005) e KLEIN (2001), é possível identificar fatores que podem prejudicar o processo de integração, independente do tamanho e domínio das fontes que serão integradas. Em consequência, a literatura descreve algumas soluções que permitem eliminar ou diminuir o impacto na integração semântica. A próxima seção abordará algumas destas soluções.

2.2.2 Soluções Existentes

Existem duas abordagens para integração de dados (LINKOVá, 2007):

- Ela pode ser materializada, *i.e.*, uma nova fonte de dados é criada a partir de uma integração com outras fontes; e
- Ou pode ser virtual, *i.e.*, os dados apenas são extraídos das fontes.

Na abordagem de integração materializada, uma cópia dos dados é feita (LINKOVá, 2007). Então, quando houver a necessidade de inserir novos dados, os dados previamente

copiados precisam ser mesclados com os novos dados. A consistência e qualidade dos dados precisam ser avaliadas sempre que novos dados forem inseridos.

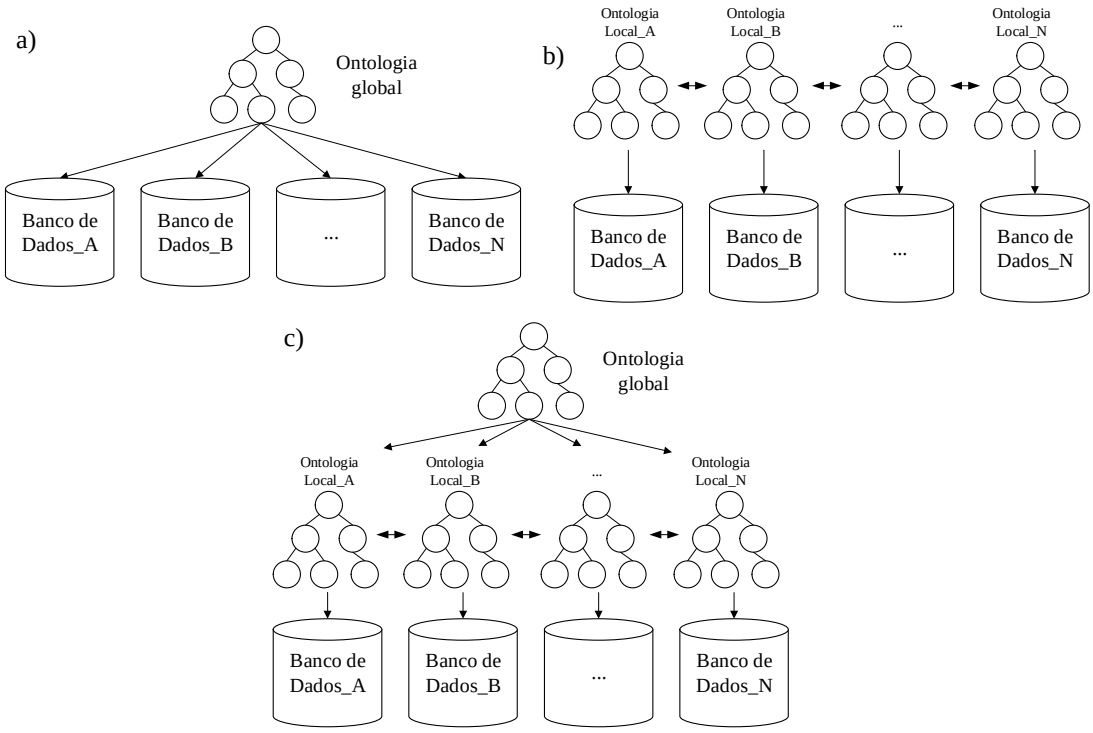
A abordagem de integração virtual tem como base uma interface para extração de dados em diferentes fontes (LINKOVá, 2007). Esta abordagem é indicada em integrações de grandes fontes de dados (onde replicar os dados em uma nova fonte torna-se inviável devido à quantidade de dados ou dificuldade de manutenção) ou fontes de dados que mudam com bastante frequência (como alterações no esquema ou nos tipos de dados).

Estas duas abordagens supracitadas mostram como os dados extraídos da fonte podem ser utilizados. Existem outras abordagens que descrevem profundamente como os sistemas de integração devem ser construídos de modo que seja possível extrair dados relevantes das fontes.

2.2.2.1 GAV, LAV e GLAV

As abordagens mais referenciadas para construção de sistemas para integração de dados são o Global as View (GAV), Local as View (LAV) e Global-Local as View (GLAV) (ALASOUD; HAARSLEV; SHIRI, 2009). O GAV consiste em mapear os conceitos do esquema global com os conceitos dos esquemas locais, já o LAV consiste em definir os esquemas locais como um conjunto de *views* que são mapeadas com o esquema global. O GLAV é uma abordagem híbrida, nela existem os dois tipos de mapeamentos. Estas três abordagens são descritas na Figura 2.9 e abordadas mais profundamente a seguir.

Figura 2.9: Abordagens para integração: a) GAV, b) LAV e c) GLAV.



Fonte: WACHE et al. (2001)

A abordagem GAV (Figura 2.9.a) apresenta o princípio descrito anteriormente em que

uma ontologia global é utilizada como vocabulário de consulta único entre diversas fontes. Portanto, realizar estratégias de consultas é uma tarefa simples no GAV. Entretanto, conforme os esquemas locais vão mudando, é necessário revisar os mapeamentos. Por outro lado, a abordagem LAV (Figura 2.9.b) permite a troca dos esquemas locais sem afetar o esquema global, isso acontece pelo fato de que os esquemas locais são definidos como *views* sobre as fontes locais, mas isso torna difícil o processamento de consultas (pode ser necessário reescrever a consulta duas vezes ou a *view* pode não expressar corretamente a fonte local). Por fim, o modelo híbrido GLAV (Figura 2.9.c) apresenta uma ontologia global que aglutina ontologias locais que representam e mapeiam o conteúdo de bancos de dados específicos.

Comparado ao LAV, o GAV permite a criação de uma estratégia simples e prática para mediação de consultas (Seção 2.2.2.4).

2.2.2.2 OIS e OBDA

Para realizar tarefas de integração semântica, o Ontology Integration System (OIS) foi descrito por CALVANESE; GIACOMO; LENZERINI (2001) como um *framework* formal para integração de ontologias. Formalmente, um OIS é uma tripla $\{G, S, M_{G,S}\}$, onde G é a ontologia global, S é o conjunto de ontologias locais e $M_{G,S}$ é o mapeamento entre G e S. No OIS existem três componentes principais:

- **Ontologia Global**. Fornece uma visão global e unificada das ontologias locais;
- **Ontologias Locais**. São ontologias heterogêneas e independentes. É onde estão as instâncias e axiomas; e
- **Alinhamentos**. *Link* semântico que relaciona os conceitos da ontologia global com os conceitos das ontologias locais.

Uma abordagem posterior, o Ontology-Based Data Access (OBDA), apresenta como funcionalidade principal a possibilidade de realizar integração por meio de uma ontologia global como vocabulário para consultar bancos de dados locais (KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013). No OBDA, uma ontologia descreve um esquema global para os bancos de dados relacionais, e provê um vocabulário para realização de consultas. A ontologia é mapeada com os bancos de dados, e as consultas feitas para as ontologias são traduzidas em consultas SQL e executadas nos bancos de dados. No OBDA existem três componentes principais:

- **Ontologia Global**. Fornece uma visão global e unificada dos esquemas dos bancos de dados locais;
- **Bancos de Dados Locais**. São fontes de dados heterogêneas e independentes. É onde estão os dados; e

- **Mapeamentos**. *Link* semântico que relaciona os conceitos da ontologia global com o esquemas dos bancos de dados locais.

Tanto no OIS quanto no OBDA, existe uma ontologia que representa um esquema global e unificado, as fontes locais (que podem ser outras ontologias ou bancos de dados) e o *link* semântico (alinhamentos e mapeamentos) entre a ontologia global e estas fontes locais.

Nos dois casos, existe uma camada semântica que garante um alto nível representacional sobre a camada de dados. Por meio dos links semânticos, é possível relacionar os conceitos do esquema global com os conceitos dos esquemas locais, diminuindo assim os problemas relacionados à heterogeneidade dos dados (DOAN; MADHAVAN, 2002).

2.2.2.3 Alinhamento e Mapeamento

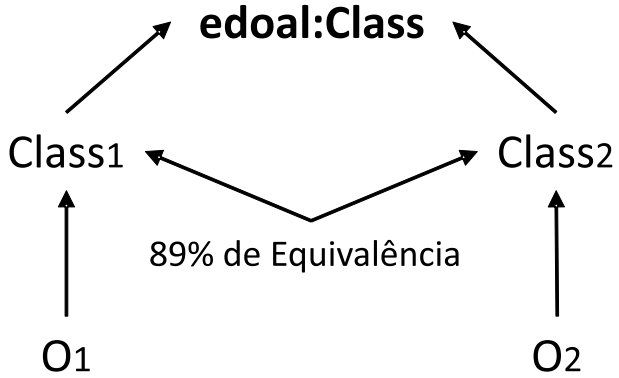
Os alinhamentos (*matching*) representam a correspondência semântica entre elementos de duas ontologias (classes, propriedades e relacionamentos) DAVID et al. (2011). Eles podem ser criados manualmente ou semiautomaticamente, com o auxílio de soluções já existentes. DAVID et al. (2011).

Segundo LI et al. (2009), dadas duas ontologias $O1$ e $O2$, o alinhamento encontra, para cada entidade em $O1$, uma entidade correspondente em $O2$. $O1$ é conhecida como a ontologia fonte e $O2$ como ontologia alvo.

De acordo com SEDDIQUI; AONO (2009), o alinhamento A é definido como um conjunto de correspondências com quadruplas $< E, F, R, L >$, onde E e F são as duas entidades alinhadas, R representa a relação entre essas entidades (como igualdade ou sentido oposto) e, L representa o nível de confiança deste alinhamento (valor que varia de 0 a 1, onde 0 significa pouca confiança e 1 total confiança).

Uma visão esquemática do processo de alinhamento é apresentada na Figura 2.10.

Figura 2.10: Representação de um alinhamento entre duas classes de ontologias distintas.



Fonte: O Autor

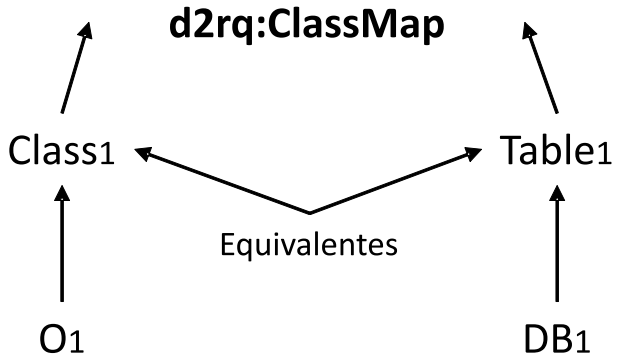
A Figura 2.10 representa o alinhamento entre duas classes (*Class*1 e *Class*2) de duas ontologias ($O1$ e $O2$) distintas. Neste exemplo, a similaridade entre estes recursos é descrita por

meio de *edoal* : *Class*. Expressive and Declarative Ontology Alignment Language (EDOAL) é uma linguagem que permite representar a correspondência semântica entre entidades de diferentes ontologias (DAVID et al., 2011).

O mapeamento é similar ao alinhamento, a diferença é que o mapeamento acontece entre a ontologia global e os bancos de dados locais. De acordo com CULLOT; GHAWI; YéTONGNON (2007), no processo de mapeamento, os componentes do banco de dados (tabelas, colunas e restrições) são comparados com os conceitos da ontologia (classes, propriedades e relacionamentos), a fim de encontrar similaridades. Neste contexto, o mapeamento pode ser descrito como um conjunto de correspondências entre os componentes do banco de dados e os conceitos da ontologia (CULLOT; GHAWI; YéTONGNON, 2007).

Uma visão esquemática do processo de mapeamento é apresentada na Figura 2.11.

Figura 2.11: Representação de um mapeamento entre uma coluna de um banco de dados e uma propriedade de uma ontologia.



Fonte: O Autor

No exemplo da Figura 2.11, é realizado um mapeamento entre a tabela *Table*1 de um banco de dados e uma classe *Class*1 de uma ontologia. A similaridade desta coluna com a propriedade é descrita com *d2rq* : *PropertyBridge*. D2RQ é uma plataforma para acessar bancos de dados relacionais por meio de consultas SPARQL (BIZER, 2004). Ela possui uma linguagem de mapeamento para descrever a similaridade entre os conceitos da ontologia com o esquema do banco de dados.

DAVID et al. (2011) apontam alguns benefícios relacionados aos alinhamentos e mapeamentos:

- Do ponto de vista da web semântica, é possível, dinamicamente, pesquisar e reusar alinhamentos e mapeamentos existentes por meio do uso de web services semânticos especializados na descoberta de serviços (USCHOLD, 2004);
- Do ponto de vista da engenharia de software, os alinhamentos e mapeamentos podem ser utilizados entre vários sistemas para facilitar a integração de seus bancos de dados (TRINKUNAS; VASILECAS, 2007); e

- Por fim, do ponto de vista da engenharia de ontologias, os alinhamentos estarão envolvidos com o ciclo de vida das ontologias. A partir do momento que os conceitos da ontologia são bem modelados e descritos, será mais fácil alinhá-los, ou mapeá-los, de forma automática (SEDDIQUI; AONO, 2009).

Os alinhamentos e mapeamentos desempenham um papel crucial na reescrita (transformação) de consultas, uma etapa importante no processo de integração semântica (LINKOVá, 2007). Outra função dos alinhamentos é mesclar ontologias, assim como os mapeamentos podem ser utilizados para mesclar bancos de dados (EHRIG; STAAB; SURE, 2005).

EHRIG; STAAB; SURE (2005) apontam alguns requisitos fundamentais para que um algoritmo de alinhamento ou mapeamento seja eficiente:

- O resultado precisa ser de alta qualidade, ou seja, a maioria dos alinhamentos ou mapeamentos precisam ser coerentes;
- O processo de alinhamento ou mapeamento precisa ser eficiente e rápido, mesmo em grandes fontes de dados;
- Interação opcional com o usuário pode contribuir com resultados mais precisos;
- O algoritmo precisa ser flexível para respeitar os diferentes modelos e cenários; e
- Por último, o algoritmo deve possibilitar a extensão do mesmo. Dessa forma será possível adicionar novas regras e abordagens, melhorando o processo sob demanda.

2.2.2.4 Mediação de Consultas

Quando duas ou mais fontes de dados precisam ser integradas virtualmente, normalmente é preciso criar mecanismos que direcionem corretamente para o conteúdo respectivo de cada fonte, devido à heterogeneidade dos dados. Isto pode aumentar a complexidade da integração (KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013).

Para evitar a criação de diferentes consultas, existem técnicas para reescrevê-las utilizando alinhamentos (entre ontologias) ou mapeamentos (entre ontologias e bancos de dados). Este procedimento é conhecido como mediação de consultas. Para mediar consultas corretamente, é preciso ter, além dos alinhamentos e mapeamentos, um esquema global que englobe o conteúdo das fontes locais, ou proporcione a organização destas. Como esquema global, frequentemente é utilizada uma ontologia. Nesse sentido, a ontologia global precisa ser alinhada com as ontologias locais; e, mapeada com os bancos de dados locais, criando o meio para reescrever as consultas.

Basicamente, uma consulta é construída e realizada para a ontologia global, depois a mesma é reescrita para as fontes de dados locais com a ajuda de um tradutor que utiliza os mapeamentos como referência. Essa estratégia permite que uma única consulta possa ser utilizada para extrair informações de múltiplas fontes de dados heterogêneas (KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013).

2.2.2.5 Reescrita de Consultas

Com o advento da integração semântica, a reescrita de consultas, que já era aplicada na integração de dados, vem ganhando cada vez mais atenção por cumprir um número crescente de tarefas como otimização de consultas (STOCKER et al., 2008), decomposição de consultas (QUILITZ; LESER, 2008), tradução de consultas (BHOWMICK; KüNG; WAGNER, 2009), inferência com lógica de descrição (JING; JEONG; BAIK, 2008) e, por último, na própria integração semântica (LYTRAS et al., 2009).

O processo de reescrita de consultas desempenha um papel fundamental na integração semântica, mais precisamente na mediação de consultas (LINKOVá, 2007).

A reescrita utiliza a ontologia global e o conjunto de alinhamentos (para as ontologias locais) ou mapeamentos (para os bancos de dados locais) (CRUZ; XIAO, 2005). A consulta original deve ser criada utilizando o vocabulário (conceitos) da ontologia global. Para cada fonte de dados local, esta consulta é reescrita utilizando o alinhamento ou mapeamento para encontrar e substituir os relacionamentos entre as fontes (CRUZ; XIAO, 2005).

Nos sistemas baseados na abordagem OBDA, por exemplo, a consulta, escrita com o vocabulário da ontologia global, é reescrita utilizando os mapeamentos. Estas novas consultas são delegadas ao Relational DataBase Management System (RDBMS), que por sua vez é responsável por avaliar e executá-las (RODRíGUEZ-MURO; KONTCHAKOV; ZAKHARYASCHEV, 2013).

Em termos formais, uma consulta q escrita com o vocabulário de uma ontologia T, deve ser reescrita em uma nova consulta q' para cada possível fonte A, utilizando o mapeamento m da mesma (KONTCHAKOV; RODRÍGUEZ-MURO; ZAKHARYASCHEV, 2013).

O exemplo a seguir demonstra como a reescrita de consultas funciona na mediação de consultas. A Consulta 2.1 ilustra uma consulta SPARQL feita com o vocabulário de uma ontologia. Na Consulta 2.2, uma nova consulta SPARQL foi criada para outra ontologia, por meio de um alinhamento entre a mesma e a primeira ontologia. De forma similar, também foi criada uma nova consulta SQL (Consulta 2.3), por meio de um mapeamento entre um banco de dados e a ontologia da Consulta 2.1.

Listing 2.1: Consulta SPARQL original.

```
PREFIX store: <http://ontology.owl/store.owl#>
SELECT ?name WHERE {
        ?x a store:Book . ?x store:name ?name
}
```

Fonte: O Autor

Na reescrita da consulta SPARQL, é possível observar que a classe *store* : *Book* (da Consulta 2.1) foi substituida pela classe *bookstore* : *Textbook* (da Consulta 2.2), assim como a propriedade *store* : *name* foi substituida pela propriedade *bookstore* : *title*.

Listing 2.2: Consulta SPARQL reescrita para ontologia.

```
PREFIX bookstore: <http://ontology.owl/bookstore.owl#>
SELECT ?name WHERE {
        ?x a bookstore:Textbook . ?x bookstore:title ?name
}
```

Fonte: O Autor

Na reescrita da consulta SQL, acontece algo semelhante. A classe *store* : *Book* (da Consulta 2.1) foi substituida pela tabela *books* (da Consulta 2.3), assim como a propriedade *store* : *name* foi substituida pela coluna *bookTitle*.

Listing 2.3: Consulta SQL reescrita para banco de dados.

```
SELECT bookTitle AS name FROM books;
```

Fonte: O Autor

2.3 Conclusão

Este capítulo abordou os principais conceitos relacionados ao trabalho em questão, necessários para sua compreensão. Foram tratados assuntos relacionados à web semântica (as camadas relevantes para trabalho, ontologias, OWL, RDF e SPARQL) e integração semântica (problemas de heterogeneidade, OIS, OBDA, mediação de consultas, entre outros).

No próximo capítulo serão abordadas algumas soluções existentes relacionadas a integração semântica. Estas soluções utilizam as tecnologias abordadas neste capítulo.

3

Trabalhos Relacionados

Diversos esforços estão sendo empregados na integração semântica. A literatura descreve algumas ferramentas da web semântica necessárias à integração. Algumas delas serão abordadas nas próximas subseções.

3.1 Integração de Ontologias

3.1.1 Aber-OWL

Aber-OWL (HOEHNDORF et al., 2015) consiste em um repositório de ontologias, um conjunto de *web services* e um *framework* que possibilitam a realização de consultas em ontologias do domínio biológico. É possível realizar consultas em SPARQL nas ontologias presentes no repositório. Os resultados das consultas são salvos no formato JavaScript Object Notation (JSON).

3.1.2 Exelixis

Exelixis (KONDYLAKIS; PLEXOUSAKIS, 2011) é uma plataforma *web* que permite a consulta em ontologias RDFS que estão em constante evolução sem a necessidade de redefinir mapeamentos. Após feito o primeiro mapeamento (manual), as mudanças entre as ontologias são automaticamente detectadas e descritas em uma linguagem específica. Ele possui uma *interface web* pela qual o usuário pode registrar diferentes versões das ontologias.

3.1.3 PROMPT

PROMPT (NOY; MUSEN, 2000) é uma ferramenta gráfica que provê uma abordagem semiautomática para alinhamento e mesclagem de ontologias (integração materializada). PROMPT executa algumas tarefas automaticamente e guia o usuário de modo a concluir outras tarefas com a sua intervenção. Em outras palavras, PROMPT recebe duas ontologias como entrada e ajuda o usuário a criar uma nova ontologia como saída, mesclando as ontologias de entrada.

PROMPT guia o usuário durante o processo de mesclagem das ontologias. Propõe sugestões, determina conflitos e descreve estratégias para resolve-los. Ainda, utiliza técnicas de similaridade entre palavras a fim de encontrar pistas baseadas na estrutura da ontologia e por meio da interação com o usuário (NOY, 2004).

3.2 Integração de Bancos de Dados Baseada em Ontologias

3.2.1 Ontop

-ontop- (RODRíGUEZ-MURO; KONTCHAKOV; ZAKHARYASCHEV, 2013) é uma plataforma para consultar bancos de dados como grafos RDF virtuais utilizando SPARQL. Possui uma coleção de ferramentas que facilitam a execução de tarefas relacionadas ao OBDA.

-ontop- é dividido em duas partes: Quest, seu motor de consultas, e o -ontopPro-, um plug-in para Protégé.

Quest é um mecanismo criado para reescrever consultas SPARQL em SQL. Segundo RODRíGUEZ-MURO; KONTCHAKOV; ZAKHARYASCHEV (2013), suas principais características são:

- Suporte ao SPARQL 1.0;
- Suporte aos RDBMS PostgreSQL, MySQL, H2, DB2, SQL Server, Teiid e Oracle; e
- Suporte para OWLAPI 3, Sesame 2.7 e Protege 4.3.

O plug-in -ontopPro- estende o Protégé ao permitir o mapeamento entre ontologia e banco de dados por meio de um editor visual de mapeamentos. O plug-in também possui uma interface SPARQL integrada com o Quest.

A linguagem de mapeamento utilizada pelo -ontop- é a RDB to RDF mapping language (R2RML) (RODRÍGUEZ-MURO; REZK, 2015). O mapeamento da R2RML é representado por um grafo RDF formado por várias triplas, correspondendo às tabelas, colunas e restrições do banco de dados. Os mapeamentos são especificados na sintaxe Turtle.

3.2.2 ONTOFUSION

ONTOFUSION (PéREZ-REY et al., 2006) é um sistema baseado no OBDA para integrar bancos de dados biomédicos. Seu processo se baseia em duas etapas: mapear e unificar. O mapeamento é semiautomático. A unificação é automática, consistindo em mediar consultas por meio de sua reescrita, utilizando os mapeamentos.

3.2.3 OntoGrate

O OntoGrate (DOU; QIN; LEPENDU, 2010) é um framework para integração automatizada. Seu objetivo é integrar bancos de dados relacionais utilizando tecnologias da web semântica, a fim de automatizar o processo de mapeamento e possibilitar o uso de inferência na reescrita de consultas.

O OntoGrate utiliza técnicas de engenharia reversa de bancos de dados (TRINKUNAS; VASILECAS, 2007; ASTROVA, 2004; LI; DU; WANG, 2005) para representar, automaticamente, o esquema de banco de dados como uma ontologia. Por meio desta representação ontológica do banco de dados, é possível reescrever consultas, de OWL Query Language (OWL-QL) (FIKES; HAYES; HORROCKS, 2004) ou SPARQL para SQL, e traduzir os resultados extraídos do banco de dados para uma linguagem da web semântica, RDF ou OWL.

O processo de mapeamento do OntoGrate utiliza diversas técnicas como similaridade entre palavras (HALL; DOWLING, 1980; ISLAM; INKPEN, 2008), reconciliação de objetos (MITZENMACHER; VARGHESE, 2012; BUSSLER et al., 2004) e mineração de dados multi-relacional (DzEROSKI, 2003; KNOBBE et al., 1999).

3.2.4 Optique

Optique (CALVANESE et al., 2013) é uma plataforma desenvolvida para integrar grandes fontes de dados, também chamada por *Big Data*, por meio de uma abordagem baseada no OBDA. Sua arquitetura possui três camadas:

- **Camada de Apresentação**. Nesta camada o usuário pode criar consultas, visualizar resultados das consultas e gerenciar a plataforma;
- **Camada de Aplicação**. Essa camada é responsável por executar as consultas e mapear a ontologia com as fontes de dados; e
- **Camada de Dados**. Nesta camada, encontram-se os dados que podem ser provenientes de bancos de dados relacionais, semiestruturados e dados temporais.

3.3 Conclusão

Este capítulo apresentou algumas abordagens existentes na área da integração semântica, apontando suas principais caracteristicas. Estas abordagens foram organizadas em duas seções, Seção 3.1 e Seção 3.2.

Na Seção 3.1 foram apresentadas abordagens responsáveis por integrar somente ontologias. Já na Seção 3.2, as abordagens apresentadas realizavam a integração entre bancos de dados, utilizando uma ontologia como esquema global.

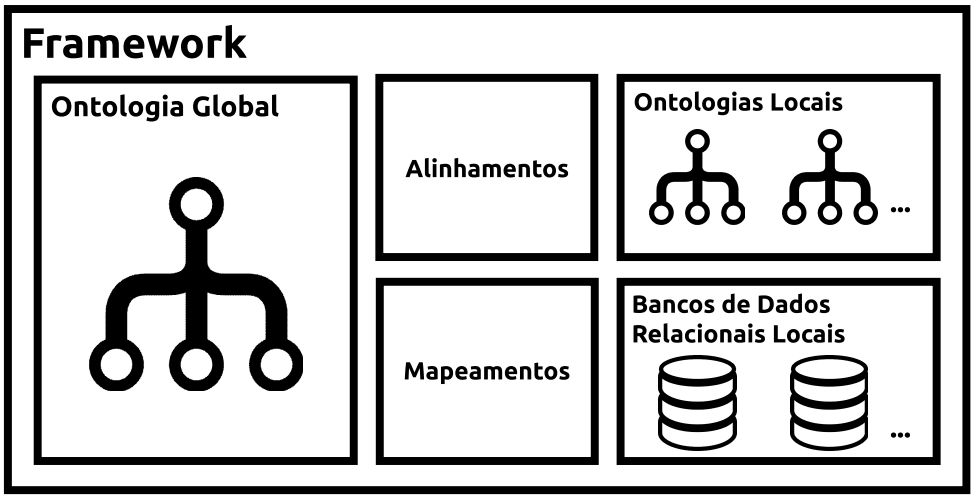
O próximo capítulo irá apresentar a arquitetura proposta neste trabalho. Esta arquitetura possui semelhanças com as abordagens apresentadas neste capítulo, bem como possui características únicas.

4

Arquitetura Proposta

Esta arquitetura propõe a realização de uma integração virtual, mediada por consultas, entre ontologias e bancos de dados relacionais. A Figura 4.1 ilustra quais componentes estão presentes na arquitetura e como eles interagem entre si.

Figura 4.1: Arquitetura proposta. Fonte: O Autor



Fonte: NOY (2004)

Como é possível observar na Figura 4.1, a arquitetura é composta por:

- Um *framework*;
- Uma ontologia global;
- Nenhuma, uma ou mais ontologias locais;
- Nenhum, um ou mais bancos de dados locais;
- Alinhamentos entre a ontologia global e as ontologias locais; e
- Mapeamento entre a ontologia global e os bancos de dados locais.

Baseados no OBDA, OIS e GAV, projetou-se uma arquitetura capaz de realizar uma integração virtual em três diferentes cenários (descritos na Seção 4.3):

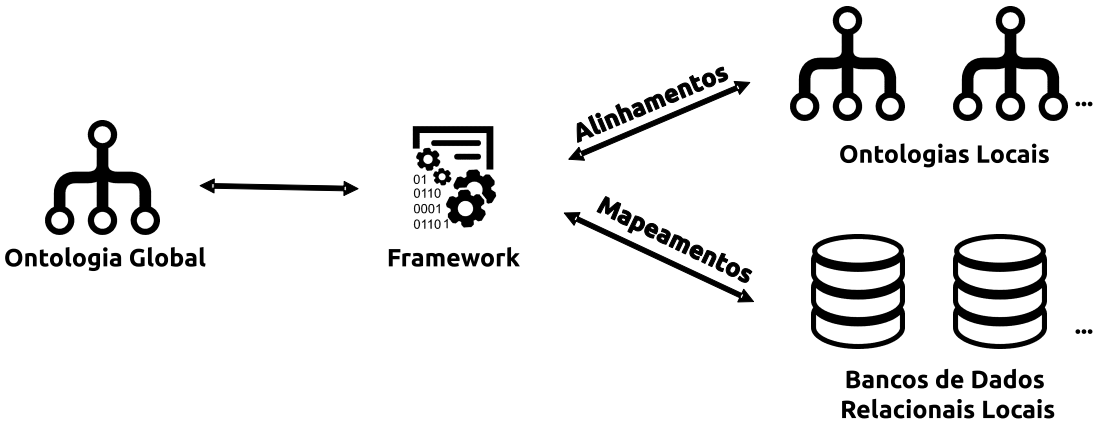
- Quando há apenas ontologias locais;
- Quando há apenas bancos de dados locais; e
- Ou quando há ontologias e bancos de dados locais simultaneamente.

A Seção 4.1 define os componentes da arquitetura e explica o papel de cada um. Na Seção 4.2, o processo de integração da arquitetura é detalhado passo-a-passo. A Seção 4.3 descreve em quais cenários de integração essa arquitetura é eficiente. Por fim, a Seção 4.4 lista os requisitos fundamentais para que uma implementação desta arquitetura seja eficiente.

4.1 Componentes da Arquitetura

Assim como no OBDA e OIS, esta arquitetura pode ser dividida em componentes. Os seis componentes desta arquitetura são: *framework*, ontologia global, alinhamentos, mapeamentos, ontologias locais e bancos de dados locais. Eles estão ilustrados na Figura 4.2 e descritos em seguida.

Figura 4.2: Componentes da arquitetura.



Fonte: O Autor

Nesta arquitetura, o *framework* é responsável por gerenciar o processo de integração. Ele deve realizar a comunicação entre a ontologia global e as fontes locais, alinhar as ontologias, mapear os bancos de dados e mediar consultas. O propósito de incluir o *framework* na arquitetura é incentivar o reuso desta implementação, pois um *framework* deve ser desenvolvido para ser facilmente utilizado em qualquer aplicação.

A ontologia global é utilizada como vocabulário para identificar os componentes da consulta do usuário e traduzí-los em componentes das fontes locais. Ela deve possuir classes do domínio que representam genericamente, ou especificam o conteúdo das fontes locais. Esse

tipo de construção facilita e otimiza o alinhamento e mapeamento das fontes, já que todas são construídas sobre um mesmo domínio, e sobre uma mesma base ontológica.

Como mencionado na Seção 2.2.2.3, os alinhamentos, representam a correspondência semântica entre os componentes (classes, propriedade e instâncias) da ontologia global e das ontologias locais.

Os mapeamentos, como mencionados anteriormente, representam a correspondência semântica entre componentes da ontologia global e os esquemas (tabelas, colunas e relacionamentos) dos bancos de dados.

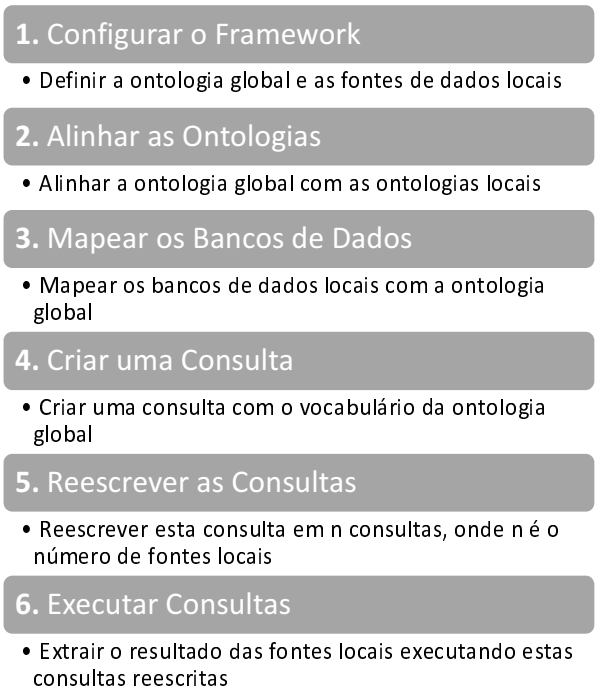
As ontologias locais possuem as instâncias e axiomas que serão recuperados. Elas são responsáveis por apresentar classes e relações, em conformidade com a ontologia global (mesmo domínio).

Por fim, os bancos de dados locais são as fontes de dados das quais os resultados da consulta serão recuperados. É nesta camada onde se encontram os dados que serão recuperados.

4.2 Processo de Integração

Dado os componentes da arquitetura, o próximo passo será explicar como o processo de integração deve ocorrer. Todo o processo pode ser descrito em seis etapas, como visto na Figura 4.3. Estas etapas vão desde a definição das fontes de dados até a obtenção dos resultados das consultas de todas as fontes de dados locais.

Figura 4.3: Processo de integração da arquitetura.



Fonte: O Autor

A Etapa 1 consiste em configurar o *framework*, informando quem é a ontologia global e quem são as ontologias e bancos de dados locais. Esta é a configuração básica para iniciar o processo de integração. Outras configurações opcionais também podem estar presentes no *framework*, como:

- Definição da linguagem de consulta;
- Tipo de algoritmo de alinhamento; e
- Intervalo de confiança desejado para os alinhamentos (*threshold*).

É na Etapa 2 onde o alinhamento entre a ontologia global e as ontologias locais ocorre. Existem diversos algoritmos e implementações que realizam esta tarefa. A arquitetura não se limita a nenhum tipo de solução. Pelo contrário, recomenda o uso de soluções que permitam gerar alinhamentos dentro do intervalo de confiança estipulado pelo usuário.

Na Etapa 3 os bancos de dados locais são mapeados com a ontologia global. Assim como no alinhamento, existem diversos algoritmos e implementações que realizam esta tarefa.

É na Etapa 4 onde a consulta é criada. Ela deve ser criada utilizando o vocabulário da ontologia global. Por se tratar de uma ontologia, essa consulta pode ser feita por meio de várias linguagens, como SPARQL, SPARQL Protocol and RDF Query Language - Description Logics (SPARQL-DL) ou OWL-QL.

Na Etapa 5, a consulta criada utilizando como vocabulário a ontologia global, deve ser reescrita para o vocabulário correspondente das fontes de dados locais. Isto é possível pelo uso dos alinhamentos (Etapa 2) e mapeamentos (Etapa 3), pois estes são necessários para realizar essa etapa. Ao fazer essa transcrição, haverá a garantia de que a consulta criada será aplicada aos dados locais com o vocabulário que os representa. A respeito da linguagem de consulta, para as ontologias locais recomenda-se usar a mesma linguagem na qual a consulta original foi feita, como o SPARQL, já para os bancos de dados locais, as consultas devem ser reescritas para uma linguagem compatível, como SQL.

Na Etapa 6, as consultas reescritas (da etapa anterior) são executadas em suas respectivas fontes de dados. As consultas podem ser executadas em paralelo ou uma após a outra (em fila). O resultado de cada consulta pode ser salvo separadamente ou todos os resultados podem ser mesclados gerando um resultado unificado. Os resultados podem ser salvos em diferentes formatos, como RDF, XML, JSON ou Comma-Separated Values (CSV).

4.3 Cenários de Integração

Integrações podem requerer configurações distintas umas das outras como, por exemplo, suporte a diferentes RDBMS, suporte a diferentes linguagens de consultas, suporte à integração de ontologias e bancos de dados simultaneamente ou apenas um deles. Pelo fato desta arquitetura

não se limitar a linguagens e tecnologias, cabe à implementação dela oferecer um conjunto de funcionalidades que possibilitem a integração nestes e em outros cenários.

Devido à forma como essa arquitetura foi projetada, há três possíveis cenários de integração pela qual, por si só, a mesma pode atuar. Estes cenários são descritos nas subseções 4.3.1, 4.3.2 e 4.3.3.

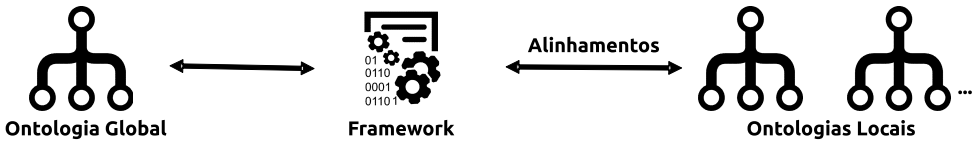
4.3.1 Cenário 1

No primeiro cenário, representado na Figura 4.4, ocorre uma integração semântica apenas com ontologias locais, mediadas por uma ontologia global.

A arquitetura proposta da suporte a este cenário, sendo necessário apenas modificar as configurações do processo de integração, descrito na Figura 4.5. A etapa de mapeamento dos bancos de dados locais foi removida, pois não há banco de dados para serem mapeados.

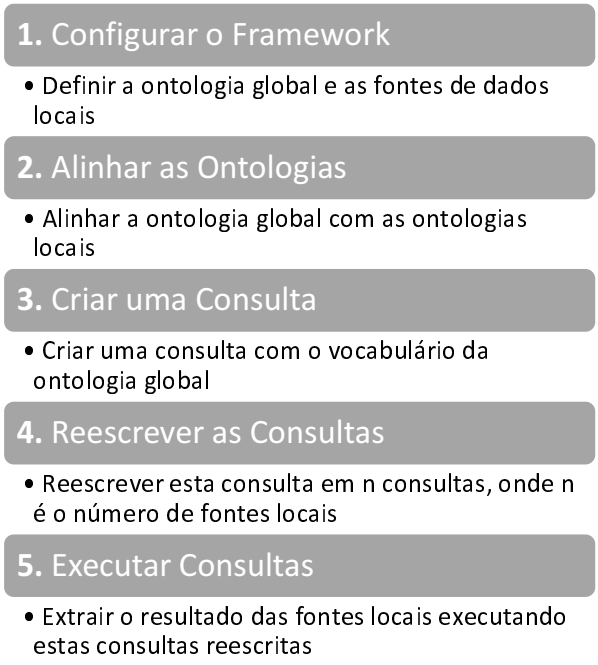
Este cenário pode ser aplicado para recuperar instâncias de ontologias.

Figura 4.4: Cenário de integração onde só há ontologias locais.



Fonte: O Autor

Figura 4.5: Processo de integração quando só há ontologias locais.



Fonte: O Autor

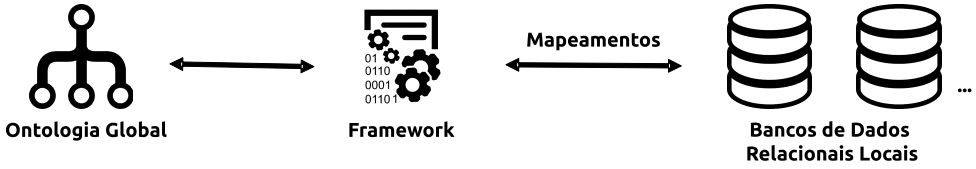
4.3.2 Cenário 2

No segundo cenário, representado na Figura 4.6, ocorre uma integração apenas com bancos de dados locais, mediados por uma ontologia global.

A arquitetura proposta também da suporte a este cenário, sendo necessário apenas modificar as configurações do processo de integração, descrito na Figura 4.7. A etapa de alinhamento das ontologias locais é removida, pois não há ontologias locais para alinhar com a ontologia global.

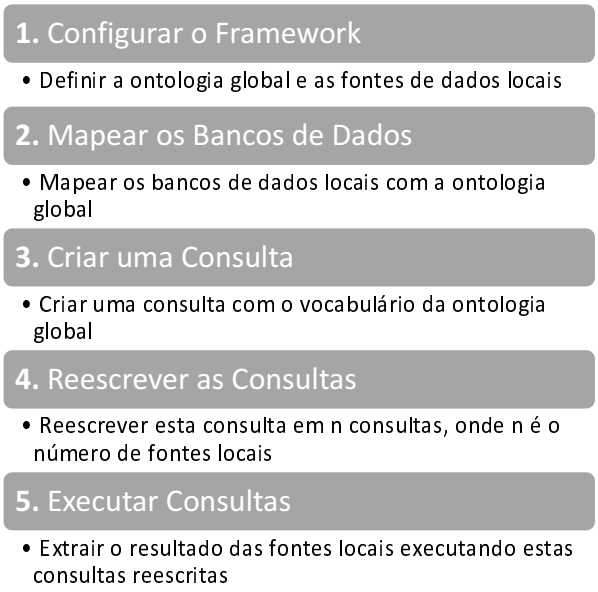
Este cenário pode ser aplicado para recuperar dados das tabelas de bancos de dados.

Figura 4.6: Cenário de integração onde só há bancos de dados locais.



Fonte: O Autor

Figura 4.7: Processo de integração quando só há bancos de dados locais.



Fonte: O Autor

4.3.3 Cenário 3

O terceiro cenário já foi descrito na Figura 4.2. Nele ocorre uma integração com ontologias e bancos de dados locais, também mediados por uma ontologia global.

A arquitetura proposta foi projetada para este tipo de cenário. Não é necessário ajustar o processo de integração descrito na Figura 4.3.

Este cenário pode ser aplicado para recuperar, simultaneamente, instâncias de ontologias e dados de bancos de dados.

4.4 Requisitos

Uma implementação desta arquitetura deve possuir um conjunto de funcionalidades que permitam a execução satisfatória, *i.e.*, funcionalidades as quais admitam a execução dos passos da integração sem erros de execução, além de outras características. A seguir são listados os requisitos que devem ser levados em consideração ao desenvolver uma implementação da arquitetura proposta.

Completo. Precisa realizar todas as etapas descritas no processo de integração, *i.e.*, alinhar ontologias, mapear bancos de dados, reescrever consultas, entre outros. Precisa também ser capaz de integrar ontologias e bancos de dados com diferentes níveis de complexidade e granularidade.

Fácil de usar. Deve ser possível utilizá-lo em diferentes aplicações com diferentes propósitos. Para isso, a implementação deve contar com documentação extensiva, facilidade de configuração e integração com outras soluções.

Confiável. Os resultados precisam ser precisos e a resposta rápida. Os algoritmos para alinhamento, mapeamento e reescrita de consultas precisam extrair informações de qualidade das fontes de dados locais.

Escalável. Deve ser capaz de manipular grandes fontes de dados com o mínimo de impacto no desempenho.

4.5 Conclusão

Este capitulo apresentou a arquitetura desenvolvida neste trabalho. Foram descritos os componentes desta arquitetura, seu processo de integração e os cenários de integração para qual ela foi construida. Além disso, foram descritos os requisitos necessários para uma implementação bem-sucedida.

No próximo capítulo será apresentado o Gryphon Framework, a implmentação da arquitetura proposta. Neste framework foi implementado todo o processo de integração descrito neste capítulo, assim como todos os requisitos sugerigos pela arquitetura foram levados em consideração.

5

Gryphon Framework

Como prova de conceito, foi desenvolvido o Gryphon Framework. O Gryphon é um *framework* que tem como objetivo implementar a arquitetura proposta no Capítulo 4. Um *framework* pode ser definido como um conjunto extensível de classes, interfaces e métodos dedicados a resolver problemas de um domínio específico. Em outras palavras, um *framework* provê funcionalidades que devem ser integradas à aplicação.

O Gryphon é um *framework* de código aberto que disponibiliza classes, interfaces e métodos que facilitam a realização de tarefas de integração semântica. Como aplicação, o Gryphon é distribuído sob a licença MIT[1], desenvolvido com Java 8[2] e Eclipse Mars[3]. Seu código fonte está disponível no GitHub[4].

A licença MIT, criada pelo *Massachusetts Institute of Technology*[5], foi utilizada por ser uma licença permissiva. Ou seja, não existe proibição sobre o que pode ser feito com o *software* e seu código, desde que o autor seja referenciado.

Java é uma linguagem de programação de código aberto orientada a objetos, fortemente tipada (a declaração do tipo é obrigatória), multiplataforma (funciona em diversos sistemas operacionais como Windows, MacOS e Linux) e concorrente (execução simultânea de várias tarefas). O Java 8 foi escolhido para desenvolver o Gryphon Framework por ser uma linguagem multiplataforma e por ser utilizada em outros projetos relacionados ao tema, *e.g.*, soluções de alinhamento, mapeamento e reescrita de consultas.

Eclipse é um Integrated Development Environment (IDE) criado em Java e de código aberto. O Eclipse possibilita o desenvolvimento de aplicações para plataformas *desktop*, *web* e *mobile*. Ele dá suporte a diversas linguagens, dentre as quais o Java, e pode ter suas funcionalidades estendidas por meio da instalação de plug-ins. A versão do Eclipse utilizada para o desenvolvimento foi o Eclipse Mars. Ele foi escolhido por ser um IDE consolidado e por possuir uma suíte completa e estendível com diversas funcionalidades que facilitam o desenvolvimento.

[1] http://opensource.org/licenses/MIT
[2] http://oracle.com/java
[3] http://eclipse.org
[4] http://github.com/adrielcafe/GryphonFramework
[5] http://web.mit.edu

Na Seção 5.1, são listados os componentes utilizados no desenvolvimento do Gryphon Framework. Em seguida, a Seção 5.2 descreve como ele foi implementado. Por fim, a Seção 5.3 descreve como o processo de integração da arquitetura foi implementado no *framework*.

5.1 Componentes Utilizados

Como o Gryphon Framework é uma solução utilizada para intermediar e facilitar o processo de integração, outras soluções foram reutilizadas para a execução de subtarefas, *i.e.*, alinhamento de ontologias, mapeamento de bancos de dados, reescrita de consultas e execução de consultas. Um melhor entendimento sobre este *framework* pode ser alcançado se considerado como um ecossistema formado por soluções de código aberto, implementando a arquitetura proposta neste trabalho. As soluções utilizadas para implementar o Gryphon Framework são descritas nas subseções 5.1.1, 5.1.2, 5.1.3 e 5.1.4.

5.1.1 Sesame

Sesame (BROEKSTRA; KAMPMAN; HARMELEN, 2002) é um *framework* Java de código aberto que facilita a construção de aplicações para web semântica. Com ele é possível, entre outros:

- Manipular grafos em RDF e ontologias em OWL;
- Realizar raciocínio em ontologias OWL;
- Executar consultas SPARQL; e
- Criar endpoints SPARQL;

Sesame foi escolhido por ser um *framework* com recursos que facilitam o desenvolvimento de novas soluções para web semântica. No Gryphon Framework, Sesame foi utilizado para executar consultas SPARQL nas ontologias locais.

5.1.2 D2RQ

D2RQ (BIZER, 2004) é uma plataforma de código aberto que permite acessar bancos de dados relacionais por meio de consultas SPARQL. Constitui-se por três componentes:

- Um servidor Hypertext Transfer Protocol (HTTP);
- Uma linguagem de mapeamento; e
- Um motor para reescrita de consultas.

O servidor HTTP é responsável por prover um endpoint SPARQL para realizar consultas em bancos de dados relacionais. A linguagem de mapeamento é responsável por descrever a relação entre a estrutura do banco de dados com os componentes de uma ontologia. O motor de reescrita de consultas utiliza estes mapeamentos para reescrever consultas SPARQL em SQL.

No Gryphon Framework, D2RQ foi utilizado para mapear a ontologia global com os bancos de dados locais e para reescrever consultas SPARQL em SQL, possibilitando a consulta em bancos de dados relacionais.

5.1.3 AML

O AgreementMakerLight (AML) (CRUZ; ANTONELLI; STROE, 2009) é um *framework* de código aberto, (semi) automático e escalável. Ele é responsável por alinhar ontologias e foi desenvolvido primariamente para o domínio biomédico. Uma das vantagens em utilizar o AML é a capacidade de manipular grandes ontologias, incluindo um algoritmo para reparo de alinhamentos e um *background knowledge* composto por WordNet[6], DOID[7], MeSH[8], Uberon[9] e UMLS[10].

No Gryphon Framework, o AML foi utilizado para alinhar a ontologia global com as ontologias locais.

5.1.4 Mediation

Mediation (CORRENDO et al., 2010) é uma Application Programming Interface (API) de código aberto responsável por reescrever consultas SPARQL por meio de alinhamentos descritos no formato EDOAL (DAVID et al., 2011). O mesmo utiliza um algoritmo de reescrita baseado no BGP (KOLLIA; GLIMM; HORROCKS, 2011) (conjunto de triplas formado por sujeito, predicado e objeto).

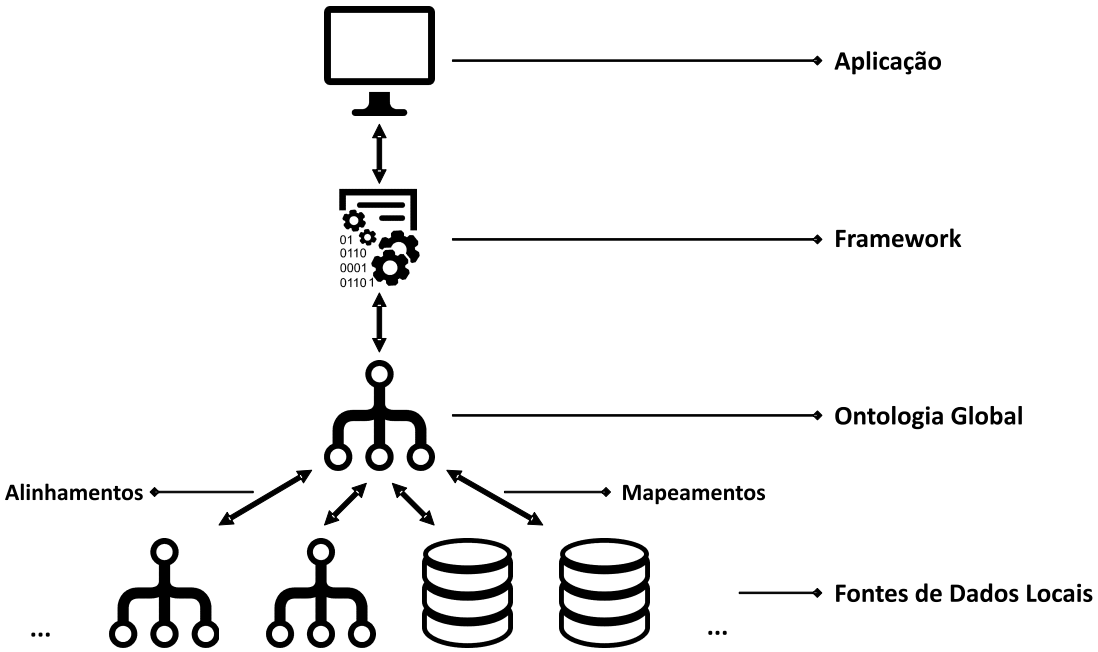
No Gryphon Framework, Mediation foi utilizado para reescrever consultas SPARQL com base nos alinhamentos gerados pelo AML.

5.2 Gryphon Framework

O Gryphon Framework foi desenvolvido para ser rápido, fácil de usar e de integrar com soluções já existentes. O mesmo cumpriu todos os requisitos descritos na Seção 4.4. Para a utilização do Gryphon, este precisa ser inserido em uma aplicação. Feito isso, este será responsável por gerenciar e efetuar todo o processo de integração. A Figura 5.1 mostra como a interação entre aplicação, *framework* e fontes de dados são relacionados.

[6] http://wordnet.princeton.edu
[7] http://disease-ontology.org
[8] http://nlm.nih.gov/mesh
[9] http://uberon.github.io
[10] http://nlm.nih.gov/research/umls

Figura 5.1: Fluxo do Gryphon Framework.

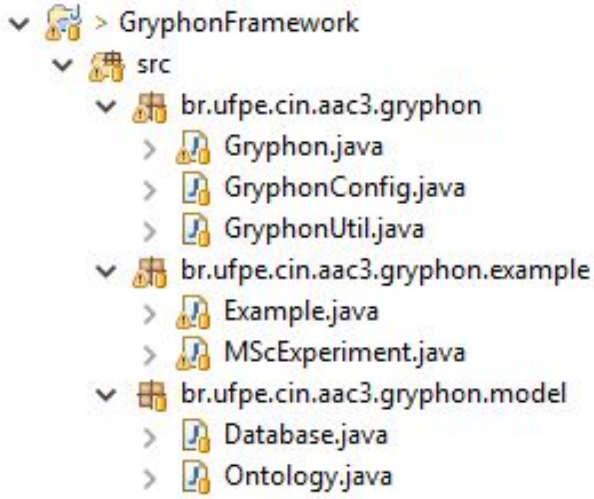


Fonte: O Autor

A aplicação utiliza os métodos do Gryphon Framework, como *setGlobalOntology*(), *addLocalOntology*(), *addLocalDatabase*(), *alignAndMap*() e *query*(), para realizar a integração semântica. Internamente, esses métodos se encarregam de executar todas as etapas do processo de integração descritos na Seção 4.2, como alinhar as ontologias, mapear os bancos de dados e reescrever consultas.

A Figura 5.2 mostra como o projeto foi estruturado. Por ser desenvolvido em Java, o projeto está organizado em *packages* (mecanismo para organizar as classes e interfaces). A composição de cada *package* é descrita em seguida.

Figura 5.2: Estrutura do projeto.



Fonte: O Autor

O *package br.ufpe.cin.aac3.gryphon* contém as principais classes do projeto:

- **Gryphon.java** - Esta classe é responsável por realizar todo o processo de integração, i.e., definir a ontologia global e as fontes locais e realizar consultas;
- **GryphonConfig.java** - Classe que contém as opções de configuração do framework, i.e., informar o diretório onde os alinhamentos, mapeamentos e resultados serão salvos; e
- **GryphonUtil.java** - Classe que contém métodos utilitários, i.e., exibir mensagens de log no console e manipular InputStreams.

O *package br.ufpe.cin.aac3.gryphon.model* contém classes que representam ontologias e bancos de dados:

- **Ontology.java** - Classe utilizada para gerenciar a comunicação entre o framework e a ontologia; e
- **Database.java** - utilizada para gerenciar a comunicação entre o framework e o banco de dados relacional.

O *package br.ufpe.cin.aac3.gryphon.model.example* contém exemplos de uso do Gryphon Framework:

- **MScExperiment.java** - Classe utilizada nos experimentos deste trabalho; e
- **Example.java** - Classe com algumas integrações de demonstração.

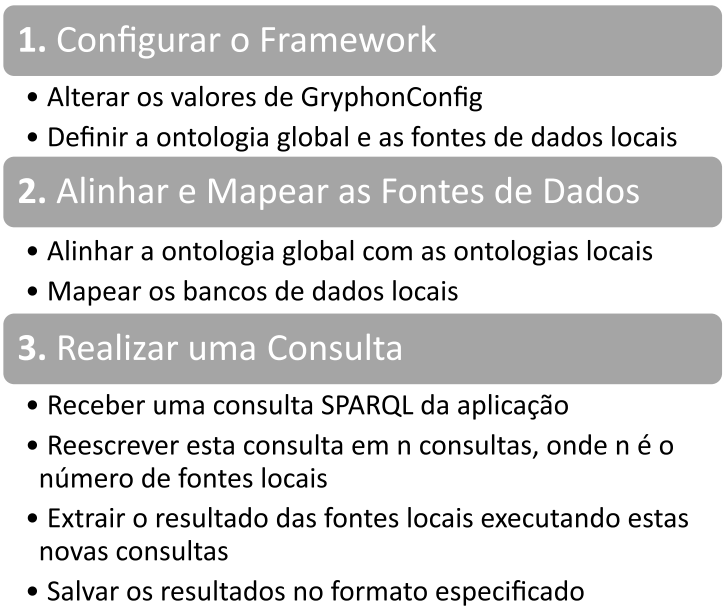
O Gryphon Framework dá suporte a ontologias nos formatos OWL e RDF. Os RDBMS suportados são o MySQL[11] e PostgreSQL[12]. Como linguagem de consulta, foi adotada SPARQL. Os resultados podem ser salvos em JSON, XML e CSV.

5.3 Processo de Integração Otimizado

Na Seção 4.2, foi descrito o processo de integração da arquitetura, composto por seis etapas. O Gryphon Framework implementa todas estas etapas. Entretanto, para tornar o processo de integração o mais transparente possível, algumas etapas foram abstraídas. Neste processo otimizado, apenas três etapas são necessárias para realizar uma integração mediada por consulta no Gryphon Framework. Estas etapas são descritas na Figura 5.3 e serão detalhadas a seguir.

[11] http://mysql.com
[12] http://postgresql.org

Figura 5.3: Etapas para realizar uma integração com o Gryphon Framework.



Fonte: O Autor

5.3.1 Etapa 1

A primeira etapa consiste em configurar o *framework*. Ela é responsável por executar a primeira etapa (Configurar o Framework) do processo de integração da arquitetura (Figura 4.3). Na classe *GryphonConfig*, existem algumas opções que podem ser configuradas, como por exemplo:

- Pasta onde serão salvos os alinhamentos, mapeamentos e resultados das consultas; e
- Exibir *logs* no console.

Nesta etapa, o usuário também deve informar a ontologia global e as ontologias e/ou bancos de dados locais, para isso deve:

- Instanciar as classes *Ontology* e *Database*; e
- Usar os métodos *Gryphon.setGlobalOntology*(), *Gryphon.addLocalOntology*() e *Gryphon.addLocalDatabase*() para informar, respectivamente, a ontologia global, as ontologias locais e os bancos de dados locais.

O Código-fonte 5.1 exemplifica o código-fonte da primeira etapa.

Listing 5.1: Exemplo de código da primeira etapa.

```
GryphonConfig.setWorkingDirectory(new File("myIntegration"));
GryphonConfig.setLogEnabled(true);
GryphonConfig.setShowLogo(true);
```

```
Gryphon.init();

Ontology globalOnt = new Ontology("globalOntology",
        uriToGlobalOntology);
Ontology localOnt1 = new Ontology("localOntology1",
        uriToLocalOntology1);
Ontology localOnt2 = new Ontology("localOntology2",
        uriToLocalOntology2);
Database localDB1 = new Database("localhost", 3306,
        "username", "password", "db1", Gryphon.DBMS.MySQL);
Database localDB2 = new Database("localhost", 3306,
        "username", "password", "db2", Gryphon.DBMS.PostgreSQL);

Gryphon.setGlobalOntology(globalOnt);
Gryphon.addLocalOntology(localOnt1);
Gryphon.addLocalOntology(localOnt2);
Gryphon.addLocalDatabase(localDB1);
Gryphon.addLocalDatabase(localDB2);
```

Fonte: O Autor

5.3.2 Etapa 2

Na segunda etapa, as ontologias são alinhadas e os bancos de dados são mapeados. Ela equivale à segunda (Alinhar as Ontologias) e terceira (Mapear os Bancos de Dados) etapas do processo de integração da arquitetura (Figura 4.3). Para realizar esta etapa, basta executar o método *Gryphon.alignAndMap*(). Toda a complexidade presente nessas duas etapas de integração da arquitetura foram abstraídas em um único método, facilitando o uso do Gryphon Framework por usuários com o conhecimento limitado sobre integração semântica e programação.

O Código-fonte 5.2 exemplifica a segunda etapa.

Listing 5.2: Exemplo de código da segunda etapa.

```
Gryphon.alignAndMap();
```

Fonte: O Autor

5.3.3 Etapa 3

A terceira e última etapa é responsável por realizar as consultas. Ela equivale à quarta (Criar Consulta), quinta (Reescrever Consulta) e sexta (Executar Consultas) etapas do processo de integração da arquitetura (Figura 4.3). Para realizar esta etapa é necessário:

- Criar uma consulta em SPARQL, utilizando o vocabulário da ontologia global; e
- Executar a consulta com o método *Gryphon.query*() e escolher o formato que os resultados serão salvos.

Novamente, toda a complexidade presente nas etapas quarta, quinta e sexta do processo de integração da arquitetura foram abstraídas em um único método, o *Gryphon.query*(). Neste, a consulta é reescrita em SPARQL para as ontologias locais, e em SQL para os bancos de dados locais.

O Código-fonte 5.3 exemplifica o código-fonte da terceira etapa.

Listing 5.3: Exemplo de código da terceira etapa.

```
String strQuery =
     "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> "
    +"SELECT ?x ?y "
    +"WHERE { ?x rdf:type ?y }";
Gryphon.query(strQuery, ResultFormat.JSON);
```

Fonte: O Autor

Os resultados das consultas podem ser salvos nos formatos JSON, XML e CSV.

5.4 Conclusão

Neste capítulo foi apresentado o Gryphon Framework, solução que implementa a arquitetura proposta. Seu funcionamento e forma de usar foram explicados detalhadamente. Também foram descritas as soluções existentes que foram utilizadas no desenvolvimento do framework.

No próximo capítulo serão descritos dois experimentos realizados com o Gryphon Framework. Estes experimentos desmonstram como o framework podem ser utilizados em diferentes cenários de integração e com fontes de dados de diferentes domínios.

6

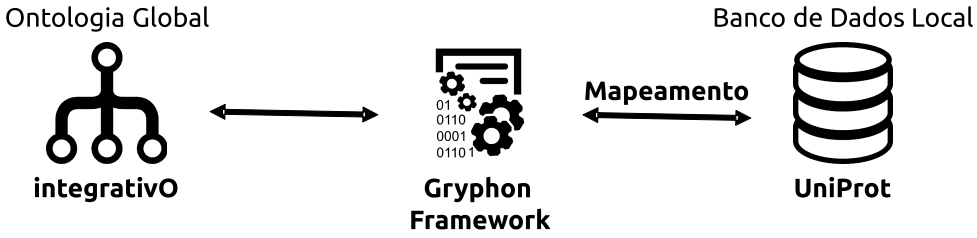
Experimentos

6.1 Experimento 1

Este experimento consiste em integrar bases de dados biológicos, utilizando o Gryphon Framework. Este cenário de integração é formado por uma ontologia global, um banco de dados local, e o mapeamento entre eles (Figura 6.1), em que:

- A ontologia global será a integrativO (descrita na Seção 6.1.1);
- O banco de dados local será o UniProt (descrito na Seção 6.1.2).

Figura 6.1: Cenário de integração do Experimento 1.



Fonte: O Autor

O objetivo desta integração é possibilitar a extração de conteúdo proveniente do UniProt, utilizando o vocabulário das ontologias que formam a integrativO. O resultado desta integração deve responder consultas do domínio biológico como *"Quais processos biológicos são promovidos por proteínas?"*. Na Seção 6.1.3, são descritas as consultas realizadas neste experimento e seus respectivos resultados.

Este experimento faz parte da tese do aluno de Doutorado do Centro de Informática Filipe Santana da Silva, sob título "Aperfeiçoando o processo de criação de ontologias em Lógica de Descrições através de enriquecimento axiomático automatizado" (título provisório), em fase de conclusão.

A classe Java, implementada utilizando o Gryphon, para o experimento encontra-se no Apêndice A. O mapeamento entre o integrativO e o UniProt encontra-se disponível no Apêndice B.

6.1.1 IntegrativO

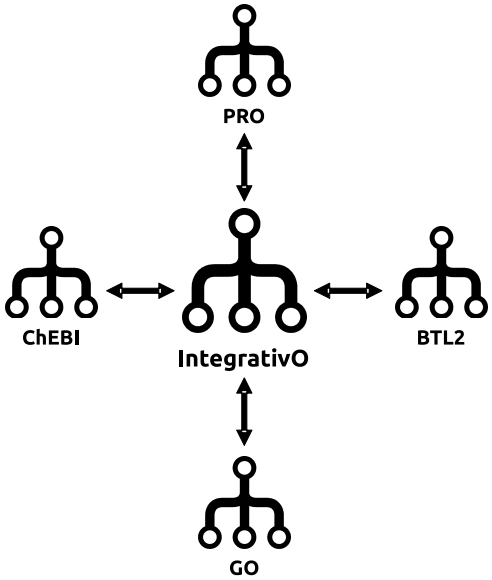
IntegrativO[1] é um projeto que tem como objetivo permitir a comunicação entre diversas ontologias do domínio biológico tomando como princípio a BioTopLite2 (BTL2). Efetivamente, o que é chamado de ontologia integrativO nada mais é que um arquivo que importa módulos representativos das ontologias utilizadas para representar o conteúdo ontológico das bases de dados do experimento.

A integrativO segue o princípio da modularização de ontologias, caracterizado por GRAU et al. (2008); JIMéNEZ-RUIZ et al. (2008); PARSIA; SATTLER; SCHNEIDER (2009). Esta técnica pode ser descrita como a decomposição de ontologias potencialmente grandes em conjuntos menores (módulos) interligados e que representam uma fração completa de uma ontologia inteira. Neste trabalho, o intuito primordial da criação dos módulos é disponibilizar para os testes da integração apenas as classes anotadas no banco de dado utilizado.

Para criar módulos de ontologias, é necessário seguir uma metodologia que, além de garantir que o módulo seja completo, mantenha características de identificação e estrutura originais. Em outras palavras, que não interfira na estratégia de representação da ontologia original. Para isso, a integrativO utiliza a metodologia de reuso de ontologias Minimum Information to Reference External Ontology Terms (MIREOT) (COURTOT et al., 2009).

Nesse sentido, integrativO é constituída a partir de módulos da Gene Ontology (GO), PRotein Ontology (PR) e Chemical Entities of Biological Interest (ChEBI). Estas ontologias, por sua vez, são organizadas seguindo a descrição formal de classes e relações do domínio biológico disponibilizada pela BTL2 (SCHULZ; BOEKER, 2013). A Figura 6.2 ilustra como a IntegrativO é organizada.

Figura 6.2: Módulos do IntegrativO.



Fonte: O Autor

[1] https://github.com/integrativo/ontos/blob/master/Sources/integrativO.owl

Uma breve descrição das ontologias que compõe a integrativO é feita nas subseções 6.1.1.1, 6.1.1.2, 6.1.1.3, 6.1.1.4 e 6.1.2.

6.1.1.1 GO

A GO (ASHBURNER et al., 2000) é uma ontologia criada para possibilitar a estruturação de um vocabulário controlado, definido e comum sobre os papéis dos genes e dos produtos gênicos de diversos organismos.

A partir do desenvolvimento da GO, estudos foram realizados, como o de BIEN et al. (2012) em que utilizaram essa ontologia para avaliar similaridade semântica entre ela e dados de sequências de proteínas, expressão gênica, interações entre proteínas, entre outras. O estudo de BURGE et al. (2012) descreveu a utilização de anotações sobre proteínas, juntamente com a GO, para inferir informações sobre sequências de proteínas não caracterizadas. Outro trabalho, desenvolvido por CHITTENDEN et al. (2012) evidenciou o desenvolvimento de um algoritmo para descobrir a subclassificação para processos biológicos relacionados aos termos disponíveis na GO.

6.1.1.2 ChEBI

A ChEBI (DEGTYARENKO et al., 2008; HASTINGS et al., 2013) foi derivada de um projeto desenvolvido pelo European Bioinformatics Institute (EBI), em 2002. O projeto tinha como objetivo criar um dicionário definitivo e gratuito de entidades químicas de interesse biológico. O trabalho foi motivado pela ausência de fontes anotadas e de alta qualidade para promover a utilização correta de termos bioquímicos em fontes de dados biológicas.

Nesse sentido, a ChEBI inclui a descrição de entidades químicas de baixo peso molecular para auxiliar no entendimento e análise do funcionamento biológico. Ela é subdividida em partes principais que tratam da representação de estruturas moleculares, papéis biológicos e partículas subatômicas. Estrutura molecular (‘$Molecularstructure'$) é relacionada à representação de moléculas e às respectivas subpartes. Papéis biológicos (‘$Biologicalrole'$) são utilizados para classificar moléculas dependendo no papel apresentado durante a realização de processos biológicos diversos. E, finalmente, partículas subatômicas (‘$Subatomicparticle'$) para a representação de entidades menores que um átomo.

6.1.1.3 PR

A PR (NATALE et al., 2014, 2011) é uma ontologia destinada à representação de entidades relacionadas às proteínas. PR é mantida pelo Protein Information Resource (PIR), uma fonte pública de dados e informações sobre proteínas. O esforço para a criação da PR resultou na integração de diversas bases de dados e a criação da estrutura atual do UniProt como uma base de dados. A PR apresenta classes para descrever formas modificadas, isoformas e

complexos proteicos de diversos organismos vivos. A estrutura da PR é subdividida em três compartimentos:

- **ProEvo**, com a descrição de relacionamentos evolucionários entre proteínas;
- **ProForm**, para proteínas geradas de acordo com locais genéticos específicos; e
- **ProComp**, com a descrição de complexos moleculares que incluem proteínas.

Do ponto de vista ontológico, PR inclui todas as classes de proteínas naturalmente ocorrentes, e axiomas que as descrevem. PR é utilizada, por exemplo, para representar a tradução de produtos de genes específicos, como proteínas e RNA. Como fundamentação, a PR reutiliza a Sequence Ontology (SO) (EILBECK et al., 2005) e a Proteomics Standards Initiative Modification Ontology (PSI-MOD) (MONTECCHI-PALAZZI et al., 2008). Em suma, na PR, proteínas são descritas como o resultado da tradução de sequências e que apresenta como partes moléculas de aminoácidos.

6.1.1.4 BTL2

BTL2 (SCHULZ; BOEKER, 2013) é uma versão menos complexa e redesenhada da BioTop (BEISSWANGER et al., 2008). A BioTop foi criada em 2006 como uma ontologia supra-domínio, *i.e.*, com um nível de abstração maior sobre o domínio biológico. Na versão reduzida, a BTL2 fornece classes e propriedades de objeto ricamente axiomatizadas e que permite a representação de domínios da biologia, como a genética ou a bioquímica. Com a estrutura mais genérica oferecida pela BTL2, por exemplo, é possível alinhar formalmente a GO, ChEBI e a PR.

A partir da BTL2 é possível criar e alinhar outras ontologias do âmbito biológico em uma abordagem *middle-out*, *i.e.*, a partir das classes mais específicas da BTL2 acopladas às classes mais genéricas das ontologias de domínio. Outra vantagem em utilizar a BTL2 é a presença de pontes e reuso de propriedades de objeto da Basic Formal Ontology (BFO) (SPEAR, 2006) e Relation Ontology (RO) (SMITH et al., 2005), respectivamente. Ontologias biológicas as quais reutilizam essas duas ontologias podem ser facilmente integradas sob o *framework* formal da BTL2.

6.1.2 UniProt

O Universal Protein Resource (UniProt) (CONSORTIUM, 2008) é um banco de dados com curadoria realizada por profissionais do domínio biológico. O UniProt é um ponto de acesso central para informações sobre proteínas. Este banco de dados fornece uma central de recursos estáveis, compreensíveis e totalmente gratuitos, sobre sequências de proteína e anotações sobre características funcionais, provenientes da GO e PR.

O UniProt contém registros com informações extraídas da literatura e por meio da análise computacional (CONSORTIUM, 2008). Para garantir a qualidade desejada, estas anotações são analisadas uma a uma e incluídas na subdivisão denominada SwissProt. Os registros que ainda não foram processados por especialistas também são disponibilizados, mas em outro banco chamado TrEMBL.

As informações presentes no UniProt consistem em:

- Funções;
- Informações específicas das enzimas;
- Domínios e sites biologicamente relevantes;
- Modificações pós-traducionais;
- Localizações subcelulares;
- Especificidade do tecido;
- Estruturas;
- Interações; e
- Doenças associadas a deficiências ou anormalidades.

dentre outros.

Somente uma parte do UniProt foi utilizada neste experimento. O banco de dados utilizado possui 650.864 registros, ele contém dados que representam processos biológicos, componentes celulares, funções moleculares e nomes de proteínas, genes e organismos.

6.1.3 Resultados

Para este experimento, um biólogo criou 4 consultas em Português que foram traduzidas para SPARQL. As consultas são descritas a seguir juntamente com seus resultados.

6.1.3.1 Consulta 1

Recuperar os organismos que incluem homocysteine

Listing 6.1: Consulta 1 em SPARQL.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX btl2: <http://purl.org/biotop/btl2.owl#>

SELECT DISTINCT ?organism
WHERE {
```

```
    ?organismId a btl2:organism ;
        rdfs:label ?organism .
    ?homocysteineId a btl2:MonoMolecularEntity .
    ?organismId btl2:includes ?homocysteineId .
}
```

Fonte: O Autor

Listing 6.2: Consulta 1 reescrita em SQL.

```
SELECT DISTINCT `T2_organism`.`organism`,
                `T1_organism`.`id`,
                `T3_molecule`.`id`
FROM   `molecule` AS `T4_molecule`,
       `organism` AS `T1_organism`,
       `organism` AS `T4_organism`,
       `organism` AS `T2_organism`,
       `molecule` AS `T3_molecule`
WHERE  ( `T1_organism`.`id` = `T2_organism`.`id`
         AND `T2_organism`.`id` = `T4_organism`.`id`
         AND `T2_organism`.`organism` IS NOT NULL
         AND `T3_molecule`.`id` = `T4_molecule`.`id`
         AND `T4_molecule`.`id` = `T4_organism`.`id` )
```

Fonte: O Autor

O objetivo da Consulta 1 foi de recuperar todos os organismos que tem incluídos em si uma partícula do aminoácido homocisteína. Esta consulta é importante por evidenciar quais organismos apresentam alguma capacidade de processar este aminoácido. Ela demorou 19 segundos para ser concluída. No total foram recuperados 1.532 registros. Os 10 primeiros resultados são descritos no Quadro 6.1.

Quadro 6.1: Os 10 primeiros resultados da Consulta 1.

?organism
Bacillus subtilis (strain 168)
Mus musculus (Mouse)
Homo sapiens (Human)
Arabidopsis thaliana (Mouse-ear cress)
Saccharomyces cerevisiae (strain ATCC 204508 / S288c) (Baker's yeast)
Escherichia coli (strain K12)
Methanothermobacter marburgensis (Methanobacterium thermoautotrophicum)
Rattus norvegicus (Rat)
Oryctolagus cuniculus (Rabbit)
Coptis japonica (Japanese goldthread)

Fonte: O Autor

6.1.3.2 Consulta 2

Recuperar os processos biológicos que estão inclusos em organismos

Listing 6.3: Consulta 2 em SPARQL.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX btl2: <http://purl.org/biotop/btl2.owl#>
PREFIX go: <http://purl.obolibrary.org/obo/go.owl#>

SELECT DISTINCT ?biologicalProcess ?organism
WHERE {
    ?biologicalProcessId a go:biological_process ;
        rdfs:label ?biologicalProcess .
    ?organismId a btl2:organism ;
        rdfs:label ?organism .
    ?biologicalProcessId btl2:isIncludedIn ?organismId .
}
```

Fonte: O Autor

Listing 6.4: Consulta 2 reescrita em SQL.

```
SELECT DISTINCT `t2_biological_process`.`biological_process`,
                `t3_organism`.`id`,
                `t4_organism`.`organism`,
                `t1_biological_process`.`id`
FROM `biological_process` AS `t5_biological_process`,
     `organism`           AS `t5_organism`,
```

```
    `biological_process` AS `t2_biological_process`,
    `organism`           AS `t4_organism`,
    `biological_process` AS `t1_biological_process`,
    `organism`           AS `t3_organism`
WHERE (`t1_biological_process`.`id` = `t2_biological_process`.`id`
    AND `t2_biological_process`.`biological_process` IS NOT NULL
    AND `t2_biological_process`.`id` = `t5_biological_process`.`id`
    AND `t3_organism`.`id` = `t5_organism`.`id`
    AND `t4_organism`.`id` = `t5_organism`.`id`
    AND `t4_organism`.`organism` IS NOT NULL
    AND `t5_biological_process`.`id` = `t5_organism`.`id`)
```

Fonte: O Autor

A Consulta 2 tem como objetivo recuperar todos os processos biológicos que organismos apresentam a capacidade de realizá-lo. Ela demorou 18 segundos para ser concluída. No total foram recuperados 15.857 registros. Os 10 primeiros resultados são descritos no Quadro 6.2.

Quadro 6.2: Os 10 primeiros resultados da Consulta 2.

?biologicalProcess	**?organism**
methionine biosynthetic process [GO:0009086]	Bacillus subtilis (strain 168)
tetrahydrofolate interconversion [GO:0035999]	Bacillus subtilis (strain 168)
L-methionine salvage [GO:0071267]	Mus musculus (Mouse)
S-adenosylmethionine metabolic process [GO:0046500]	Mus musculus (Mouse)
S-methylmethionine cycle [GO:0033528]	Mus musculus (Mouse)
S-methylmethionine metabolic process [GO:0033477]	Mus musculus (Mouse)
homocysteine metabolic process [GO:0050667]	Mus musculus (Mouse)
L-methionine biosynthetic process from S-adenosylmethionine [GO:0019284]	Homo sapiens (Human)
S-adenosylhomocysteine metabolic process [GO:0046498]	Homo sapiens (Human)
sulfur amino acid metabolic process [GO:0000096]	Homo sapiens (Human)

Fonte: O Autor

6.1.3.3 Consulta 3

Recuperar os processos biológicos e os componentes celulares onde eles podem ser encontrados

Listing 6.5: Consulta 3 em SPARQL.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX btl2: <http://purl.org/biotop/btl2.owl#>
PREFIX go: <http://purl.obolibrary.org/obo/go.owl#>
```

```
SELECT DISTINCT ?biologicalProcess ?cellularComponent
WHERE {
    ?biologicalProcessId a go:biological_process ;
        rdfs:label ?biologicalProcess .
    ?cellularComponentId a go:cellular_component ;
        rdfs:label ?cellularComponent .
    ?biologicalProcessId btl2:isIncludedIn ?cellularComponentId .
}
```

Fonte: O Autor

Listing 6.6: Consulta 3 reescrita em SQL.

```
SELECT DISTINCT `T3_cellular_component`.`id`,
                `T2_biological_process`.`biological_process`,
                `T4_cellular_component`.`cellular_component`,
                `T1_biological_process`.`id`
FROM    `biological_process` AS `T5_biological_process`,
        `cellular_component` AS `T5_cellular_component`,
        `cellular_component` AS `T4_cellular_component`,
        `biological_process` AS `T2_biological_process`,
        `biological_process` AS `T1_biological_process`,
        `cellular_component` AS `T3_cellular_component`
WHERE (`T1_biological_process`.`id` = `T2_biological_process`.`id`
       AND `T2_biological_process`.`biological_process` IS NOT NULL
       AND `T2_biological_process`.`id` = `T5_biological_process`.`id`
       AND `T3_cellular_component`.`id` = `T4_cellular_component`.`id`
       AND `T3_cellular_component`.`id` = `T5_cellular_component`.`id`
       AND `T4_cellular_component`.`cellular_component` IS NOT NULL
       AND `T5_biological_process`.`id` = `T5_cellular_component`.`id`
```

Fonte: O Autor

A Consulta 3 tem por objetivo recuperar a combinação de processos biológicos que ocorrem em componentes celulares específicos. Ela demorou 26 segundos para ser concluída. No total foram recuperados 9.017 registros. Os 10 primeiros resultados são descritos no Quadro 6.3.

Quadro 6.3: Os 10 primeiros resultados da Consulta 3.

?biologicalProcess	?cellularComponent
L-methionine salvage [GO:0071267]	cytosol [GO:0005829]
L-methionine salvage [GO:0071267]	extracellular exosome [GO:0070062]
methionine biosynthetic process [GO:0009086]	cytosol [GO:0005829]
methionine biosynthetic process [GO:0009086]	extracellular exosome [GO:0070062]
S-adenosylmethionine metabolic process [GO:0046500]	cytosol [GO:0005829]
S-adenosylmethionine metabolic process [GO:0046500]	extracellular exosome [GO:0070062]
S-methylmethionine cycle [GO:0033528]	cytosol [GO:0005829]
S-methylmethionine cycle [GO:0033528]	extracellular exosome [GO:0070062]
S-methylmethionine metabolic process [GO:0033477]	cytosol [GO:0005829]
S-methylmethionine metabolic process [GO:0033477]	extracellular exosome [GO:0070062]

Fonte: O Autor

6.1.3.4 Consulta 4

Recuperar os processos biológicos promovidos por proteínas

Listing 6.7: Consulta 4 em SPARQL.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX go: <http://purl.obolibrary.org/obo/go.owl#>
PREFIX btl2: <http://purl.org/biotop/btl2.owl#>
PREFIX pr: <http://purl.obolibrary.org/obo/pr#>

SELECT DISTINCT ?biologicalProcess ?proteinName
WHERE {
    ?biologicalProcessId a go:biological_process ;
        rdfs:label ?biologicalProcess .
    ?proteinNameId a pr:PR_000000001 ;
        rdfs:label ?proteinName .
    ?biologicalProcessId btl2:hasAgent ?proteinNameId .
}
```

Fonte: O Autor

Listing 6.8: Consulta 4 reescrita em SQL.

```
SELECT DISTINCT `T2_biological_process`.`biological_process`,
                `T4_protein_name`.`protein_name`,
                `T3_protein_name`.`id`,
                `T1_biological_process`.`id`
FROM    `biological_process` AS `T5_biological_process`,
```

```
        `protein_name` AS `T3_protein_name`,
        `protein_name` AS `T5_protein_name`,
        `biological_process` AS `T2_biological_process`,
        `biological_process` AS `T1_biological_process`,
        `protein_name` AS `T4_protein_name`
WHERE (`T1_biological_process`.`id` = `T2_biological_process`.`id`
        AND `T2_biological_process`.`biological_process` IS NOT NULL
        AND `T2_biological_process`.`id` = `T5_biological_process`.`id`
        AND `T3_protein_name`.`id` = `T5_protein_name`.`id`
        AND `T4_protein_name`.`id` = `T5_protein_name`.`id`
        AND `T4_protein_name`.`protein_name` IS NOT NULL
        AND `T5_biological_process`.`id` = `T5_protein_name`.`id` )
```

Fonte: O Autor

A Consulta 4 tem por objetivo recuperar todos os processos biológicos nos quais uma determinada proteína é a responsável direta pela realização do processo. Ela demorou 23 segundos para ser concluída. No total foram recuperados 6.273 registros. Os 10 primeiros resultados são descritos no Quadro 6.4.

Quadro 6.4: Os 10 primeiros resultados da Consulta 4.

?biologicalProcess	**?proteinName**
methionine biosynthetic process [GO:0009086]	Bifunctional homocysteine S-methyltransferase/5,10 -methylenetetrahydrofolate reductase [Includes: Homocysteine S-methyltransferase (EC 2.1.1.10) (S-methylmethionine:homocysteine methyltransferase)
tetrahydrofolate interconversion [GO:0035999]	Bifunctional homocysteine S-methyltransferase/5,10 -methylenetetrahydrofolate reductase [Includes: Homocysteine S-methyltransferase (EC 2.1.1.10) (S-methylmethionine:homocysteine methyltransferase)
methionine biosynthetic process [GO:0009086]	5,10-methylenetetrahydrofolate reductase (EC 1.5.1.20)]
tetrahydrofolate interconversion [GO:0035999]	5,10-methylenetetrahydrofolate reductase (EC 1.5.1.20)]
L-methionine salvage [GO:0071267]	S-methylmethionine–homocysteine S-methyltransferase BHMT2 (SMM-hcy methyltransferase) (EC 2.1.1.10) (Betaine–homocysteine S-methyltransferase 2)
methionine biosynthetic process [GO:0009086]	S-methylmethionine–homocysteine S-methyltransferase BHMT2 (SMM-hcy methyltransferase) (EC 2.1.1.10) (Betaine–homocysteine S-methyltransferase 2)
S-adenosylmethionine metabolic process [GO:0046500]	S-methylmethionine–homocysteine S-methyltransferase BHMT2 (SMM-hcy methyltransferase) (EC 2.1.1.10) (Betaine–homocysteine S-methyltransferase 2)
S-methylmethionine cycle [GO:0033528]	S-methylmethionine–homocysteine S-methyltransferase BHMT2 (SMM-hcy methyltransferase) (EC 2.1.1.10) (Betaine–homocysteine S-methyltransferase 2)
S-methylmethionine metabolic process [GO:0033477]	Homocysteine S-methyltransferase 1 (EC 2.1.1.10) (S-methylmethionine:homocysteine methyltransferase 1) (AtHMT-1) (SMM:Hcy S-methyltransferase 1)
methionine biosynthetic process [GO:0009086]	Homocysteine S-methyltransferase 1 (EC 2.1.1.10) (S-methylmethionine:homocysteine methyltransferase 1) (AtHMT-1) (SMM:Hcy S-methyltransferase 1)

Fonte: O Autor

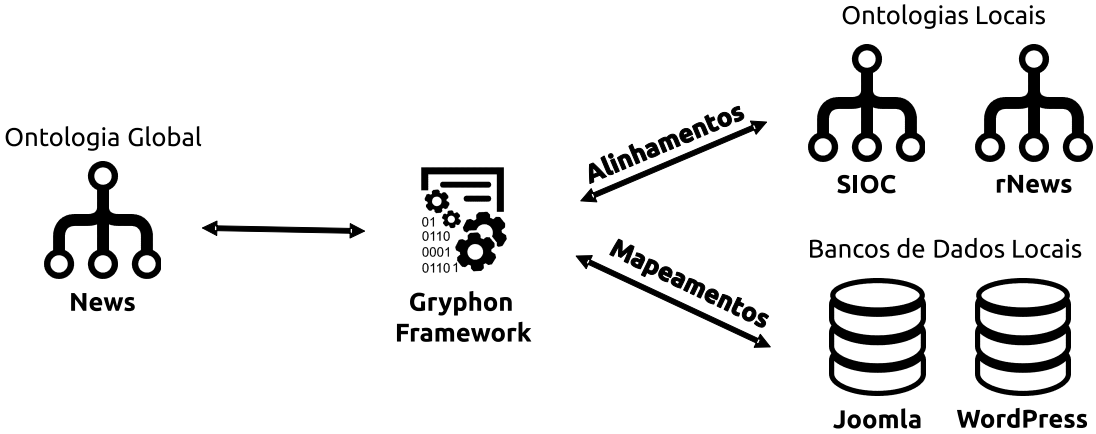
6.2 Experimento 2

Este segundo experimento consiste em integrar ontologias e bancos de dados, do domínio de notícias, utilizando o Gryphon Framework. O cenário de integração (Figura 6.3) é formado por uma ontologia global, duas ontologias locais (e seus alinhamentos com a ontologia global) e

dois bancos de dados locais (e seus mapeamentos com a ontologia global), em que:

- A ontologia global será a News (Seção 6.2.1);
- As ontologias locais serão a Semantically-Interlinked Online Communities (SIOC) (Seção 6.2.2) e rNews (Seção 6.2.3); e
- Os bancos de dados locais serão do Joomla (Seção 6.2.4) e do WordPress (Seção 6.2.5).

Figura 6.3: Cenário de integração do Experimento 2.



Fonte: O Autor

O objetivo desta integração é mostrar o potencial da arquitetura proposta e do Gryphon Framework. Foram integradas quatro fontes de dados heterogêneas (duas ontologias e dois bancos de dados) do mesmo domínio, utilizando uma camada semântica.

Para cada fonte de dados local foram inseridas dez notícias retiradas aleatóriamente de sites como The Verge[2], Reuters[3], BBC[4], TechCrunch[5], Wired[6], Huffington Post[7] e The Guardian[8]. Estas notícias pertencem às categorias "Tech" (tecnologia), "Science" (ciência) e "Business" (negócios).

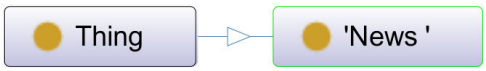
A classe Java implementada utilizando o Gryphon para este experimento encontra-se no Apêndice C. Os alinhamentos entre as ontologias locais com a ontologia global encontram-se no Apêndice D. Os mapeamentos entre os bancos de dados locais e a ontologia global estão disponíveis no Apêndice E.

[2]http://theverge.com
[3]http://reuters.com
[4]http://bbc.com
[5]http://techcrunch.com
[6]http://wired.com
[7]http://huffingtonpost.com
[8]http://theguardian.com

6.2.1 News

A ontologia News[9] foi escolhida para ser a ontologia global pelo fato de ser genérica o suficiente para ser alinhada e mapeada com as fontes locais. Como visto na Figura 6.4, essa ontologia possui apenas a classe *News*. Essa classe possui, originalmente, as propriedades *title*, *description* e *publishedOn*. A propriedade *category* foi inserida para possibilitar este experimento.

Figura 6.4: Ontologia global News.

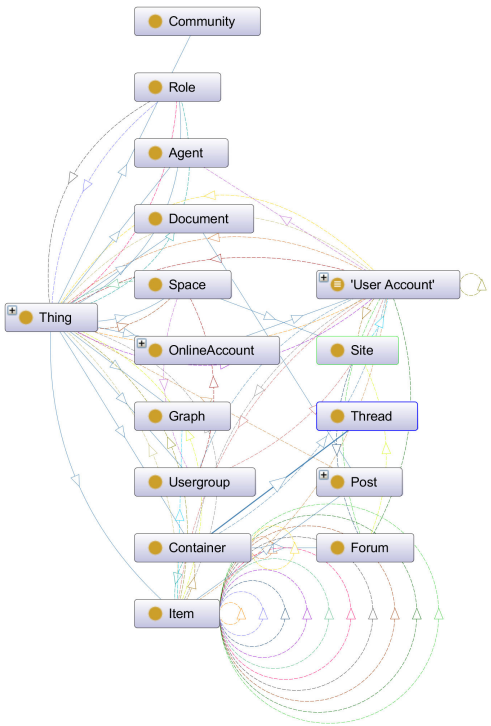


Fonte: O Autor

6.2.2 SIOC

A ontologia SIOC[10] possui conceitos e propriedades para descrever informações presentes em comunidades *online*, *e.g.*, Blogs, Wikis e Forúns. A SIOC possui diversas classes (Figura 6.5). Neste experimento foi utilizada apenas a classe *Post*, e suas propriedades, por ser o conceito mais próximo de notícia.

Figura 6.5: Ontologia local SIOC.



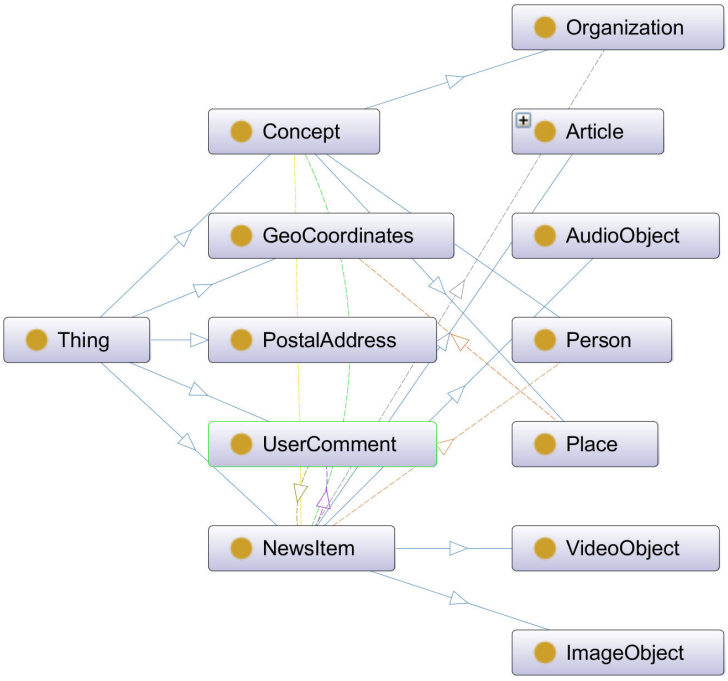
Fonte: O Autor

[9]http://ebiquity.umbc.edu/ontology/news.owl
[10]http://rdfs.org/sioc/spec/

6.2.3 rNews

A ontologia rNews[11] possui conceitos que representam um portal de notícias com conteúdo multimídia (áudio, foto e vídeo). Esta é representada na Figura 6.6. A classe *Article* foi utilizada neste experimento, juntamente com suas propriedades.

Figura 6.6: Ontologia local rNews.



Fonte: O Autor

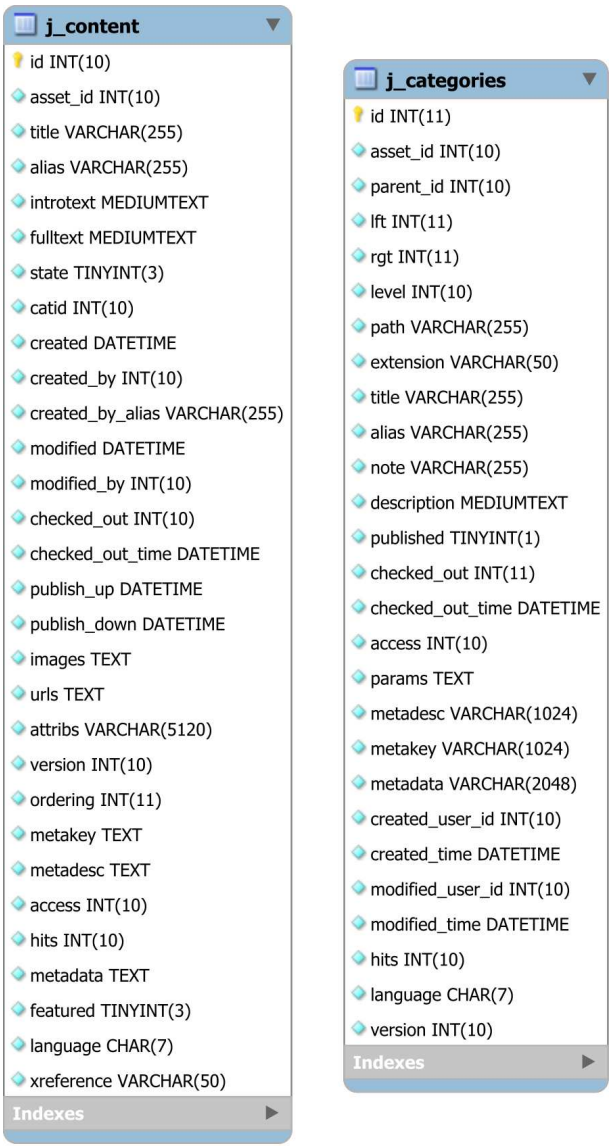
6.2.4 Joomla

Joomla[12] é um Content Management System (CMS) de código aberto que facilita a construção de sites de notícias. CMS é um *software* usado para criar, editar, gerenciar e publicar conteúdo na *web* de forma consistentemente organizada permitindo que o mesmo seja modificado, removido e adicionado com facilidade.

O esquema do banco de dados do Joomla possui diversas tabelas. Apenas as tabelas descritas na Figura 6.7 foram utilizadas neste experimento.

[11] http://dev.iptc.org/files/rNews/rnews_1.0_draft3_rdfxml.owl
[12] http://joomla.org

Figura 6.7: Banco de dados local Joomla.



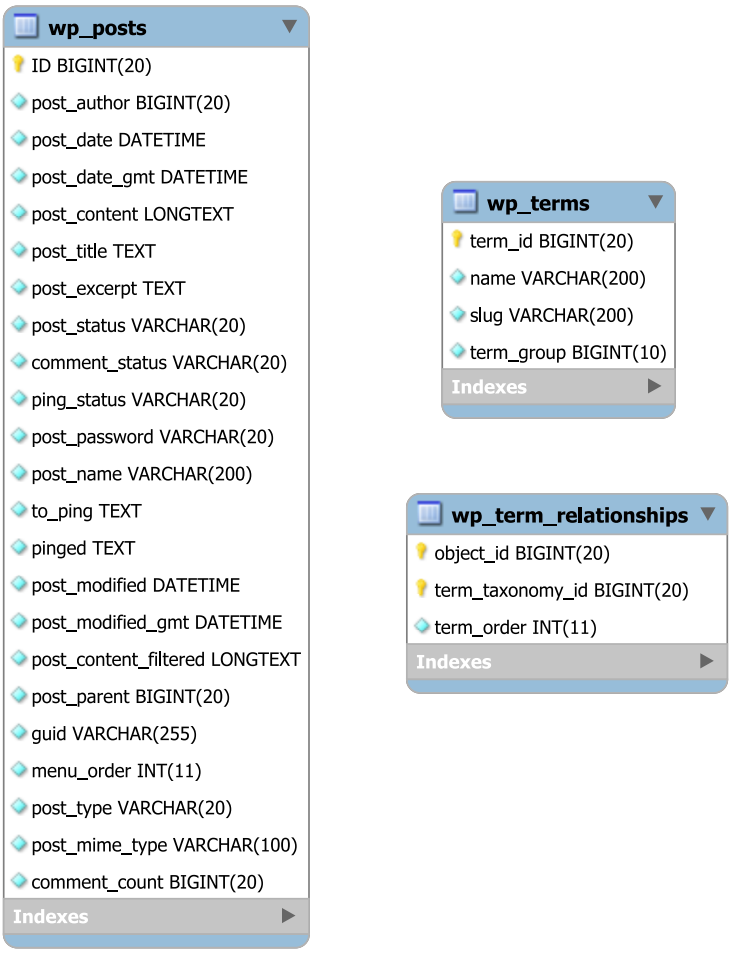
Fonte: O Autor

6.2.5 WordPress

Assim como o Joomla, o WordPress[13] é um CMS de código aberto que facilita a construção de blogs. As tabelas do WordPress utilizadas neste experimento são descritas na Figura 6.8.

[13]http://wordpress.org

Figura 6.8: Banco de dados local WordPress.



Fonte: O Autor

Ao comparar as tabelas do Joomla com as tabelas do WordPress é possível afirmar que:

- No Joomla, a relação entre as tabelas *j_content* (notícia) e *j_categories* (categoria) é de 1:N, ou seja, uma notícia pode ter apenas uma categoria, porém uma categoria pode estar associada a várias notícias; e
- No WordPress, a relação entre as tabelas *wp_posts* (notícia) e *wp_terms* (categoria) é de N:M (a tabela *wp_term_relationships* descreve essa relação), ou seja, uma notícia pode ter várias categorias, assim como uma categoria pode estar associada a várias notícias.

Essa diferença na relação entre notícia e categoria influencia diretamente a reescrita de consultas. A linguagem de mapeamento utilizada pelo Gryphon é capaz de descrever estes dois tipos de relação, possibilitando a reescrita correta das consultas.

6.2.6 Resultados

Foram criadas cinco consultas em Português. Estas consultas foram traduzidas para SPARQL utilizando o vocabulário da ontologia global News.

O Gryphon Framework reescreveu automáticamente essas consultas em SPARQL (para as ontologias locais SIOC e rNews) e em SQL (para os bancos de dados locais do Joomla e WordPress). Ou seja, para cada uma das cinco consultas originais, foram criadas mais quatro consultas (duas em SPARQL e duas em SQL).

Nas proximas subseções, serão descritas as consultas originais e as consultas reescritas, bem como os resultados de cada uma.

6.2.6.1 Consulta 1

Recuperar todas as notícias

Listing 6.9: Consulta 1 original em SPARQL.

```
PREFIX news: <http://ebiquity.umbc.edu/ontology/news.owl#>

SELECT DISTINCT ?title ?date ?category
WHERE {
    ?x a news:News ;
        news:title ?title ;
        news:publishedOn ?date ;
        news:category ?category .
}
```

Fonte: O Autor

A Consulta 1 tem como objetivo recuperar todas as notícias. Abaixo estão as consultas reescritas com seus respectivos resultados. O tempo de execução total das consultas foi de 10 segundos.

Listing 6.10: Consulta 1 reescrita em SPARQL para a ontologia SIOC.

```
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX dc: <http://purl.org/dc/terms/>

SELECT DISTINCT ?title ?date ?category
WHERE {
    ?x a sioc:Post ;
        dc:title ?title ;
        dc:date ?date ;
        dc:category ?category .
}
```

Fonte: O Autor

Quadro 6.5: Resultado da Consulta 1 da ontologia SIOC.

?title	?date	?category
Brompton Bicycle gears up for growth with bigger factory	2015-08-15	Business
Freaky “Sea Monster” Recovered From Centuries-Old Shipwreck	2015-08-16	Science
Airbus finalises $26.6bn plane deal with Indian airline IndiGo	2015-08-17	Business
Jeff Bezos defends Amazon after NYT exposé of working practices	2015-08-18	Business
Bitcoin’s forked: chief scientist launches alternative proposal for the currency	2015-08-18	Tech
Software upgrade grounds hundreds of flights over US east coast	2015-08-15	Tech
Amazon boss says Jeremy Clarkson’s Top Gear follow-on show “expensive but worth it”	2015-08-16	Tech
ISS Astronaut Shares Another Dazzling Northern Lights Video	2015-08-17	Science
This Simple Technique Can Instantly Reverse Negative Thoughts	2015-08-18	Science
New “Tatooine” Discovery Confirms Circumbinary Planets Aren’t Just Science Fiction	2015-08-16	Science

Fonte: O Autor

Listing 6.11: Consulta 1 reescrita em SPARQL para a ontologia rNews.

```
PREFIX rnews: <http://iptc.org/std/rNews/2011-10-07#>

SELECT DISTINCT ?title ?date ?category
WHERE {
    ?x a rnews:Article ;
        rnews:headline ?title ;
        rnews:dateCreated ?date ;
        rnews:articleSection ?category .
}
```

Fonte: O Autor

Quadro 6.6: Resultado da Consulta 1 da ontologia rNews.

?title	?date	?category
Online course teaches kids to program while having fun	2015-08-15	Tech
Life-size humanoid robot takes a walk in the woods	2015-08-16	Tech
Facebook updates Notes in move to get you blogging again	2015-08-17	Tech
Microsoft is allowing itself to detect pirated games on your Windows 10 PC	2015-08-18	Tech
This Jupiter-like gas giant could show us how planets form in the universe	2015-08-15	Science
Watch NASA test “the Ferrari of rocket engines” at 5PM ET	2015-08-16	Science
NASA is asking the public to design smartwatch apps for its astronauts	2015-08-17	Science
Cairn Energy cleared to start Senegal drilling operations	2015-08-15	Business
Longannet power station to close in March	2015-08-16	Business
Rates to rise soon, says Bank of England policymaker	2015-08-17	Business

Fonte: O Autor

Listing 6.12: Consulta 1 reescrita em SQL para o banco de dados Joomla.

```
SELECT DISTINCT `T2_j_content`.`title`,
                `T1_j_content`.`id`,
                `T4_j_categories`.`title`,
                `T3_j_content`.`publish_up`
FROM   `j_categories` AS `T4_j_categories`,
       `j_content` AS `T4_j_content`,
       `j_content` AS `T2_j_content`,
       `j_content` AS `T3_j_content`,
       `j_content` AS `T5_j_content`,
       `j_content` AS `T1_j_content`
WHERE  ( `T1_j_content`.`id` = `T5_j_content`.`id`
         AND `T2_j_content`.`id` = `T5_j_content`.`id`
         AND `T3_j_content`.`id` = `T5_j_content`.`id`
         AND `T4_j_categories`.`id` = `T4_j_content`.`catid`
         AND `T4_j_content`.`id` = `T5_j_content`.`id` )
```

Fonte: O Autor

Quadro 6.7: Resultado da Consulta 1 do banco de dados Joomla.

?title	?date	?category
Windows 10 is the end of cloud-free computing	2015-08-18	Tech
Google delays its Project Ara modular smartphone until 2016	2015-08-16	Tech
An exceptional planetary system discovered in Cassiopeia	2015-08-15	Science
US gives Shell the final approval it needs to drill for oil in the Arctic	2015-08-16	Science
Google Maps can now tell you if it's worth installing solar panels on your roof	2015-08-15	Science
Watch as astronauts soar over an Aurora Borealis in the space station	2015-08-17	Science
Art installation brings drone killings to life	2015-08-17	Business
The BMW Concept M4 GTS is an M4 on steroids 2014 and you'll be able to buy it	2015-08-15	Business
Samsung gives its premium headphones what they really need: a gold color option	2015-08-16	Business
BMW fixed the 3.0 CSL Hommage concept car, and now it looks amazing	2015-08-15	Business

Fonte: O Autor

Listing 6.13: Consulta 1 reescrita em SQL para o banco de dados WordPress.

```
SELECT DISTINCT `T3_wp_posts`.`post_date`,
                `T2_wp_posts`.`post_title`,
                `T1_wp_posts`.`id`,
                `T4_wp_terms`.`name`
FROM   `wp_terms` AS `T4_wp_terms`,
       `wp_posts` AS `T5_wp_posts`,
       `wp_posts` AS `T3_wp_posts`,
       `wp_posts` AS `T1_wp_posts`,
       `wp_posts` AS `T2_wp_posts`,
       `wp_posts` AS `T4_wp_posts`,
       `wp_term_relationships` AS `T4_wp_term_relationships`
WHERE  ( ( `T2_wp_posts`.`post_status` = 'publish' )
         AND ( `T2_wp_posts`.`post_type` = 'post' )
         AND ( `T3_wp_posts`.`post_status` = 'publish' )
         AND ( `T3_wp_posts`.`post_type` = 'post' )
         AND ( `T4_wp_posts`.`post_status` = 'publish' )
         AND ( `T4_wp_posts`.`post_type` = 'post' )
         AND ( `T5_wp_posts`.`post_status` = 'publish' )
         AND ( `T5_wp_posts`.`post_type` = 'post' )
         AND `T1_wp_posts`.`id` = `T2_wp_posts`.`id`
```

```
AND `T2_wp_posts`.`id` = `T3_wp_posts`.`id`
AND `T2_wp_posts`.`id` = `T4_wp_posts`.`id`
AND `T2_wp_posts`.`id` = `T5_wp_posts`.`id`
AND `T4_wp_posts`.`id` = `T4_wp_term_relationships`.`object_id
AND `T4_wp_term_relationships`.`term_taxonomy_id` =
    `T4_wp_terms`.`term_id` )
```

Fonte: O Autor

Quadro 6.8: Resultado da Consulta 1 do banco de dados WordPress.

?title	?date	?category
Amazon boss Jeff Bezos defends company's workplace culture	2015-08-17	Tech
Samsung Galaxy S6 Edge+ and Galaxy Note 5 unveiled	2015-08-15	Tech
Android security patch "flawed"	2015-08-15	Tech
Police investigate "first cyber-flashing" case	2015-08-16	Tech
Launch date for Inmarsat's delayed Global Xpress spacecraft	2015-08-16	Science
Young "alien Jupiter" planet discovered	2015-08-17	Science
Rosetta: Comet 67P makes closest approach to Sun	2015-08-15	Science
Japan hit by weaker economic growth	2015-08-16	Business
Bank of England member warns of low interest rates risk	2015-08-17	Business
How developing countries are paying a high price for the global mineral boom	2015-08-15	Business

Fonte: O Autor

6.2.6.2 Consulta 2

Recuperar notícias de ciência

Listing 6.14: Consulta 2 original em SPARQL.

```
PREFIX news: <http://ebiquity.umbc.edu/ontology/news.owl#>

SELECT DISTINCT ?title
WHERE {
    ?x a news:News ;
    news:title ?title ;
    news:category "Science" .
}
```

Fonte: O Autor

A Consulta 2 tem como objetivo recuperar as notícias que pertencem a categoria *"Science"*. Abaixo estão as consultas reescritas com seus respectivos resultados. O tempo de execução total das consultas foi de 7 segundos.

Listing 6.15: Consulta 2 reescrita em SPARQL para a ontologia SIOC.

```
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX dc: <http://purl.org/dc/terms/>

SELECT DISTINCT ?title
WHERE {
    ?x a sioc:Post ;
        dc:title ?title ;
        dc:category "Science" .
}
```

Fonte: O Autor

Quadro 6.9: Resultado da Consulta 2 da ontologia SIOC.

?title
Freaky “Sea Monster” Recovered From Centuries-Old Shipwreck
ISS Astronaut Shares Another Dazzling Northern Lights Video
This Simple Technique Can Instantly Reverse Negative Thoughts
New “Tatooine” Discovery Confirms Circumbinary Planets Aren’t Just Science Fiction

Fonte: O Autor

Listing 6.16: Consulta 2 reescrita em SPARQL para a ontologia rNews.

```
PREFIX rnews: <http://iptc.org/std/rNews/2011-10-07#>

SELECT DISTINCT ?title
WHERE {
    ?x a rnews:Article ;
        rnews:headline ?title ;
        rnews:articleSection "Science" .
}
```

Fonte: O Autor

Quadro 6.10: Resultado da Consulta 2 da ontologia rNews.

?title
This Jupiter-like gas giant could show us how planets form in the universe
Watch NASA test “the Ferrari of rocket engines” at 5PM ET
NASA is asking the public to design smartwatch apps for its astronauts

Fonte: O Autor

Listing 6.17: Consulta 2 reescrita em SQL para o banco de dados Joomla.

```
SELECT DISTINCT `T2_j_content`.`title`,
                `T1_j_content`.`id`
FROM   `j_content` AS `T2_j_content`,
       `j_categories` AS `T3_j_categories`,
       `j_content` AS `T3_j_content`,
       `j_content` AS `T1_j_content`
WHERE  ( `T1_j_content`.`id` = `T3_j_content`.`id`
         AND `T2_j_content`.`id` = `T3_j_content`.`id`
         AND `T3_j_categories`.`id` = `T3_j_content`.`catid`
         AND `T3_j_categories`.`title` = 'Science' )
```

Fonte: O Autor

Quadro 6.11: Resultado da Consulta 2 do banco de dados Joomla.

?title
US gives Shell the final approval it needs to drill for oil in the Arctic
Google Maps can now tell you if it 2019s worth installing solar panels on your roof
Watch as astronauts soar over an Aurora Borealis in the space station
An exceptional planetary system discovered in Cassiopeia

Fonte: O Autor

Listing 6.18: Consulta 2 reescrita em SQL para o banco de dados WordPress.

```
SELECT DISTINCT `T2_wp_posts`.`post_title`,
                `T1_wp_posts`.`id`
FROM   `wp_terms` AS `T3_wp_terms`,
       `wp_term_relationships` AS `T3_wp_term_relationships`,
       `wp_posts` AS `T2_wp_posts`,
       `wp_posts` AS `T3_wp_posts`,
       `wp_posts` AS `T1_wp_posts`
WHERE  ( ( `T2_wp_posts`.`post_status` = 'publish' )
         AND ( `T2_wp_posts`.`post_type` = 'post' )
         AND ( `T3_wp_posts`.`post_status` = 'publish' )
         AND ( `T3_wp_posts`.`post_type` = 'post' )
         AND `T1_wp_posts`.`id` = `T2_wp_posts`.`id`
         AND `T2_wp_posts`.`id` = `T3_wp_posts`.`id`
         AND `T3_wp_posts`.`id` = `T3_wp_term_relationships`.`object_id
         AND `T3_wp_term_relationships`.`term_taxonomy_id` =
             `T3_wp_terms`.`term_id`
         AND `T3_wp_terms`.`name` = 'Science' )
```

Fonte: O Autor

Quadro 6.12: Resultado da Consulta 2 do banco de dados WordPress.

?title
Launch date for Inmarsat's delayed Global Xpress spacecraft
Young "alien Jupiter" planet discovered
Rosetta: Comet 67P makes closest approach to Sun

Fonte: O Autor

6.2.6.3 Consulta 3

Recuperar notícias que não são de ciência

Listing 6.19: Consulta 3 original em SPARQL

```
PREFIX news: <http://ebiquity.umbc.edu/ontology/news.owl#>

SELECT DISTINCT ?title ?category
WHERE {
    ?x a news:News ;
    news:title ?title ;
    news:category ?category .
    FILTER NOT EXISTS {
        FILTER (?category = "Science")
    }
}
```

Fonte: O Autor

A Consulta 3 tem como objetivo recuperar as notícias não pertencem a categoria *"Science"*. Abaixo estão as consultas reescritas com seus respectivos resultados. O tempo de execução total das consultas foi de 7 segundos.

Listing 6.20: Consulta 3 reescrita em SPARQL para a ontologia SIOC.

```
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX dc: <http://purl.org/dc/terms/>

SELECT DISTINCT ?title ?category
WHERE {
    ?x a sioc:Post ;
        dc:title ?title ;
        dc:category ?category .
```

```
    FILTER NOT EXISTS {
        FILTER (?category = "Science")
    }
}
```

Fonte: O Autor

Quadro 6.13: Resultado da Consulta 3 da ontologia SIOC.

?title	?category
Brompton Bicycle gears up for growth with bigger factory	Business
Airbus finalises $26.6bn plane deal with Indian airline IndiGo	Business
Jeff Bezos defends Amazon after NYT exposé of working practices	Business
Bitcoin's forked: chief scientist launches alternative proposal for the currency	Tech
Software upgrade grounds hundreds of flights over US east coast	Tech
Amazon boss says Jeremy Clarkson's Top Gear follow-on show "expensive but worth it"	Tech

Fonte: O Autor

Listing 6.21: Consulta 3 reescrita em SPARQL para a ontologia rNews.

```
PREFIX rnews: <http://iptc.org/std/rNews/2011-10-07#>

SELECT DISTINCT ?title ?category
WHERE {
    ?x a rnews:Article ;
        rnews:headline ?title ;
        rnews:articleSection ?category .
    FILTER NOT EXISTS {
        FILTER (?category = "Science")
    }
}
```

Fonte: O Autor

Quadro 6.14: Resultado da Consulta 3 da ontologia rNews.

?title	?category
Online course teaches kids to program while having fun	Tech
Life-size humanoid robot takes a walk in the woods	Tech
Facebook updates Notes in move to get you blogging again	Tech
Microsoft is allowing itself to detect pirated games on your Windows 10 PC	Tech
Cairn Energy cleared to start Senegal drilling operations	Business
Longannet power station to close in March	Business
Rates to rise soon, says Bank of England policymaker	Business

Fonte: O Autor

Listing 6.22: Consulta 3 reescrita em SQL para o banco de dados Joomla.

```
SELECT DISTINCT `T2_j_content`.`title`,
                `T1_j_content`.`id`,
                `T3_j_categories`.`title`
FROM   `j_content` AS `T2_j_content`,
       `j_categories` AS `T3_j_categories`,
       `j_content` AS `T3_j_content`,
       `j_content` AS `T1_j_content`
WHERE  ( `T1_j_content`.`id` = `T3_j_content`.`id`
         AND `T2_j_content`.`id` = `T3_j_content`.`id`
         AND `T3_j_categories`.`id` = `T3_j_content`.`catid` )
```

Quadro 6.15: Resultado da Consulta 3 do banco de dados Joomla.

?title	?category
Windows 10 is the end of cloud-free computing	Tech
Google delays its Project Ara modular smartphone until 2016	Tech
Art installation brings drone killings to life	Business
The BMW Concept M4 GTS is an M4 on steroids 2014 and you'll be able to buy it	Business
Samsung gives its premium headphones what they really need: a gold color option	Business
BMW fixed the 3.0 CSL Hommage concept car, and now it looks amazing	Business

Fonte: O Autor

Listing 6.23: Consulta 3 reescrita em SQL para o banco de dados WordPress.

```
SELECT DISTINCT `T2_wp_posts`.`post_title`,
                `T3_wp_terms`.`name`,
                `T1_wp_posts`.`id`
FROM   `wp_terms` AS `T3_wp_terms`,
```

```
        `wp_term_relationships` AS `T3_wp_term_relationships`,
        `wp_posts` AS `T2_wp_posts`,
        `wp_posts` AS `T3_wp_posts`,
        `wp_posts` AS `T1_wp_posts`
WHERE  ( ( `T2_wp_posts`.`post_status` = 'publish' )
          AND ( `T2_wp_posts`.`post_type` = 'post' )
          AND ( `T3_wp_posts`.`post_status` = 'publish' )
          AND ( `T3_wp_posts`.`post_type` = 'post' )
          AND `T1_wp_posts`.`id` = `T2_wp_posts`.`id`
          AND `T2_wp_posts`.`id` = `T3_wp_posts`.`id`
          AND `T3_wp_posts`.`id` = `T3_wp_term_relationships`.`object_id
          AND `T3_wp_term_relationships`.`term_taxonomy_id` =
              `T3_wp_terms`.`term_id` )
```

Fonte: O Autor

Quadro 6.16: Resultado da Consulta 3 do banco de dados WordPress.

?title	?category
Amazon boss Jeff Bezos defends company's workplace culture	Tech
Samsung Galaxy S6 Edge+ and Galaxy Note 5 unveiled	Tech
Android security patch "flawed"	Tech
Police investigate "first cyber-flashing" case	Tech
Japan hit by weaker economic growth	Business
Bank of England member warns of low interest rates risk	Business
How developing countries are paying a high price for the global mineral boom	Business

Fonte: O Autor

6.2.6.4 Consulta 4

Recuperar notícias criadas a partir de 17/08/2015

Listing 6.24: Consulta 4 original em SPARQL.

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX news: <http://ebiquity.umbc.edu/ontology/news.owl#>

SELECT DISTINCT ?title ?date
WHERE {
    ?x a news:News ;
    news:title ?title ;
    news:publishedOn ?date .
    FILTER (?date >= "2015-08-17"^^xsd:dateTime) .
```

```
} ORDER BY ?date
```

Fonte: O Autor

A Consulta 4 tem como objetivo recuperar as notícias criadas a partir da data 17/08/2015. Abaixo estão as consultas reescritas com seus respectivos resultados. O tempo de execução total das consultas foi de 9 segundos.

Listing 6.25: Consulta 4 reescrita em SPARQL para a ontologia SIOC.

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX dc: <http://purl.org/dc/terms/>

SELECT DISTINCT ?title ?date
WHERE {
    ?x a sioc:Post ;
        dc:title ?title ;
        dc:date ?date .
    FILTER (?date >= "2015-08-17"^^xsd:dateTime)
} ORDER BY ?date
```

Fonte: O Autor

Quadro 6.17: Resultado da Consulta 4 da ontologia SIOC.

?title	**?date**
Airbus finalises $26.6bn plane deal with Indian airline IndiGo	2015-08-17
ISS Astronaut Shares Another Dazzling Northern Lights Video	2015-08-17
Jeff Bezos defends Amazon after NYT exposé of working practices	2015-08-18
Bitcoin's forked: chief scientist launches alternative proposal for the currency	2015-08-18
This Simple Technique Can Instantly Reverse Negative Thoughts	2015-08-18

Fonte: O Autor

Listing 6.26: Consulta 4 reescrita em SPARQL para a ontologia rNews.

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rnews: <http://iptc.org/std/rNews/2011-10-07#>

SELECT DISTINCT ?title ?date
WHERE {
    ?x a rnews:Article ;
        rnews:headline ?title ;
        rnews:dateCreated ?date .
    FILTER (?date >= "2015-08-17"^^xsd:dateTime)
```

```
} ORDER BY ?date
```

Fonte: O Autor

Quadro 6.18: Resultado da Consulta 4 da ontologia rNews.

?title	?date
Facebook updates Notes in move to get you blogging again	2015-08-17
NASA is asking the public to design smartwatch apps for its astronauts	2015-08-17
Rates to rise soon, says Bank of England policymaker	2015-08-17
Microsoft is allowing itself to detect pirated games on your Windows 10 PC	2015-08-18

Fonte: O Autor

Listing 6.27: Consulta 4 reescrita em SQL para o banco de dados Joomla.

```
SELECT DISTINCT `T2_j_content`.`title`,
                `T1_j_content`.`id`,
                `T3_j_content`.`publish_up`
FROM   `j_content` AS `T2_j_content`,
       `j_content` AS `T3_j_content`,
       `j_content` AS `T1_j_content`
WHERE  ( `T1_j_content`.`id` = `T3_j_content`.`id`
         AND `T2_j_content`.`id` = `T3_j_content`.`id`
         AND `T3_j_content`.`publish_up` >= '2015-08-17' )
```

Fonte: O Autor

Quadro 6.19: Resultado da Consulta 4 do banco de dados Joomla.

?title	?date
Art installation brings drone killings to life	2015-08-17
Watch as astronauts soar over an Aurora Borealis in the space station	2015-08-18
Windows 10 is the end of cloud-free computing	2015-08-18

Fonte: O Autor

Listing 6.28: Consulta 4 reescrita em SQL para o banco de dados WordPress.

```
SELECT DISTINCT `T3_wp_posts`.`post_date`,
                `T2_wp_posts`.`post_title`,
                `T1_wp_posts`.`id`
FROM   `wp_posts` AS `T2_wp_posts`,
       `wp_posts` AS `T3_wp_posts`,
       `wp_posts` AS `T1_wp_posts`
WHERE  ( ( `T2_wp_posts`.`post_status` = 'publish' )
```

```
AND ( `T2_wp_posts`.`post_type` = 'post' )
AND ( `T3_wp_posts`.`post_status` = 'publish' )
AND ( `T3_wp_posts`.`post_type` = 'post' )
AND `T1_wp_posts`.`id` = `T2_wp_posts`.`id`
AND `T2_wp_posts`.`id` = `T3_wp_posts`.`id`
AND `T3_wp_posts`.`post_date` >= '2015-08-17' )
```

Fonte: O Autor

Quadro 6.20: Resultado da Consulta 4 do banco de dados WordPress.

?title	?date
Amazon boss Jeff Bezos defends company's workplace culture	2015-08-17
Young "alien Jupiter" planet discovered	2015-08-17
Bank of England member warns of low interest rates risk	2015-08-17
How developing countries are paying a high price for the global mineral boom	2015-08-18

Fonte: O Autor

6.2.6.5 Consulta 5

Recuperar notícias com a palavra-chave "planet"

Listing 6.29: Consulta 5 original em SPARQL.

```
PREFIX news: <http://ebiquity.umbc.edu/ontology/news.owl#>

SELECT DISTINCT ?title
WHERE {
    ?x a news:News ;
    news:title ?title .
    FILTER (regex(?title, "planet", "i")) .
}
```

Fonte: O Autor

A Consulta 5 tem como objetivo recuperar as notícias que possuem a palavra-chave "planet" no título. Abaixo estão as consultas reescritas com seus respectivos resultados. O tempo de execução total das consultas foi de 8 segundos.

Listing 6.30: Consulta 5 reescrita em SPARQL para a ontologia SIOC.

```
PREFIX sioc: <http://rdfs.org/sioc/ns#>
PREFIX dc: <http://purl.org/dc/terms/>

SELECT DISTINCT ?title
WHERE {
```

```
    ?x a sioc:Post ;
        dc:title ?title .
    FILTER regex(?title, "planet", "i")
}
```

Fonte: O Autor

Quadro 6.21: Resultado da Consulta 5 da ontologia SIOC.

?title
New “Tatooine” Discovery Confirms Circumbinary Planets Aren’t Just Science Fiction

Fonte: O Autor

Listing 6.31: Consulta 5 reescrita em SPARQL para a ontologia rNews.

```
PREFIX rnews: <http://iptc.org/std/rNews/2011-10-07#>

SELECT DISTINCT ?title
WHERE {
    ?x a rnews:Article ;
        rnews:headline ?title .
    FILTER regex(?title, "planet", "i")
}
```

Fonte: O Autor

Quadro 6.22: Resultado da Consulta 5 da ontologia rNews.

?title
This Jupiter-like gas giant could show us how planets form in the universe

Fonte: O Autor

Listing 6.32: Consulta 5 reescrita em SQL para o banco de dados Joomla.

```
SELECT DISTINCT `T2_j_content`.`title`,
                `T1_j_content`.`id`
FROM   `j_content` AS `T2_j_content`,
       `j_content` AS `T1_j_content`
WHERE  `T1_j_content`.`id` = `T2_j_content`.`id`
```

Fonte: O Autor

Quadro 6.23: Resultado da Consulta 5 do banco de dados Joomla.

?title
An exceptional planetary system discovered in Cassiopeia

Fonte: O Autor

Listing 6.33: Consulta 5 reescrita em SQL para o banco de dados WordPress.

```
SELECT DISTINCT `T2_wp_posts`.`post_title`,
                `T1_wp_posts`.`id`
FROM   `wp_posts` AS `T2_wp_posts`,
       `wp_posts` AS `T1_wp_posts`
WHERE  ( ( `T2_wp_posts`.`post_status` = 'publish' )
         AND ( `T2_wp_posts`.`post_type` = 'post' )
         AND `T1_wp_posts`.`id` = `T2_wp_posts`.`id` )
```

Fonte: O Autor

Quadro 6.24: Resultado da Consulta 5 do banco de dados WordPress.

?title
Young “alien Jupiter” planet discovered

Fonte: O Autor

6.3 Conclusão

Este capítulo descreveu dois experimentos que foram realizados neste trabalho a fim de solucionar problemas reais de integração de dados e comprovar a eficácia desta arquitetura e sua implementação.

No primeiro experimento, foram integrados bancos de dados do domínio biológico, utilizando uma ontologia global para possibilitar a mediação de consultas. No segundo experimento, foram integrados ontologias e bancos de dados do domínio de notícias, também utilizando uma ontologia global.

No próximo capítulo serão feitas as considerações finais deste trabalho. Serão abordadas as limitações, contribuições e uma breve análise comparativa com os trabalhos relacionados.

7

Conclusões

Neste trabalho foi criada uma estratégia, juntamente com uma implementação, capaz de realizar a integração semântica entre ontologias e bancos de dados.

Este trabalho apoiou-se num conjunto de conceitos e abordagens amplamente discutidos e validados pela literatura. O estudo de técnicas e ferramentas relacionadas à integração de dados, integração semântica e à própria web semântica foram essenciais para avaliação deste trabalho.

Para alcançar este objetivo, foram descritos trabalhos sobre a mesma temática em uma revisão da literatura, para conhecer o domínio. Foram descritos conceitos relacionados à web semântica e abordadas suas tecnologias. O conceito de ontologias, a linguagem de representação OWL, a linguagem de grafos RDF e a linguagem de consulta SPARQL foram abordados.

Em seguida, foi descrito o conceito de integração semântica. Foram discutidas as principais abordagens de integração, juntamente com suas vantagens e desvantagens. Os problemas de heterogeneidade dos dados foram descritos, bem como as soluções existentes para resolvê-los, *e.g.*, alinhamento de ontologias, mapeamento entre ontologia e banco de dados e reescrita de consultas.

Foram abordados alguns trabalhos relevantes ao contexto. Como foi possível identificar, existem soluções que buscam automatizar o processo de integração e minimizar o problema de heterogeneidade dos dados. Em comparação, é possível destacar algumas vantagens do Gryphon Framework, principalmente por dá suporte a SPARQL 1.1 e por possibilitar a integração simultânea de ontologias e bancos de dados.

Com base na avaliação da literatura, foi proposta uma arquitetura. Foi apresentada uma solução baseada nas abordagens OIS e OBDA. A nova arquitetura destaca-se por possibilitar a integração de ontologias e bancos de dados relacionais simultaneamente, utilizando uma camada semântica composta por uma ontologia.

Os componentes dessa arquitetura foram detalhados (formada por 6 componentes), bem como o processo de integração (que possui 6 etapas). Em seguida, foram levantados requisitos para garantir a aplicabilidade da arquitetura, levando em consideração o desenvolvimento de uma solução de integração semântica que implementa a arquitetura proposta.

Foi abordado o desenvolvimento do Gryphon Framework. Todo o processo de desenvol-

vimento e modo de utilização foram especificados. Adicionalmente, foram descritas as soluções existentes utilizadas em seu desenvolvimento (*i.e.*, Sesame, D2RQ, AML e Mediation). Estas soluções foram fundamentais para o desenvolvimento do trabalho. Esta implementação cumpriu todos os requisitos levantados pela arquitetura, *i.e.* o Gryphon Framework realiza todas as seis etapas de integração, pode ser utilizada em diferentes aplicações (por ser um *framework*) e seus resultados mostraram-se confiáveis nos experimentos.

Como prova de conceito, foram realizados dois experimentos a fim de comprovar a eficácia, tanto da arquitetura, quanto do Gryphon Framework.

O primeiro experimento consistiu em integrar uma base de dados extensa do domínio biológico (o UniProt) com a ontologia modularizada IntegrativO, esta sendo formada por módulos da GO, PR e ChEBI, organizados formalmente pela BTL2. Como resultado, foi possível identificar as facilidades em realizar integração semântica com o Gryphon Framework para o domínio biológico, com consultas reais criadas por um especialista do domínio, em um ambiente de utilização real. De maneira geral, o esforço necessário para integrar as fontes pode ser amenizado devido ao uso do Gryphon.

No segundo experimento, foram integradas duas ontologias (SIOC e rNews) e dois bancos de dados (um do Joomla e outro do WordPress) do domínio de notícias utilizando a ontologia News como camada semântica. Foram criadas cinco consultas e o resultado foi o esperado, *i.e.*, as consultas foram reescritas sem erros e os dados extraídos estavam corretos.

Com o desenvolvimento deste trabalho, é possível concluir que muitos avanços estão sendo feitos na área da integração semântica, a qual está em constante evolução. Entretanto, ainda há problemas de heterogeneidade a serem resolvidos (em nível de sintaxe, estrutura e semântica). Mesmo com soluções específicas para problemas de integração (como alinhamento, mapeamento ou reescritas e consultas) o processo de integração não é totalmente automático. Porém, é possível minimizar o esforço do usuário disponibilizando uma estratégia que utilize estas soluções em conjunto a fim de automatizar este processo ao máximo. Justamente sobre este ponto que o presente trabalho apresenta uma de suas contribuições.

7.1 Análise Comparativa

No Capítulo 3 foram descritas as principais características das soluções de integração semântica existentes. A seguir será realizada uma comparação destes trabalhos com a arquitetura e implementação propostas.

Aber-OWL é uma solução especializada em integrar ontologias do domínio biológico. O Gryphon Framework é uma ferramenta de propósito geral, é possível utilizá-lo em qualquer domínio. Tanto Aber-OWL quanto Gryphon Framework realizam consultas em SPARQL. Porém, cabe ressaltar que Aber-OWL salva os resultados apenas em JSON, enquanto Gryphon Framework permite salvar em JSON, XML e CSV.

Exelixis facilita a manutenção dos alinhamentos quando as ontologias evoluem, porém,

ele da suporte a apenas ontologias em RDFS. Como mencionado na Seção 2.1.3, se comparado ao OWL (suportado pelo Gryphon Framework), a linguagem RDFS não possui tanta expressividade para descrever os conceitos de uma ontologia.

PROMPT, diferente de Aber-OWL e Exelixis, foi desenvolvido para mesclar ontologias. Realiza uma integração materializada que, no final do processo, cria uma nova ontologia. Como descrito na Seção 2.2.2, essa abordagem pode ocasionar problemas de inconsistência, *e.g.*, ontologias que estão em constante evolução podem adicionar/remover/alterar frequentemente o nome e definição de seus conceitos. Pelo fato da arquitetura proposta utilizar uma integração virtual, problemas como esses são mais difíceis de acontecerem.

Assim como Gryphon, -ontop- permite a realização de consultas SPARQL em bancos de dados relacionais, os tratando como grafos RDF virtuais. Enquanto o -ontop- dá suporte a SPARQL 1.0, Gryphon dá suporte a SPARQL 1.1, a última versão da linguagem que traz melhorias fundamentais, *e.g.*, funções agregadas[1], subconsultas[2], negação[3], caminhos de propriedade[4], novos comandos[5] e também facilitar a criação de consultas distribuídas entre diversas fontes de dados. -ontop- pode ser usado em conjunto com o OWLAPI, Sesame e Protégé, o que facilita sua extensão e aprimoramento.

De forma similar, com ONTOFUSION também é possível integrar bancos de dados por meio de uma camada semântica. Porém, esta solução dá suporte apenas ao domínio biomédico. Como já foi afirmado, é possível utilizar Gryphon Framework em qualquer domínio.

Também é possível integrar bancos de dados por meio de uma camada semântica com o OntoGrate. Essa ferramenta dá suporte às linguagens de consulta OWL-QL e SPARQL (versão não informada), enquanto Gryphon dá suporte apenas a SPARQL, linguagem mais utilizada para consultar dados integrados por meio de ontologias.

Após realizar essa comparação, foi possível perceber que, tanto a arquitetura, quanto sua implementação, precisam evoluir em alguns aspectos. Atualmente, os RDBMS com suporte no Gryphon Framework são MySQL e PostgreSQL, mas existem outros como Oracle[6], SQL Server[7] e H2[8]. O suporte a uma única linguagem de consulta pode se tornar uma limitação pelo fato de que o SPARQL foi criado para RDF e não para OWL. Existem outras linguagens que buscam resolver essa limitação, *e.g.*, SPARQL-DL (WANG; ZHAI; FAN, 2009), OWL-QL (FIKES; HAYES; HORROCKS, 2004) e Semantic Query-Enhanced Web Rule Language (SQWRL) (O'CONNOR; DAS, 2009).

Dentre as soluções abordadas neste capítulo, nenhuma apresentou a característica chave presente na arquitetura proposta: a integração entre ontologias e bancos de dados simultane-

[1] http://w3.org/TR/sparql-features/Aggregates
[2] http://w3.org/TR/sparql-features/Subqueries
[3] http://w3.org/TR/sparql-features/Negation
[4] http://w3.org/TR/sparql-features/Property_paths
[5] http://w3.org/TR/sparql-features/Language_syntax
[6] https://oracle.com/database
[7] microsoft.com/pt-br/server-cloud/products/sql-server/
[8] http://h2database.com

amente. As soluções abordadas na Seção 3.2 (Integração de Bancos de Dados Baseada em Ontologias) utilizam uma ontologia como camada semântica para integrar bancos de dados, já na Seção 3.1 (Integração de Ontologias), as soluções apresentadas integram apenas ontologias. A arquitetura proposta também possui uma ontologia como camada semântica (ontologia global), mas também possui ontologias e bancos de dados locais, ou seja, é possível integrar múltiplas ontologias e bancos de dados simultaneamente utilizando um vocabulário em comum (a camada semântica), como visto nas Seção 4.1 e Seção 4.2.

O Quadro 7.1 ilustra a comparação do presente trabalho com os demais. As principais caracteristicas observadas nesta análise foram integração de ontologias, integração de bancos de dados, linguagem de consulta e limitação de domínio.

Quadro 7.1: Tabela comparativa entre este trabalho e os trabalhos relacionados

	Integração de Ontologias	**Integração de Bancos de Dados**	**Linguagem de Consulta**	**Domínio**
Este trabalho	Sim	Sim	SPARQL 1.1	Sem limitação
Aber-OWL	Sim	Não	SPARQL 1.1	Domínio biológico
Exelixis	Sim	Não	SPARQL 1.1	Sem limitação
PROMPT	Sim	Não	-	Sem limitação
-ontop-	Não	Sim	SPARQL 1.0	Sem limitação
ONTOFUSION	Não	Sim	SPARQL 1.1	Domínio biomédico
OntoGrate	Não	Sim	OWL-QL, SPARQL 1.1	Sem limitação
Optique	Não	Sim	SPARQL 1.1	Sem limitação

Fonte: O Autor

7.2 Limitações

Durante o desenvolvimento deste trabalho, as seguintes limitações foram encontradas:

- O Gryphon Framework dá suporte apenas a linguagem de consulta SPARQL. Isto é uma limitação pelo fato de que SPARQL não é totalmente compatível com a linguagem de representação OWL;
- O Gryphon Framework não permite a edição de alinhamentos e mapeamentos por meio de métodos Java;
- O Gryphon Framework também não permite a integração de ontologias distribuidas pela Internet. É preciso acessá-las localmente; e
- Os mapeamentos gerados pelo D2RQ não são tão precisos quanto os alinhamentos gerados pelo AML.

7.3 Contribuições

As principais contribuições deste trabalho são enumeradas a seguir:

- Elaboração de uma estratégia para integração simultânea de múltiplas ontologias e bancos de dados, utilizando alinhamentos e mapeamentos, respectivamente, para possibilitar a reescrita de consultas em diferentes linguagens (*e.g.*, SPARQL e SQL);
- Disponibilização de uma solução de código aberto capaz de simplificar o processo de integração e facilitar a utilização da integração semântica em diversas aplicações, não restrito apenas a um domínio, mas podendo ser aplicado também nos domínios de dados abertos (SUBRAMANIAN et al., 2015; NISHIKATA; TOYODA, 2012; EBERIUS et al., 2012), dados corporativos (PEQUENO; PIRES, 2009; RAHMAN et al., 2008; WIJEGUNARATNE; FERNANDEZ; VALTOUDIS, 2000), esportes (NGUYEN et al., 2012), entre outros;
- Demonstração da viabilidade de reutilizar ferramentas de código aberto (plataformas, *frameworks* e API) para construir algo novo e funcional; e
- Finalmente, este trabalho também contribuiu com uma revisão da literatura na qual foram destacados os problemas e soluções relacionados à integração semântica.

7.4 Trabalhos Futuros

Com a conclusão deste trabalho, vislumbra-se um conjunto de possíveis trabalhos futuros para continuação do mesmo. Dentre eles é possível destacar:

- Realizar testes comparativos entre as plataformas -ontop- e o D2RQ (atual plataforma utilizada pelo Gryphon Framework para mapear os bancos de dados e reescrever consultas SPARQL para SQL). Caso o -ontop- se mostre superior ao D2RQ, seria válido substituir as plataformas;
- Criação de uma interface (*web* ou *desktop*) para facilitar a integração. O usuário poderia informar quem seria a ontologia global, e quais seriam as ontologias e/ou bancos de dados locais. Feito isso, seria realizado o alinhamento e mapeamento automaticamente e haveria a possibilidade do usuário editá-los;
- Estender a arquitetura para dar suporte à integração de sistemas de bancos de dados não relacionais (NoSQL) (KIRAN; VIJAYAKUMAR, 2014; POKORNY, 2013; HAN et al., 2011), *e.g.*, MongoDB[9], Cassandra[10], Redis[11] ou Neo4J[12]; e

[9] http://mongodb.org
[10] https://cassandra.apache.org
[11] http://redis.io
[12] http://neo4j.org

- Implementar algoritmos de aprendizado de máquina e treinar o Gryphon Framework para realizar alinhamentos e mapeamentos cada vez mais precisos com base em experiências anteriores (DOAN; DOMINGOS; HALEVY, 2001; DOAN; JAYANT MADHAVAN PEDRO DOMINGOS, 2003; DOAN; DOMINGOS; ALON, 2000).

Referências

ALASOUD, A.; HAARSLEV, V.; SHIRI, N. A Hybrid Approach for Ontology Integration. , [S.l.], 2009.

ALLEMANG, D.; HENDLER, J. Semantic Web for the Working Ontologist: effective modeling in rdfs and owl. , [S.l.], July 2011.

ANTONIOU, G.; HARMELEN, F. van. A Semantic Web Primer, 2nd Edition (Cooperative Information Systems). , [S.l.], Mar. 2008.

ASHBURNER, M. et al. Gene ontology: tool for the unification of biology. the gene ontology consortium. **Nature genetics**, [S.l.], v.25, n.1, p.25–9, May 2000.

BUSSLER, C. J. et al. (Ed.). **Reverse Engineering of Relational Databases to Ontologies**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. 327 – 341p. (Lecture Notes in Computer Science, v.3053).

AUER, S.; IVES, Z. **Integrating Ontologies and Relational Data**. 2007.

BEISSWANGER, E. et al. BioTop: an upper domain ontology for the life sciences: a description of its current structure, contents and interfaces to obo ontologies. **Applied Ontology**, [S.l.], v.3, n.4, p.205–212, Dec. 2008.

BERGAMASCHI, S. et al. Semantic integration of heterogeneous information sources. **Data & Knowledge Engineering**, [S.l.], v.36, n.3, p.215–249, Mar. 2001.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The Semantic Web. **Scientific American**, [S.l.], v.284, n.5, p.34–43, May 2001.

BHOWMICK, S. S.; KüNG, J.; WAGNER, R. (Ed.). **Database and Expert Systems Applications**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. (Lecture Notes in Computer Science, v.5690).

BIEN, S. J. et al. Bi-directional semantic similarity for gene ontology to optimize biological and clinical analyses. **Journal of the American Medical Informatics Association : JAMIA**, [S.l.], v.19, n.5, p.765–74, Jan. 2012.

BIZER, C. D2RQ - treating non-RDF databases as virtual RDF graphs. , [S.l.], 2004.

BREITMAN, K. **Web Semântica. A Internet do Futuro**. [S.l.]: LTC - GRUPO GEN, 2005.

BROEKSTRA, J.; KAMPMAN, A.; HARMELEN, F. van. Sesame: a generic architecture for storing and querying rdf and rdf schema. , [S.l.], p.54–68, June 2002.

BURGE, S. et al. Manual GO annotation of predictive protein signatures: the interpro approach to go curation. **Database**, [S.l.], v.2012, p.bar068–bar068, Feb. 2012.

BUSSLER, C. J. et al. (Ed.). **The Semantic Web**: research and applications. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. (Lecture Notes in Computer Science, v.3053).

CALVANESE, D. A framework for ontology integration. **The Emerging Semantic ...**, [S.l.], 2002.

CALVANESE, D. et al. Optique: obda solution for big data. In: REVISED SELECTED PAPERS OF ESWC 2013 SATELLITE EVENTS. **Anais...** [S.l.: s.n.], 2013. v.7955, p.293–295.

CALVANESE, D.; GIACOMO, G. D.; LENZERINI, M. Ontology of Integration and Integration of Ontologies. **Description Logics**, [S.l.], 2001.

CHITTENDEN, T. W. et al. nEASE: a method for gene ontology subclassification of high-throughput gene expression data. **Bioinformatics (Oxford, England)**, [S.l.], v.28, n.5, p.726–8, Mar. 2012.

CONSORTIUM, T. U. The universal protein resource (UniProt). **Nucleic acids research**, [S.l.], v.36, n.Database issue, p.D190–5, Jan. 2008.

CORRENDO, G. et al. SPARQL query rewriting for implementing data integration over linked data. In: INTERNATIONAL WORKSHOP ON DATA SEMANTICS - DATASEM '10, 1., New York, New York, USA. **Proceedings...** ACM Press, 2010. p.1.

COURTOT, M. et al. MIREOT: the minimum information to reference an external ontology term. **Nature Precedings**, [S.l.], v.6, n.1, p.23–33, Aug. 2009.

CRUZ, I. F.; ANTONELLI, F. P.; STROE, C. AgreementMaker: efficient matching for large real-world schemas and ontologies. , [S.l.], 2009.

CRUZ, I.; XIAO, H. The role of ontologies in data integration. **...intelligent systems for electrical engineering and ...**, [S.l.], p.1–18, 2005.

CULLOT, N.; GHAWI, R.; YéTONGNON, K. DB2OWL: a tool for automatic database-to-ontology mapping. , [S.l.], 2007.

DAVID, J. et al. The Alignment API 4.0. **Semantic Web**, [S.l.], v.2, n.1, p.3–10, Jan. 2011.

DEGTYARENKO, K. et al. ChEBI: a database and ontology for chemical entities of biological interest. **Nucleic acids research**, [S.l.], v.36, n.Database issue, p.D344–50, Jan. 2008.

DOAN, A.; DOMINGOS, P.; ALON, L. Learning Source Descriptions for Data Integration. , [S.l.], 2000.

DOAN, A.; DOMINGOS, P.; HALEVY, A. Y. Reconciling schemas of disparate data sources: a machine-learning approach. **ACM SIGMOD Record**, [S.l.], v.30, n.2, p.509–520, June 2001.

DOAN, A.; JAYANT MADHAVAN PEDRO DOMINGOS, A. H. Ontology Matching: a machine learning approach. , [S.l.], 2003.

DOAN, A.; MADHAVAN, J. Learning to map between ontologies on the semantic web. **...on World Wide Web**, [S.l.], p.662–673, 2002.

DOU, D.; MCDERMOTT, D.; QI, P. Ontology Translation on the Semantic Web. , [S.l.], 2005.

DOU, D.; QIN, H.; LEPENDU, P. OntoGrate: towards automatic integration for relational databases and the semantic web through an ontology-based framework. **International Journal of Semantic Computing**, [S.l.], v.04, n.01, p.123–151, Mar. 2010.

DzEROSKI, S. Multi-relational data mining. **ACM SIGKDD Explorations Newsletter**, [S.l.], v.5, n.1, p.1, July 2003.

EBERIUS, J. et al. Identifying and weighting integration hypotheses on open data platforms. In: FIRST INTERNATIONAL WORKSHOP ON OPEN DATA - WOD '12, New York, New York, USA. **Proceedings...** ACM Press, 2012. p.22.

EHRIG, M.; STAAB, S.; SURE, Y. Framework for Ontology Alignment and Mapping. , [S.l.], p.1–34, 2005.

EILBECK, K. et al. The Sequence Ontology: a tool for the unification of genome annotations. **Genome biology**, [S.l.], v.6, n.5, p.R44, Jan. 2005.

FIKES, R.; HAYES, P.; HORROCKS, I. OWL-QL - A Language for Deductive Query Answering on the Semantic. , [S.l.], 2004.

FREITAS, F. Ontologias : fundamentos, aplicações e a web semântica. , [S.l.], 2003.

GAGNON, M. Ontology-based integration of data sources. In: INTERNATIONAL CONFERENCE ON INFORMATION FUSION, 2007. **Anais...** IEEE, 2007. p.1–8.

GRAU, B. C. et al. Modular Reuse of Ontologies: theory and practice. , [S.l.], 2008.

GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge Acquisition**, [S.l.], v.5, n.2, p.199–220, June 1993.

GRUNINGER, M.; LEE, J. Ontology - Applications and Design. **Communications of the ACM**, [S.l.], v.45, n.2, p.39 – 41, 2002.

GUIZZARDI, G.; HALPIN, T. Ontological foundations for conceptual modelling. **Applied Ontology**, [S.l.], v.3, n.1-2, p.1–12, Jan. 2008.

HALL, P. A. V.; DOWLING, G. R. Approximate String Matching. **ACM Computing Surveys**, [S.l.], v.12, n.4, p.381–402, Dec. 1980.

HAN, J. et al. Survey on NoSQL database. In: INTERNATIONAL CONFERENCE ON PERVASIVE COMPUTING AND APPLICATIONS, 2011. **Anais...** IEEE, 2011. p.363–366.

HASTINGS, J. et al. The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013. **Nucleic acids research**, [S.l.], v.41, n.Database issue, p.D456–63, Jan. 2013.

HOEHNDORF, R. et al. Aber-OWL: a framework for ontology-based data access in biology. **BMC bioinformatics**, [S.l.], v.16, n.1, p.26, Jan. 2015.

HORROCKS, I. et al. OWL: a description logic based ontology language for the semantic web. , [S.l.], 2005.

HUBBARD, T. The Ensembl genome database project. **Nucleic Acids Research**, [S.l.], v.30, n.1, p.38–41, Jan. 2002.

ISLAM, A.; INKPEN, D. Semantic text similarity using corpus-based word similarity and string similarity. **ACM Transactions on Knowledge Discovery from Data**, [S.l.], v.2, n.2, p.1–25, July 2008.

JEAN-MARY, Y. R.; SHIRONOSHITA, E. P.; KABUKA, M. R. Ontology Matching with Semantic Verification. **Web semantics (Online)**, [S.l.], v.7, n.3, p.235–251, Sept. 2009.

JESúS BARRASA ÓSCAR CORCHO, A. G.-p. R2O, an Extensible and Semantically based Database-to-Ontology Mapping Language. , [S.l.], 2004.

JI, X. Social Data Integration and Analytics for Health Intelligence. In: VLDB 2014 PHD WORKSHOP. **Proceedings...** [S.l.: s.n.], 2014.

JIMéNEZ-RUIZ, E. et al. Safe and Economic Re-Use of Ontologies: a logic-based methodology and tool support. , [S.l.], 2008.

JING, Y.; JEONG, D.; BAIK, D.-K. SPARQL Graph Pattern Rewriting for OWL-DL Inference Query. In: FOURTH INTERNATIONAL CONFERENCE ON NETWORKED COMPUTING AND ADVANCED INFORMATION MANAGEMENT, 2008. **Anais...** IEEE, 2008. v.2, p.675–680.

KIRAN, V. K.; VIJAYAKUMAR, R. Ontology based data integration of NoSQL datastores. In: INTERNATIONAL CONFERENCE ON INDUSTRIAL AND INFORMATION SYSTEMS (ICIIS), 2014. **Anais...** IEEE, 2014. p.1–6.

KISHORE, R.; SHARMAN, R. **Computational Ontologies and Information Systems I: foundations**. 2004. v.14, n.1.

KLEIN, M. Combining and Relating Ontologies: an analysis of problems and solutions. , [S.l.], 2001.

KNOBBE, A. et al. Multi-relational data mining. , [S.l.], Jan. 1999.

KOLLIA, I.; GLIMM, B.; HORROCKS, I. SPARQL query answering over OWL ontologies. , [S.l.], p.382–396, May 2011.

KONDYLAKIS, H.; PLEXOUSAKIS, D. Exelixis: evolving ontology-based data integration system. In: MANAGEMENT OF DATA - SIGMOD '11, 2011., New York, New York, USA. **Proceedings...** ACM Press, 2011. p.1283.

KONTCHAKOV, R.; RODRÍGUEZ-MURO, M.; ZAKHARYASCHEV, M. Ontology-Based Data Access with Databases: a short course. , [S.l.], 2013.

LENZERINI, M. Data Integration: a theoretical perspective. In: ACM SIGMOD-SIGACT-SIGART SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS - PODS '02, New York, New York, USA. **Proceedings...** ACM Press, 2002. p.233.

LI, J. et al. RiMOM: a dynamic multistrategy ontology alignment framework. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.21, n.8, p.1218–1232, Aug. 2009.

LI, M.; DU, X.-Y.; WANG, S. Learning ontology from relational database. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYBERNETICS, 2005. **Anais...** IEEE, 2005. v.6, p.3410–3415 Vol. 6.

LINKOVá, Z. Ontology-Based Schema Integration. **SOFSEM (2)**, [S.l.], 2007.

LYTRAS, M. D. et al. (Ed.). **Visioning and Engineering the Knowledge Society. A Web Science Perspective**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. (Lecture Notes in Computer Science, v.5736).

MITZENMACHER, M.; VARGHESE, G. The complexity of object reconciliation, and open problems related to set difference and coding. In: ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING (ALLERTON), 2012. **Anais...** IEEE, 2012. p.1126–1132.

MONTECCHI-PALAZZI, L. et al. The PSI-MOD community standard for representation of protein modification data. **Nature biotechnology**, [S.l.], v.26, n.8, p.864–6, Aug. 2008.

NATALE, D. A. et al. The Protein Ontology: a structured representation of protein forms and complexes. **Nucleic acids research**, [S.l.], v.39, n.Database issue, p.D539–45, Jan. 2011.

NATALE, D. A. et al. Protein Ontology: a controlled structured network of protein entities. **Nucleic acids research**, [S.l.], v.42, n.Database issue, p.D415–21, Jan. 2014.

NGUYEN, Q.-M. et al. Towards Efficient Sport Data Integration through Semantic Annotation. In: FOURTH INTERNATIONAL CONFERENCE ON KNOWLEDGE AND SYSTEMS ENGINEERING, 2012. **Anais...** IEEE, 2012. p.99–106.

NISHIKATA, K.; TOYODA, T. BioLOD.org: ontology-based integration of biological linked open data. In: INTERNATIONAL WORKSHOP ON SEMANTIC WEB APPLICATIONS AND TOOLS FOR THE LIFE SCIENCES - SWAT4LS '11, 4., New York, New York, USA. **Proceedings...** ACM Press, 2012. p.92–93.

NOY, N. Semantic integration: a survey of ontology-based approaches. **ACM Sigmod Record**, [S.l.], v.33, n.4, p.65–70, 2004.

NOY, N. F.; MUSEN, M. A. PROMPT: algorithm and tool for automated ontology merging and alignment. , [S.l.], 2000.

O'CONNOR, M. J.; DAS, A. K. SQWRL: a query language for owl. **OWL: Experiences and Directions (OWLED)**, [S.l.], 2009.

PARSIA, B.; SATTLER, U.; SCHNEIDER, T. Mechanisms for Importing Modules. , [S.l.], 2009.

PASSIN, T. B. Explorer's Guide to the Semantic Web. , [S.l.], Mar. 2004.

PEQUENO, V. M. a.; PIRES, J. a. C. M. Reference model and perspective schemata inference for enterprise data integration. , [S.l.], p.135–152, Nov. 2009.

PéREZ-REY, D. et al. ONTOFUSION: ontology-based integration of genomic and clinical databases. **Computers in biology and medicine**, [S.l.], v.36, n.7-8, p.712–30, Jan. 2006.

POKORNY, J. NoSQL databases: a step to database scalability in web environment. **International Journal of Web Information Systems**, [S.l.], v.9, n.1, p.69–82, Mar. 2013.

QUILITZ, B.; LESER, U. Querying distributed RDF data sources with SPARQL. , [S.l.], p.524–538, June 2008.

RAHMAN, M. et al. Enterprise Data Integration towards Web Service Personalization. In: INTERNATIONAL CONFERENCE ON COMPUTER AND ELECTRICAL ENGINEERING, 2008. **Anais...** IEEE, 2008. p.715–720.

RHODE, M. Groß e. **Semantic Integration of Heterogeneous Software Specifications**. [S.l.]: Springer Science & Business Media, 2013. 330p.

RODRíGUEZ-MURO, M.; KONTCHAKOV, R.; ZAKHARYASCHEV, M. Ontop at work. , [S.l.], 2013.

RODRÍGUEZ-MURO, M.; REZK, M. Efficient SPARQL-to-SQL with R2RML mappings. **Web Semantics: Science, Services and Agents on the World Wide Web**, [S.l.], v.33, p.141–169, Mar. 2015.

SCHULZ, S.; BOEKER, M. BioTopLite: an upper level ontology for the life sciences. evolution, design and application. In: IEEE. **Anais...** [S.l.: s.n.], 2013. p.1889 – 1899.

SEDDIQUI, M. H.; AONO, M. An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. **Web Semantics: Science, Services and Agents on the World Wide Web**, [S.l.], v.7, n.4, p.344–356, Dec. 2009.

SHADBOLT, N.; BERNERS-LEE, T.; HALL, W. The Semantic Web Revisited. **IEEE Intelligent Systems**, [S.l.], v.21, n.3, p.96–101, May 2006.

SHVAIKO, P.; EUZENAT, J. Ontology matching: state of the art and future challenges. , [S.l.], v.X, n.X, p.1–20, 2012.

SMITH, B. Ontology. , [S.l.], 2003.

SMITH, B. et al. Relations in biomedical ontologies. **Genome biology**, [S.l.], v.6, n.5, p.R46, Jan. 2005.

SMITH, B. et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. **Nature biotechnology**, [S.l.], v.25, n.11, p.1251–5, Nov. 2007.

SOMA, R. et al. Semantic web technologies for smart oil field applications. , [S.l.], 2008.

SPEAR, A. D. **Ontology for the Twenty First Century: an introduction with recommendations**. 2006. 1–132p.

STOCKER, M. et al. SPARQL basic graph pattern optimization using selectivity estimation. In: PROCEEDING OF THE 17TH INTERNATIONAL CONFERENCE ON WORLD WIDE WEB - WWW '08, New York, New York, USA. **Anais...** ACM Press, 2008. p.595.

SUBRAMANIAN, A. et al. Semantic Integration of Structured Data Powered by Linked Open Data. In: INTERNATIONAL CONFERENCE ON WEB INTELLIGENCE, MINING AND SEMANTICS - WIMS '15, 5., New York, New York, USA. **Proceedings...** ACM Press, 2015. p.1–6.

TRINKUNAS, J.; VASILECAS, O. Building ontologies from relational databases using reverse engineering methods. In: COMPUTER SYSTEMS AND TECHNOLOGIES - COMPSYSTECH '07, 2007., New York, New York, USA. **Proceedings...** ACM Press, 2007. p.1.

USCHOLD, M. Ontologies and Semantics for Seamless Connectivity. , [S.l.], v.33, n.4, p.58–64, 2004.

UZDANAVICIUTE, V.; BUTLERIS, R. Ontology-based Foundations for Data Integration. , [S.l.], 2011.

WACHE, H. et al. Ontology-Based Integration of Information - A Survey of Existing Approaches. , [S.l.], 2001.

WANG, H.; ZHAI, S.; FAN, L. Query for Semantic Web Services Using SPARQL-DL. In: SECOND INTERNATIONAL SYMPOSIUM ON KNOWLEDGE ACQUISITION AND MODELING, 2009. **Anais...** IEEE, 2009. p.367–370.

WIJEGUNARATNE, I.; FERNANDEZ, G.; VALTOUDIS, J. A Federated Architecture for Enterprise Data Integration. , [S.l.], p.159, Apr. 2000.

ZIEGLER, P.; DITTRICH, K. R. Three Decades of Data Integration - All Problems Solved? **18th IFIP World Computer Congress**, [S.l.], v.12, p.3–12, 2004.

Apêndice

A

Classe Java do Experimento 1

```
package br.ufpe.cin.aac3.gryphon.example;

import java.io.File;
import java.net.URI;
import java.net.URISyntaxException;

import br.ufpe.cin.aac3.gryphon.Gryphon;
import br.ufpe.cin.aac3.gryphon.Gryphon.ResultFormat;
import br.ufpe.cin.aac3.gryphon.GryphonConfig;
import br.ufpe.cin.aac3.gryphon.GryphonUtil;
import br.ufpe.cin.aac3.gryphon.model.Database;
import br.ufpe.cin.aac3.gryphon.model.Ontology;

public final class MScExperiment1 {
  public static void main(String[] args) {
    // 1. Configure
    GryphonConfig.setWorkingDirectory(new File("
       integrationMScExperiment1"));
    GryphonConfig.setLogEnabled(true);
    GryphonConfig.setShowLogo(true);
    Gryphon.init();

    try {
      // 2. Set the global ontology and local sources
      Ontology globalOnt = new Ontology("integrativo", new URI(
         GryphonUtil.getCurrentURI() + "mscExperiment1/
         integrativO.owl"), new File("mscExperiment/sources"));
```

```
        Database localDB = new Database("localhost", 3306, "root
            ", "", "uniprot", Gryphon.DBMS.MySQL);

        Gryphon.setGlobalOntology(globalOnt);
        Gryphon.addLocalDatabase(localDB);

        // 3. Aligns ontologies and maps databases
        Gryphon.alignAndMap();

        // 4. Query Using SPARQL
        long startTime = System.currentTimeMillis();

        String query = getQuery1();
        Gryphon.query(query, ResultFormat.JSON);

        long endTime = System.currentTimeMillis();
        System.out.println("Query Duration: " + ((endTime -
            startTime) / 1000 % 60) + "s");
    } catch(URISyntaxException e){
        e.printStackTrace();
    }

    System.exit(0);
}

// Q1: Retrieve organisms that include homocysteine
private static String getQuery1(){
    return ""
        + "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
        + "PREFIX btl2: <http://purl.org/biotop/btl2.owl#> "
        + "SELECT DISTINCT ?organism "
        + "WHERE { "
            + "?organismId a btl2:organism ;"
            + "rdfs:label ?organism ."
            + "?homocysteineId a btl2:MonoMolecularEntity ."
            + "?organismId btl2:includes ?homocysteineId ."
        + "}";
}
```

```
// Q2: Retrieve biological processes that is included in
   organisms
private static String getQuery2(){
  return ""
    + "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
    + "PREFIX go: <http://purl.obolibrary.org/obo/go.owl#> "
    + "PREFIX btl2: <http://purl.org/biotop/btl2.owl#> "
    + "SELECT DISTINCT ?biologicalProcess ?organism "
    + "WHERE { "
      + "?biologicalProcessId a go:biological_process ;"
      + "rdfs:label ?biologicalProcess ."
      + "?organismId a btl2:organism ;"
      + "rdfs:label ?organism ."
      + "?biologicalProcessId btl2:isIncludedIn ?organismId
         ."
    + "}";
}

// Q3: Retrieve biological processes and the cellular
   components where they can be located
private static String getQuery3(){
  return ""
    + "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
    + "PREFIX go: <http://purl.obolibrary.org/obo/go.owl#> "
    + "PREFIX btl2: <http://purl.org/biotop/btl2.owl#> "
    + "SELECT DISTINCT ?biologicalProcess ?cellularComponent
       "
    + "WHERE { "
      + "?biologicalProcessId a go:biological_process ;"
      + "rdfs:label ?biologicalProcess ."
      + "?cellularComponentId a go:cellular_component ;"
      + "rdfs:label ?cellularComponent ."
      + "?biologicalProcessId btl2:isIncludedIn ?
         cellularComponentId ."
    + "}";
}

// Q4: Retrieve biological processes promoted by proteins
private static String getQuery4(){
```

```
    return ""
      + "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> "
      + "PREFIX go: <http://purl.obolibrary.org/obo/go.owl#> "
      + "PREFIX pr: <http://purl.obolibrary.org/obo/pr#> "
      + "PREFIX btl2: <http://purl.org/biotop/btl2.owl#> "
      + "SELECT DISTINCT ?biologicalProcess ?proteinName "
      + "WHERE { "
        + "?biologicalProcessId a go:biological_process ;"
        + "rdfs:label ?biologicalProcess ."
        + "?proteinNameId a pr:PR_000000001 ;"
        + "rdfs:label ?proteinName ."
        + "?biologicalProcessId btl2:hasAgent ?proteinNameId ."
      + "}";
  }
}
```

B

Mapeamento do Experimento 1

```
@prefix map: <#> .
@prefix db: <> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ
   /0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .
@prefix btl2: <http://purl.org/biotop/btl2.owl#> .
@prefix go: <http://purl.obolibrary.org/obo/go.owl#> .
@prefix pr: <http://purl.obolibrary.org/obo/pr#> .
@prefix integrativo: <http://www.cin.ufpe.br/~integrativo#> .

map:database a d2rq:Database;
  d2rq:jdbcDriver "com.mysql.jdbc.Driver";
  d2rq:jdbcDSN "jdbc:mysql://localhost:3306/uniprot";
  d2rq:username "root";
  d2rq:password "";
  jdbc:autoReconnect "true";
  jdbc:zeroDateTimeBehavior "convertToNull";
  .

# Table biological_process
map:biological_process a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "biological_process/@@biological_process.id@@
     ";
```

```
  d2rq:class go:biological_process;
  d2rq:classDefinitionLabel "biological_process";
  .
map:biological_process__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:property rdfs:label;
  d2rq:pattern "biological_process #@@biological_process.
     biological_process@@";
  .
map:biological_process_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:property vocab:biological_process_id;
  d2rq:propertyDefinitionLabel "biological_process id";
  d2rq:column "biological_process.id";
  d2rq:datatype xsd:integer;
  .
map:biological_process_biological_process a d2rq:PropertyBridge
   ;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:property vocab:biological_process_biological_process;
  d2rq:propertyDefinitionLabel "biological_process
     biological_process";
  d2rq:column "biological_process.biological_process";
  .

# Table cellular_component
map:cellular_component a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "cellular_component/@@cellular_component.id@@
     ";
  d2rq:class go:cellular_component;
  d2rq:classDefinitionLabel "cellular_component";
  .
map:cellular_component__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:cellular_component;
  d2rq:property rdfs:label;
  d2rq:pattern "cellular_component #@@cellular_component.
     cellular_component@@";
  .
```

```
map:cellular_component_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:cellular_component;
  d2rq:property vocab:cellular_component_id;
  d2rq:propertyDefinitionLabel "cellular_component id";
  d2rq:column "cellular_component.id";
  d2rq:datatype xsd:integer;
  .
map:cellular_component_cellular_component a d2rq:PropertyBridge
   ;
  d2rq:belongsToClassMap map:cellular_component;
  d2rq:property vocab:cellular_component_cellular_component;
  d2rq:propertyDefinitionLabel "cellular_component
     cellular_component";
  d2rq:column "cellular_component.cellular_component";
  .

# Table gene_name
map:gene_name a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "gene_name/@@gene_name.id@@";
  d2rq:class integrativo:GeneMolecule;
  d2rq:classDefinitionLabel "gene_name";
  .
map:gene_name__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:gene_name;
  d2rq:property rdfs:label;
  d2rq:pattern "gene_name #@@gene_name.gene_name@@";
  .
map:gene_name_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:gene_name;
  d2rq:property vocab:gene_name_id;
  d2rq:propertyDefinitionLabel "gene_name id";
  d2rq:column "gene_name.id";
  d2rq:datatype xsd:integer;
  .
map:gene_name_gene_name a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:gene_name;
  d2rq:property vocab:gene_name_gene_name;
  d2rq:propertyDefinitionLabel "gene_name gene_name";
```

```
  d2rq:column "gene_name.gene_name";
  .

# Table molecular_function
map:molecular_function a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "molecular_function/@@molecular_function.id@@
     ";
  d2rq:class go:molecular_function;
  d2rq:classDefinitionLabel "molecular_function";
  .
map:molecular_function__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:molecular_function;
  d2rq:property rdfs:label;
  d2rq:pattern "molecular_function #@@molecular_function.
     molecular_function@@";
  .
map:molecular_function_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:molecular_function;
  d2rq:property vocab:molecular_function_id;
  d2rq:propertyDefinitionLabel "molecular_function id";
  d2rq:column "molecular_function.id";
  d2rq:datatype xsd:integer;
  .
map:molecular_function_molecular_function a d2rq:PropertyBridge
   ;
  d2rq:belongsToClassMap map:molecular_function;
  d2rq:property vocab:molecular_function_molecular_function;
  d2rq:propertyDefinitionLabel "molecular_function
     molecular_function";
  d2rq:column "molecular_function.molecular_function";
  .

# Table molecule
map:molecule a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "molecule/@@molecule.id@@";
  d2rq:class btl2:MonoMolecularEntity;
  d2rq:classDefinitionLabel "molecule";
```

```
  .
map:molecule__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:molecule;
  d2rq:property rdfs:label;
  d2rq:pattern "molecule #@@molecule.molecule@@";
  .
map:molecule_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:molecule;
  d2rq:property vocab:molecule_id;
  d2rq:propertyDefinitionLabel "molecule id";
  d2rq:column "molecule.id";
  d2rq:datatype xsd:integer;
  .
map:molecule_molecule a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:molecule;
  d2rq:property vocab:molecule_molecule;
  d2rq:propertyDefinitionLabel "molecule molecule";
  d2rq:column "molecule.molecule";
  .

# Table organism
map:organism a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "organism/@@organism.id@@";
  d2rq:class btl2:organism;
  d2rq:classDefinitionLabel "organism";
  .
map:organism__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:organism;
  d2rq:property rdfs:label;
  d2rq:pattern "organism #@@organism.organism@@";
  .
map:organism_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:organism;
  d2rq:property vocab:organism_id;
  d2rq:propertyDefinitionLabel "organism id";
  d2rq:column "organism.id";
  d2rq:datatype xsd:integer;
  .
```

```
map:organism_organism a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:organism;
  d2rq:property vocab:organism_organism;
  d2rq:propertyDefinitionLabel "organism organism";
  d2rq:column "organism.organism";
  .

# Table protein_name
map:protein_name a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "protein_name/@@protein_name.id@@";
  d2rq:class pr:PR_000000001;
  d2rq:classDefinitionLabel "protein_name";
  .
map:protein_name__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:protein_name;
  d2rq:property rdfs:label;
  d2rq:pattern "protein_name #@@protein_name.protein_name@@";
  .
map:protein_name_id a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:protein_name;
  d2rq:property vocab:protein_name_id;
  d2rq:propertyDefinitionLabel "protein_name id";
  d2rq:column "protein_name.id";
  d2rq:datatype xsd:integer;
  .
map:protein_name_protein_name a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:protein_name;
  d2rq:property vocab:protein_name_protein_name;
  d2rq:propertyDefinitionLabel "protein_name protein_name";
  d2rq:column "protein_name.protein_name";
  .

# Properties
map:organism_molecule_includes a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:organism;
  d2rq:refersToClassMap map:molecule;
  d2rq:property btl2:includes;
    d2rq:join "organism.id => molecule.id";
```

```
  .
map:biological_process_organism_isIncludedIn a d2rq:
   PropertyBridge;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:refersToClassMap map:organism;
  d2rq:property btl2:isIncludedIn;
    d2rq:join "biological_process.id => organism.id";
  .
map:biological_process_cellular_component_isIncludedIn a d2rq:
   PropertyBridge;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:refersToClassMap map:cellular_component;
  d2rq:property btl2:isIncludedIn;
    d2rq:join "biological_process.id => cellular_component.id";
  .
map:biological_process_protein_name_component_isIncludedIn a
   d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:biological_process;
  d2rq:refersToClassMap map:protein_name;
  d2rq:property btl2:hasAgent;
    d2rq:join "biological_process.id => protein_name.id";
  .
```

C

Classe Java do Experimento 2

```
package br.ufpe.cin.aac3.gryphon.example;

import java.io.File;
import java.net.URI;
import java.net.URISyntaxException;

import br.ufpe.cin.aac3.gryphon.Gryphon;
import br.ufpe.cin.aac3.gryphon.Gryphon.ResultFormat;
import br.ufpe.cin.aac3.gryphon.GryphonConfig;
import br.ufpe.cin.aac3.gryphon.GryphonUtil;
import br.ufpe.cin.aac3.gryphon.model.Database;
import br.ufpe.cin.aac3.gryphon.model.Ontology;

public final class MScExperiment2 {
  public static void main(String[] args) {
    // 1. Configure
    GryphonConfig.setWorkingDirectory(new File("
       integrationMScExperiment2"));
    GryphonConfig.setLogEnabled(true);
    GryphonConfig.setShowLogo(true);
    Gryphon.init();

    try {
      // 2. Set the global ontology and local sources
      Ontology globalOntNews = new Ontology("news", new URI(
         GryphonUtil.getCurrentURI() + "mscExperiment2/news.owl
         "));
```

```
        Ontology localOntSioc = new Ontology("sioc", new URI(
            GryphonUtil.getCurrentURI() + "mscExperiment2/sioc.owl
            "));
        Ontology localOntRnews = new Ontology("rnews", new URI(
            GryphonUtil.getCurrentURI() + "mscExperiment2/rnews.
            owl"));
        Database localDBJoomla = new Database("localhost", 3306,
            "root", "", "joomla", Gryphon.DBMS.MySQL);
        Database localDBWordPress = new Database("localhost",
            3306, "root", "", "wordpress", Gryphon.DBMS.MySQL);

        Gryphon.setGlobalOntology(globalOntNews);
        Gryphon.addLocalOntology(localOntSioc);
        Gryphon.addLocalOntology(localOntRnews);
        Gryphon.addLocalDatabase(localDBJoomla);
        Gryphon.addLocalDatabase(localDBWordPress);

        // 3. Aligns ontologies and maps databases
        Gryphon.alignAndMap();

        // 4. Query Using SPARQL
        long startTime = System.currentTimeMillis();

        String query = getQuery1();
        Gryphon.query(query, ResultFormat.JSON);

        long endTime = System.currentTimeMillis();
        System.out.println("Query Duration: " + ((endTime -
            startTime) / 1000 % 60) + "s");
      } catch(URISyntaxException e){
        e.printStackTrace();
      }

      System.exit(0);
    }

    // Q1: Retrieve all news
    private static String getQuery1(){
      return ""
```

```
        + "PREFIX news: <http://ebiquity.umbc.edu/ontology/news
            .owl#> "
        + "SELECT DISTINCT ?title ?date ?category "
        + "WHERE { "
          + "?x a news:News ; "
          + "news:title ?title ; "
          + "news:publishedOn ?date ; "
          + "news:category ?category . "
        + "}";
  }

  // Q2: Retrieve science news
  private static String getQuery2(){
    return ""
        + "PREFIX news: <http://ebiquity.umbc.edu/ontology/news
            .owl#> "
        + "SELECT DISTINCT ?title "
        + "WHERE { "
          + "?x a news:News ; "
          + "news:title ?title ; "
          + "news:category 'Science' . "
        + "}";
  }

  // Q3: Retrieve news that are not from science
  private static String getQuery3(){
    return ""
        + "PREFIX news: <http://ebiquity.umbc.edu/ontology/news
            .owl#> "
        + "SELECT DISTINCT ?title ?category "
        + "WHERE { "
          + "?x a news:News ; "
          + "news:title ?title ; "
          + "news:category ?category . "
          + "FILTER NOT EXISTS { "
            + "FILTER (?category = 'Science') . "
          + " } . "
        + "}";
  }
```

```
  // Q4: Retrieve news starting in 08/17/2015
  private static String getQuery4(){
    return ""
        + "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#> "
        + "PREFIX news: <http://ebiquity.umbc.edu/ontology/news
           .owl#> "
        + "SELECT DISTINCT ?title ?date "
        + "WHERE { "
          + "?x a news:News ; "
          + "news:title ?title ; "
          + "news:publishedOn ?date . "
          + "FILTER (?date >= '2015-08-17'^^xsd:dateTime) . "
        + "} ORDER BY ?date";
  }

  // Q5: Retrieve news with 'planet' keyword
  private static String getQuery5(){
    return ""
        + "PREFIX news: <http://ebiquity.umbc.edu/ontology/news
           .owl#> "
        + "SELECT DISTINCT ?title "
        + "WHERE { "
          + "?x a news:News ; "
          + "news:title ?title . "
          + "FILTER (regex(?title, 'planet', 'i')) . "
        + "}";
  }
}
```

D

Alinhamentos do Experimento 2

D.1 Alinhamento da Ontologia SIOC com a Ontologia Global

```
<?xml version='1.0' encoding='utf-8'?>
<rdf:RDF xmlns='http://knowledgeweb.semanticweb.org/
    heterogeneity/alignment#'
    xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
    xmlns:xsd='http://www.w3.org/2001/XMLSchema#'
    xmlns:align='http://knowledgeweb.semanticweb.org/
        heterogeneity/alignment#'
    xmlns:edoal='http://ns.inria.org/edoal/1.0/#'>
<Alignment>
  <xml>yes</xml>
  <level>2EDOAL</level>
  <type>**</type>
  <method>fr.inrialpes.exmo.align.edoal.EDOALAlignment#toEDOAL
      </method>
  <onto1>
    <Ontology rdf:about="http://ebiquity.umbc.edu/ontology/news
        .owl#">
      <location></location>
      <formalism>
        <Formalism align:name="OWL1.0" align:uri="http://www.w3
            .org/2002/07/owl#"/>
      </formalism>
    </Ontology>
  </onto1>
```

```
<onto2>
  <Ontology rdf:about="http://rdfs.org/sioc/ns#">
    <location></location>
    <formalism>
      <Formalism align:name="OWL1.0" align:uri="http://www.w3
          .org/2002/07/owl#"/>
    </formalism>
  </Ontology>
</onto2>
<map>
  <Cell>
    <entity1>
      <edoal:Class rdf:about="http://ebiquity.umbc.edu/
          ontology/news.owl#News"/>
    </entity1>
    <entity2>
      <edoal:Class rdf:about="http://rdfs.org/sioc/ns#Post"/>
    </entity2>
    <relation>=</relation>
    <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
        float">1.0</measure>
  </Cell>
</map>
<map>
  <Cell>
    <entity1>
      <edoal:Property rdf:about="http://ebiquity.umbc.edu/
          ontology/news.owl#description"/>
    </entity1>
    <entity2>
      <edoal:Property rdf:about="http://rdfs.org/sioc/ns#
          content"/>
    </entity2>
    <relation>=</relation>
    <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
        float">1.0</measure>
  </Cell>
</map>
<map>
```

```
    <Cell>
      <entity1>
        <edoal:Property rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#title"/>
      </entity1>
      <entity2>
        <edoal:Property rdf:about="http://purl.org/dc/terms/
            title"/>
      </entity2>
      <relation>=</relation>
      <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">1.0</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1>
        <edoal:Property rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#publishedOn"/>
      </entity1>
      <entity2>
        <edoal:Property rdf:about="http://purl.org/dc/terms/
            date"/>
      </entity2>
      <relation>=</relation>
      <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">1.0</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1>
        <edoal:Property rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#category"/>
      </entity1>
      <entity2>
        <edoal:Property rdf:about="http://purl.org/dc/terms/
            category"/>
      </entity2>
```

```
        <relation >=</relation >
        <measure  rdf:datatype="http://www.w3.org/2001/XMLSchema#
            float">1.0</measure>
      </Cell>
   </map>
</Alignment>
</rdf:RDF>
```

D.2 Alinhamento da Ontologia rNews com a Ontologia Global

```
<?xml version='1.0' encoding='utf-8'?>
<rdf:RDF xmlns='http://knowledgeweb.semanticweb.org/
    heterogeneity/alignment#'
    xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
    xmlns:xsd='http://www.w3.org/2001/XMLSchema#'
    xmlns:align='http://knowledgeweb.semanticweb.org/
        heterogeneity/alignment#'
    xmlns:edoal='http://ns.inria.org/edoal/1.0/#'>
<Alignment>
  <xml>yes</xml>
  <level>2EDOAL</level>
  <type>**</type>
  <method>fr.inrialpes.exmo.align.edoal.EDOALAlignment#toEDOAL
      </method>
  <onto1>
    <Ontology rdf:about="http://ebiquity.umbc.edu/ontology/news
        .owl">
      <location></location>
      <formalism>
        <Formalism align:name="OWL1.0" align:uri="http://www.w3
            .org/2002/07/owl#"/>
      </formalism>
    </Ontology>
  </onto1>
  <onto2>
    <Ontology rdf:about="http://iptc.org/std/rNews
        /2011-10-07#">
      <location></location>
      <formalism>
        <Formalism align:name="OWL1.0" align:uri="http://www.w3
            .org/2002/07/owl#"/>
      </formalism>
    </Ontology>
  </onto2>
  <map>
```

```
    <Cell>
      <entity1>
        <edoal:Class rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#News"/>
      </entity1>
      <entity2>
        <edoal:Class rdf:about="http://iptc.org/std/rNews
            /2011-10-07#Article"/>
      </entity2>
      <relation>=</relation>
      <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">1.0</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1>
        <edoal:Property rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#title"/>
      </entity1>
      <entity2>
        <edoal:Property rdf:about="http://iptc.org/std/rNews
            /2011-10-07#headline"/>
      </entity2>
      <relation>=</relation>
      <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">1.0</measure>
    </Cell>
  </map>
  <map>
    <Cell>
      <entity1>
        <edoal:Property rdf:about="http://ebiquity.umbc.edu/
            ontology/news.owl#description"/>
      </entity1>
      <entity2>
        <edoal:Property rdf:about="http://iptc.org/std/rNews
            /2011-10-07#articleBody"/>
      </entity2>
```

```
        <relation >=</relation >
        <measure  rdf:datatype="http://www.w3.org/2001/XMLSchema#
            float">1.0</measure>
      </Cell>
    </map>
    <map>
      <Cell>
        <entity1 >
          <edoal:Property  rdf:about="http://ebiquity.umbc.edu/
              ontology/news.owl#category"/>
        </entity1 >
        <entity2 >
          <edoal:Property  rdf:about="http://iptc.org/std/rNews
              /2011-10-07#articleSection"/>
        </entity2 >
        <relation >=</relation >
        <measure  rdf:datatype="http://www.w3.org/2001/XMLSchema#
            float">1.0</measure>
      </Cell>
    </map>
    <map>
      <Cell>
        <entity1 >
          <edoal:Property  rdf:about="http://ebiquity.umbc.edu/
              ontology/news.owl#publishedOn"/>
        </entity1 >
        <entity2 >
          <edoal:Property  rdf:about="http://iptc.org/std/rNews
              /2011-10-07#dateCreated"/>
        </entity2 >
        <relation >=</relation >
        <measure  rdf:datatype="http://www.w3.org/2001/XMLSchema#
            float">1.0</measure>
      </Cell>
    </map>
</Alignment>
</rdf:RDF>
```

E

Mapeamentos do Experimento 2

E.1 Mapeamento do Banco de Dados do Joomla com a Ontologia Global

```
@prefix map: <#> .
@prefix db: <> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ
   /0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .
@prefix news: <http://ebiquity.umbc.edu/ontology/news.owl#> .

map:database a d2rq:Database;
   d2rq:jdbcDriver "com.mysql.jdbc.Driver";
   d2rq:jdbcDSN "jdbc:mysql://localhost:3306/joomla";
   d2rq:username "root";
   d2rq:password "";
   jdbc:autoReconnect "true";
   jdbc:zeroDateTimeBehavior "convertToNull";
   .

map:j_content a d2rq:ClassMap;
   d2rq:dataStorage map:database;
   d2rq:uriPattern "j_content/@@j_content.id@@";
   d2rq:class news:News;
```

```
  d2rq:classDefinitionLabel "j_content";
  .
map:j_content__label a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:j_content;
  d2rq:property rdfs:label;
  d2rq:pattern "j_content #@@j_content.title@@";
  .
map:j_content_title a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:j_content;
  d2rq:property news:title;
  d2rq:propertyDefinitionLabel "j_content title";
  d2rq:column "j_content.title";
  .
map:j_content_introtext a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:j_content;
  d2rq:property news:description;
  d2rq:propertyDefinitionLabel "j_content introtext";
  d2rq:column "j_content.introtext";
  .
map:j_content_publish_up a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:j_content;
  d2rq:property news:publishedOn;
  d2rq:propertyDefinitionLabel "j_content publish_up";
  d2rq:column "j_content.publish_up";
  d2rq:datatype xsd:dateTime;
  .
map:j_content_catid a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:j_content;
  d2rq:property news:category;
  d2rq:propertyDefinitionLabel "j_content catid";
  d2rq:column "j_categories.title";
    d2rq:join "j_content.catid => j_categories.id";
  .
```

E.2 Mapeamento do Banco de Dados do WordPress com a Ontologia Global

```
@prefix map: <#> .
@prefix db: <> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ
   /0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .
@prefix news: <http://ebiquity.umbc.edu/ontology/news.owl#> .

map:database a d2rq:Database;
   d2rq:jdbcDriver "com.mysql.jdbc.Driver";
   d2rq:jdbcDSN "jdbc:mysql://localhost:3306/wordpress";
   d2rq:username "root";
   d2rq:password "";
   jdbc:autoReconnect "true";
   jdbc:zeroDateTimeBehavior "convertToNull";
   .

map:wp_posts a d2rq:ClassMap;
   d2rq:dataStorage map:database;
   d2rq:uriPattern "wp_posts/@@wp_posts.ID@@";
   d2rq:class news:News;
   d2rq:classDefinitionLabel "wp_posts";
   .
map:wp_posts__label a d2rq:PropertyBridge;
   d2rq:belongsToClassMap map:wp_posts;
   d2rq:property rdfs:label;
   d2rq:pattern "wp_posts #@@wp_posts.post_title@@";
   .
map:wp_posts_post_title a d2rq:PropertyBridge;
   d2rq:belongsToClassMap map:wp_posts;
   d2rq:property news:title;
   d2rq:propertyDefinitionLabel "wp_posts post_title";
```

```
  d2rq:column "wp_posts.post_title";
  d2rq:condition "wp_posts.post_status = 'publish'";
  d2rq:condition "wp_posts.post_type = 'post'";
  .
map:wp_posts_post_content a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:wp_posts;
  d2rq:property news:description;
  d2rq:propertyDefinitionLabel "wp_posts post_content";
  d2rq:column "wp_posts.post_content";
  d2rq:condition "wp_posts.post_status = 'publish'";
  d2rq:condition "wp_posts.post_type = 'post'";
  .
map:wp_posts_post_date a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:wp_posts;
  d2rq:property news:publishedOn;
  d2rq:propertyDefinitionLabel "wp_posts post_date";
  d2rq:column "wp_posts.post_date";
  d2rq:datatype xsd:dateTime;
  d2rq:condition "wp_posts.post_status = 'publish'";
  d2rq:condition "wp_posts.post_type = 'post'";
  .
map:post_terms a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:wp_posts;
  d2rq:property news:category;
  d2rq:column "wp_terms.name";
  d2rq:condition "wp_posts.post_status = 'publish'";
  d2rq:condition "wp_posts.post_type = 'post'";
  d2rq:join "wp_posts.ID <= wp_term_relationships.object_id";
  d2rq:join "wp_term_relationships.term_taxonomy_id => wp_terms
     .term_id";
  .
```