Manipulação de dados abertos para construção de novas aplicações

Alan Moreira dos Santos

JUIZ DE FORA
JULHO, 2010

Manipulação de dados abertos para construção de novas aplicações

ALAN MOREIRA DOS SANTOS

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharel em Ciência da Computação

Orientador: Jairo Francisco de Souza

JUIZ DE FORA
JULHO, 2010

MANIPULAÇÃO DE DADOS ABERTOS PARA CONSTRUÇÃO DE NOVAS APLICAÇÕES

Alan Moreira dos Santos

MONOGRAFIA SUBMETIDADA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

_____
Jairo Francisco de Souza, orientador.
MSc em Engenharia de Sistemas e Computação COPPE/UFRJ

_____
Regina Maria Maciel Braga Villela
DSc em Engenharia de Sistemas e Computação COPPE/UFRJ

_____
Alessandreia Marta de Oliveira
MSc em Engenharia de Sistemas e Computação COPPE/UFRJ

JUIZ DE FORA, MG - BRASIL
JULHO, 2010

Resumo

A Web, tal como é conhecida hoje em dia, vem sendo gradativamente modificada com a inclusão de padrões estabelecidos por pesquisas na área da Web Semântica. Cada vez mais surgem grupos de desenvolvedores dispostos a melhorar a experiência do usuário final quando a questão é a recuperação de dados de seus repositórios. O conceito de dados ligados abertos como um subgrupo da Web Semântica, tal como sua inclusão no desenvolvimento de aplicações inovadoras, se torna fundamental para propiciar esse tipo de facilidade e interatividade entre usuários e máquinas. Este trabalho apresenta conceitos e ferramentas que possibilitam essa prática e mostra um protótipo que coleta e organiza dados sobre plataformas musicais, como resultado de um estudo de caso.

Palavras-chave: Dados Ligados Abertos, Web Semântica, Ontologias, Repositórios de Dados, Redes Sociais.

Abstract

The Web, as known today, has been gradually changed with the inclusion of standards established by researches in the Semantic Web. Increasingly arise groups of developers willing to improve end-user experience when it comes to recovering data from their repositories. The concept of open linked data as a subset of the Semantic Web, as its inclusion in the development of innovative applications, becomes essential to provide this type of facility and interactivity between users and machines. This paper presents concepts and tools that allow this practice and shows a prototype that collects and organizes data of musical platforms, as a result of a study case.

Keywords: *Linked Open Data, Semantic Web, Ontologies, Data Repositories, Social Networks.*

Agradecimentos

Agradeço imensamente à minha família,
à minha mãe Regina pelo carinho,
ao meu pai José Carlos pela presença amiga,
à minha irmã Aline pela paciência,
à minha vó Therezinha pela ternura,
aos meus amigos pelo apoio incondicional
e ao meu orientador Jairo por toda a atenção e dedicação.

Para Isabella, motivação e inspiração para a realização deste trabalho

.Sumário

Lista de figuras

Lista de tabelas

Lista de listagens

Lista de abreviações

API	Application Programming Interface
DAML	DARPA Agent Markup Language
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JSP	Java Server Page
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RSS	Really Simple Syndication (RSS 2.0)
SQL	Structured Query Language
SPARQL	Simple Protocol and RDF Query Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language for Transformation

1. Introdução

Este capítulo trata da motivação que guiou o desenvolvimento deste trabalho e introduz o cenário mais comum que se conhece hoje em dia quando se lida com informações na Web. A seção 1.1 explora esse tipo de assunto. A seguir, na seção 1.2 são descritos os principais objetivos do trabalho além da apresentação da ideia da aplicação desenvolvida, descrita no último capítulo. A seção 1.3 apresenta a estruturação deste trabalho.

1.1. MOTIVAÇÃO

A Web de documentos é desenhada para consumo humano. Os objetos que compõem esse tipo de Web são documentos que não possuem a característica de se ligarem justamente pela simplicidade de não serem estruturados. Portanto o resultado desse tipo de Web é uma coleção de dados desconectados que dificilmente podem ser integrados de forma mais complexa. Os principais mecanismos de busca atualmente funcionam com a indexação simples por *robots* e *spiders* de palavras-chave registradas em documentos por seus desenvolvedores na forma de *meta tags keywords*. Porém, quando um usuário deseja procurar por alguma informação contextual, que tenha o sentido específico da frase explicitada na busca, ele terá enormes dificuldades de alcançar o resultado esperado.

Outra característica marcante da Web que se conhece hoje em dia é a grande quantidade de redes sociais como **Facebook**[1], **Myspace**[2], **Orkut**[3], **LastFM**[4], **Hi5**[5], **LiveJournal**[6]**,** dentre as mais populares, que permitem aos seus usuários variadas formas de disponibilização de informações pessoais básicas (nome, idade, endereço, estado civil, interesses diversos etc.) e informações mais elaboradas específicas para cada rede social (um histórico de hábitos musicais através das faixas ouvidas pelo usuário no **LastFM**) e o intercâmbio dessas informações, mensagens e atualizações entre seus contatos. Um aspecto ruim das redes sociais é que elas geralmente estão isoladas entre si, ou seja, não é

[1] http://www.facebook.com/
[2] http://www.myspace.com/
[3] http://www.orkut.com/
[4] http://www.last.fm/
[5] http://www.hi5.com/
[6] http://www.livejournal.com/

automática a importação de todos os seus amigos de uma rede para outra assim que se cria um perfil, mesmo que seus amigos tenham perfis nessa nova rede social, ou seja, isso pode se tornar um processo repetitivo que gera uma expressão chamada Social Network Fatigue segundo Passant (2008). Seguindo essa mesma ideia é natural pensar que esse conjunto de informações de um usuário que participa de diferentes redes sociais poderia ser agrupado para que se pudesse inferir as características de um perfil de usuário mais próximas da realidade.

A Web de dados ligados surge como possível solução para esse cenário apresentado objetivando ser um grande banco de dados global, desenhado inicialmente para máquinas. Os objetos primários da Web de dados ligados são as "coisas" ou "*things*" que têm a propriedade de possuírem relações entre si. Nesse caso, a semântica do conteúdo das páginas e suas ligações seriam explícitas e não implícitas como no caso da Web de documentos. A Web de dados ligados também estimula a reutilização de código criado, reduz a redundância e maximiza a interconectividade entre dados (HEATH, 2009).

1.1.1. USUÁRIO COMO CRIADOR DE INFORMAÇÕES

A Web 2.0 trouxe um novo paradigma na forma com que os usuários se relacionam com a informação na Internet (PASSANT, 2008). Logo após o surgimento da Web só existia a percepção de que o usuário fosse apenas um consumidor dessas informações. No entanto, o que acontece atualmente é a existência de um usuário que produz suas próprias informações continuamente. Esse novo tipo de usuário dispõe de uma opção muito grande de cenários e possibilidades para expressar e disponibilizar todo o tipo de material relacionado a ele, sejam suas próprias características na forma de dados pessoais ou suas produções e composições na forma de mídias como imagens, vídeos, áudios e textos com ou sem intenção profissional. Portanto um mesmo usuário de redes sociais pode possuir um perfil no **LiveJournal** em que se concentra na exposição de suas ideias através de *posts* de variados assuntos, um perfil no **Myspace** em que pode manter contato com seus artistas favoritos e informar suas preferências musicais, um perfil no **Flickr**[7] em que monta vários álbuns de fotos com imagens produzidas por ele mesmo e ainda outro perfil no **moviemago**[8] em que interage com usuários, geralmente cinéfilos,

[7] http://www.flickr.com/
[8] http://www.moviemago.com/

para a sugestão e descoberta de filmes interessantes. Seria muito interessante se, quando algum outro usuário visitasse o perfil do **Flickr** do usuário anteriormente citado e desejasse saber que tipo de filmes ele costuma assistir, por exemplo, pudesse ter esse tipo de informação de forma transparente sendo que o primeiro usuário realmente possui um perfil na rede social **moviemago**. Além dessa ligação poderiam existir todas as outras conexões possíveis como por exemplo: dentro do perfil do **Myspace** desse usuário seria possível pesquisar se ele já escreveu uma crítica sobre alguma das bandas listadas em sua preferência em algum dos *posts* do perfil mantido sob o **LiveJournal**. Verifica-se atualmente que todo o cruzamento dessas informações, portanto, é limitado devido à característica de isolamento ou falta de comunicação entre esses portais de informação, principalmente as redes sociais.

1.1.2. REPOSITÓRIOS ESPECIALIZADOS

Existe na Internet um grupo de repositórios de dados especializados capazes de gerarem resultados satisfatórios para consultas limitadas exclusivamente ao estilo de dados armazenado. O repositório **IMDb**[9] é um dos repositórios especializados em cinema mais populares atualmente e ele permite ao usuário obter consultas acerca de todo o universo relacionado a essa indústria como os filmes lançados recentemente, os filmes clássicos, as séries de TV, as companhias que produziram determinado filme assim como as sinopses e resenhas de cada filme ou mesmo a biografia de um determinado ator. O repositório **AMG**[10] referencia-se exclusivamente a dados musicais e oferece ao usuário a opção de consultar as informações gerais de um grupo ou artista, a descoberta de artistas, álbuns e faixas relacionados a algum estilo musical como *Rap, Blues* ou *Electronic Music,* por exemplo, e todo o tipo de informação relevante ao universo musical. Existem também repositórios que estão se adequando ao modelo da Web Semântica no sentido de que parte ou todo o conteúdo de informações estão dispostos no modelo RDF, tecnologia explicada na seção 2.1.1.

Em janeiro de 2010, o governo britânico anunciou o **Data.gov.uk**[11]. O portal é um repositório que reúne dados governamentais e contou com a ajuda de Tim Berners-Lee no seu desenvolvimento. Possui uma quantidade de *datasets* muito maior comparada ao

[9] http://www.imdb.com/
[10] http://www.allmusic.com/
[11] http://data.gov.uk/

serviço similar e pioneiro, o americano **Data.gov**[12]. A intenção do governo da Inglaterra de inovar na disponibilização de dados para acesso público é tão grande que ele lançou um concurso que premia ideias inovadoras sobre a manipulação desses dados de maneira inteligente no site do projeto **Show Us a Better Way**[13]. Outro repositório de dados muito utilizado é o **DBpedia**[14], criado para a disponibilização em formato RDF dos dados da **Wikipedia**[15]. Esses são apenas alguns dos principais repositórios de informações especializadas.

Quando um usuário deseja obter uma consulta que faça a utilização de dados não-específicos a determinado repositório como saber quais são os projetos paralelos de um integrante de uma banda que toca na trilha de um filme por ele pesquisado no **IMDb** ou qual a proximidade geográfica de onde foram gravadas as cenas do mesmo filme, de sua própria casa, ele certamente terá uma enorme dificuldade de conseguir os resultados de forma direta e não-manual, isso é, sem que tenha que percorrer cada repositório separadamente.

1.1.3. INTERFACES DE CONSULTA

A forma em que são realizadas as buscas de dados na Web ainda falha no sentido de que não oferece suporte ao nível de contextualização das palavras inseridas no campo de busca almejado pelo usuário que realiza a consulta.

> Os motores de busca, são ferramentas que dão atenção especial aos metadados nas páginas da Web, pois estes os adicionam aos seus índices em suas bases de dados. Estes metadados são de grande importância para as mais avançadas ferramentas de buscas como o Google, que não só avalia a ocorrência de palavras-chave em uma página, mas também os números de links existentes para outra página na Web.
>
> Estes mecanismos de busca se esforçam menos com relação ao seres humanos, mas ainda estão aquém de produzir índices eficientes que facilitam a busca. Em qualquer solicitação as estes motores de busca é retornado informações que não se relacionam com o contexto ou ainda que simplesmente não fazem parte do significado desta informação.
>
> SALES, 2008.

[12] http://www.data.gov/
[13] http://www.showusabetterway.co.uk/
[14] http://dbpedia.org/
[15] http://www.wikipedia.org/

Atualmente existem aplicações que se apóiam nos padrões oferecidos pela Web semântica para oferecerem melhores soluções aos usuários finais, ainda assim a maioria desses serviços estão limitados ao seu próprio conjunto de dados. Algumas delas estão descritas a seguir.

O **Yahoo! Search Monkey**[16] representa o primeiro de uma chamada nova geração de motores de busca capazes de trabalhar sob aspectos da Web Semântica. A característica principal dele é a de permitir que desenvolvedores customizem seu funcionamento com a adição de suas próprias aplicações ao serviço. A documentação do **SearchMonkey** determina que os desenvolvedores devem possuir noções de XSLT e XPath para criarem serviços personalizados de dados. O **SearchMonkey** recomenda aos proprietários de site que desejam ter seu produto incluído em sua busca que sigam algumas regras que facilitem a captura e interpretação dos dados pelo *crawler* do **SearchMonkey**. Essas regras incluem a utilização de técnicas de representação de dados através de *Microformats*, RDFa e eRDF. O **Yahoo!SearchMonkey** apresenta em uma galeria um conjunto variado de componentes úteis que enriquecem a experiência de busca do usuário.

O **Powerset**[17] é uma aplicação capaz de fornecer resultados interessantes para consultas feitas em linguagem natural, como por exemplo, “Quais os filmes em que atuou Robert de Niro?”. Esses resultados são considerados muito melhores do que os apresentados pelos motores de busca comuns. A tecnologia utilizada para realizar essas consultas é a transformação de uma linguagem humana em consultas sofisticadas apoiadas por SPARQL em *datasets* como o **DBpedia**. Infelizmente o código não é disponibilizado para averiguação ou expansão. A sede da **Powerset** é em São Francisco e a empresa foi fundada em 2005 financiada por **Foundation Capital**, **Founders Fund**, **Paperboy Ventures** e outros investidores. Em 1° de agosto de 2008, a **Powerset** foi adquirida pela **Microsoft**.

O **BooRah**[18] é um portal que oferece um mecanismo de busca de restaurantes americanos em relação à localização escolhida como filtro pelo usuário da pesquisa. O usuário pode se basear no tipo de restaurante que deseja, por exemplo, para saber quais são as melhores pizzarias de Nova York basta digitar no campo de pesquisa a palavra "pizza" e informar "New York" como a cidade no campo de filtragem (este campo de filtragem pode ser preenchido com um endereço, um código postal ou um nome de alguma cidade

[16] http://developer.yahoo.com/searchmonkey/
[17] http://www.powerset.com/
[18] http://www.boorah.com/restaurants/

americana). Os resultados obtidos são exibidos num quadro e pode-se observar a localização desses restaurantes, no caso pizzarias, na janela "Map", devido à integração com a ferramenta **Google Maps**[19]. O portal disponibiliza também um sistema de classificação para a ordenação dos resultados de uma pesquisa. Alguns restaurantes oferecem a opção de reservas on-line através da ferramenta **BooRah**.

O **Uptake**[20] é um sistema de busca para locais de viagem que faz recomendações de acordo com o tema da viagem pretendido pelo usuário, como por exemplo, viagem a negócios, viagem com a família, viagem romântica, viagem turística, entre outros temas e também pelo número de pessoas que irão fazer parte da viagem. Após o usuário definir suas preferências, o **Uptake** se encarrega de se conectar a 5000 sites de viagem para obter os melhores resultados. O sistema permite também a reserva dos locais escolhidos pelo usuário (PALMISANO, 2009).

1.2. OBJETIVOS

Este trabalho tem como objetivo destacar como podem ser criadas aplicações que saibam usufruir da ampla gama de informações heterogêneas na Web (*posts* de *blogs*, páginas da *wiki*, comunidades e perfis de usuários em redes sociais etc.) através da aplicação dos padrões consolidados pela Web Semântica e a utilização de dados ligados. A possibilidade de criação dessas aplicações se apóia também na ampla variedade de *frameworks* disponíveis para a manipulação de dados que seguem os padrões citados, principalmente o *framework open source* SESAME, utilizado para a construção da aplicação adjunta deste trabalho.

Outro objetivo fundamental é apresentar as tecnologias envolvidas na evolução da Web que se conhece atualmente e também explorar devidamente o conceito de dados ligados.

1.3. ORGANIZAÇÃO DO TRABALHO

O trabalho está organizado da seguinte maneira. Após a análise introdutória do cenário da Web que se conhece atualmente feita no capítulo 1, são apresentados os

[19] http://maps.google.com/
[20] http://www.uptake.com/

principais padrões de representação e manipulação de informação consolidados pela Web Semântica e alguns dos principais *frameworks* para construção de aplicações baseadas em dados ligados no capítulo 2. O capítulo 3 fala dos principais projetos que utilizam as tecnologias propostas no capítulo 2, além da introdução do conceito de dados ligados. O capítulo 4 mostra um estudo de caso dos conceitos anteriores, destacando a aplicação construída para este trabalho. O capítulo 5 encerra este trabalho com algumas considerações finais sobre o tema.

2. Manipulação de dados na Web

Ao longo dos últimos anos, vários padrões foram propostos para manipulação de dados disponíveis na Web. A consolidação desses padrões cria facilidades e oportunidades para o desenvolvimento de sistemas inovadores. Neste capítulo, são abordados alguns dos padrões mais usados para manipulação de dados semânticos. Primeiramente, na seção 2.1, são apresentadas algumas linguagens de representação do conhecimento em formato interpretável por aplicativos. Na seção 2.2 é apresentada a linguagem padrão para a técnica de anotação, que significa agregar conteúdo semântico aos documentos XHTML, e a principal linguagem para consulta de dados semânticos, o SPARQL. Finalmente, na seção 2.3, são apresentados alguns dos *frameworks* mais utilizados para a manipulação de dados semânticos.

2.1. PADRÕES PARA REPRESENTAÇÃO DE INFORMAÇÃO

Para possibilitar a introdução dos conceitos de Web Semântica no modelo da Internet foi proposto, por Tim Berners-Lee, uma arquitetura em camadas conhecida como “bolo de noiva” (BREITMAN, 2005). A criação dessa arquitetura parte do pressuposto de que as melhorias na Web devem ser realizadas de forma gradativa, respeitando os padrões já existentes. A figura 2.1 demonstra a forma do modelo proposto.

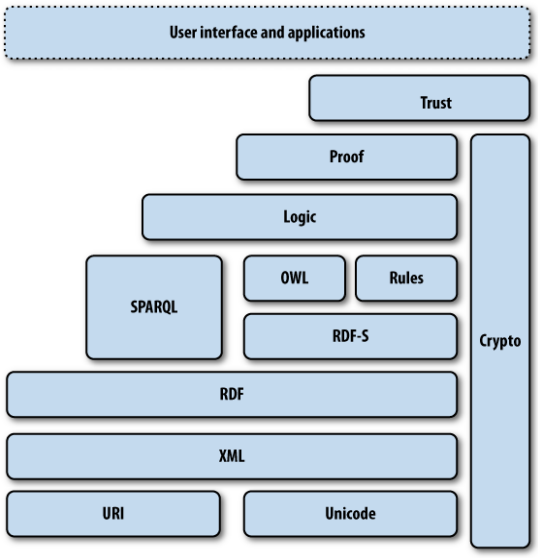

Figura 2.1: Arquitetura “bolo de noiva” (SEGARAN, 2009)

A primeira camada, o HTML (URI + Unicode), é considerada a mais simples. Quando foi criada, essa linguagem possuía apenas a intenção de fornecer informação aos seres humanos por meio de textos, imagens e *hiperlinks* e não se preocupava com a estruturação do seu conteúdo. Para que isso pudesse ser feito, houve a necessidade da criação do XML[21], uma linguagem que possibilita a estruturação dos textos e facilita o intercâmbio de documentos de dados na rede.

Na terceira camada aparece o RDF[22], linguagem que permite descrever o modelo de dados em nível mais semântico que a linguagem XML. Suas principais características são a capacidade de descrever recursos na rede segundo o conceito de metadados. Ao descrever metadados é possível descrever informações detalhadas sobre os dados, o que possibilita o melhor entendimento desses dados por pessoas e máquinas. O RDF está descrito na seção 2.1.1.

O RDFS[23] surgiu como uma extensão do RDF, na camada superior, para aumentar a expressividade do RDF. O RDFS oferece como primitivas de modelagem a construção de hierarquias, classes, propriedades, subclasses e subpropriedades. O RDFS está descrito na seção 2.1.2.

Na tentativa de criar extensões baseadas em RDFS que permitissem descrever mais semântica para os dados, foram propostas ainda algumas linguagens ao longo dos últimos anos, onde as mais conhecidas são DAML, Oil e OWL. A OWL, por ser a linguagem resultante de uma série de evoluções e devido a sua importância fundamental na construção da Web Semântica será descrita com mais detalhes na seção 2.1.3.

2.1.1. RDF

O RDF é uma linguagem que surge como uma contribuição na representação de dados semânticos pois auxilia na eliminação de ambiguidades, aspecto fundamental quando se considera que a interpretação é feita por máquinas.

O RDF é construído sobre três princípios fundamentais: sujeitos, predicados e objetos. Sujeitos ou recursos são representados por URIs ou identificadores universais que referenciam entidades do mundo real. Os identificadores universais são chaves fortes que garantem a consistência do modelo em todos os relacionamentos. É de fundamental

[21] http://www.w3.org/XML/
[22] http://www.w3.org/RDF/
[23] http://www.w3.org/TR/rdf-schema/

importância observar que URIs não são URLs. URIs estão num contexto mais generalizado, o que permite a afirmação de que toda URL é uma URI porém nem toda URI é uma URL.

Predicados ou propriedades definem relacionamentos entre recursos ou entre um recurso e um valor literal. O RDF utiliza URIs também para as propriedades. Assim as propriedades são nomeadas de forma global e única.

Objetos podem ser representados por URIs ou como literais. O RDF utiliza valores literais para representar nomes, endereços, datas, telefones e quaisquer outros tipos de dados que assumem valores. A definição de tipos é feita com o auxílio da especificação XML Schema. O idioma do literal também pode ser definido.

Por exemplo, quando se quer definir que o recurso **http://musicbrainz.org/artist/32b90c92-9978-4a07-90eb-caa4b22f4907** deva ser conhecido pelo nome “Black Rebel Motorcycle Club”, pode-se definir a sentença como uma tripla (http://musicbrainz.org/artist/32b90c92-9978-4a07-90eb-caa4b22f4907, foaf:name, “Black Rebel Motorcycle Club”)

A tripla (x, P, y) pode ser interpretada através de uma fórmula lógica P(x, y) indicando que o predicado **P** relaciona dois objetos **x** e **y**.

A linguagem RDF apresenta um vocabulário XML que permite essa representação em tripla, o que permite o processamento de informações por máquinas. A listagem 2.1 apresenta a tripla em formato RDF.

```
<mo:MusicGroup
rdf:about="http://musicbrainz.org/artist/32b90c92-9978-4a07-
90eb-caa4b22f4907">
        <foaf:name>Black Rebel Motorcycle Club</foaf:name>
</mo:MusicGroup>
```

Listagem 2.1: Tripla representada em formato RDF

O RDF, em síntese, é uma linguagem que oferece subsídios para a representação de objetos e seus relacionamentos, porém não pode ser considerada uma linguagem de ontologias, ou seja, não possui expressividade necessária para a descrição de vocabulários. Para a satisfação dessa necessidade criou-se o RDFS ou RDF-Schema, linguagem apresentada na seção seguinte.

2.1.2. RDFS

O RDFS permite a construção de vocabulários, isso significa que uma comunidade pode criar novas classes e propriedades inerentes à aplicação gerando esquemas de acordo com a sua necessidade.

O RDFS fornece um arcabouço no qual é possível descrever as classes e as propriedades.

Com o RDFS é possível modelar ontologias simples, mas ainda não é uma linguagem expressiva o suficiente para a modelagem de ontologias de aplicações mais complexas justamente por não oferecer conectivos lógicos, negação, disjunção e conjunção.

A listagem 2.2 mostra a construção de uma simplificada ontologia de guitarras que tem por base uma ontologia fictícia de instrumentos.

```
<?xml version=”1.0”?>
<rdf:RDF
xmlns:rdf=”http://www.w3.org/1999/02/22-rdf-syntax-ns#”
xmlns:rdfs=”http://www.w3.org/2000/01/rdf-schema#”
xml:base=”http://www.instrumentos.org/instrumentos”>

<rdfs:Class rdf:ID=”guitarra” />

<rdfs:Class rdf:ID=”guitarra_blues”>
      <rdfs:subClassOf rdf:resource=”#guitarra” />
</rdfs:Class>

<rdfs:Class rdf:ID=”guitarra_jazz”>
      <rdfs:subClassOf rdf:resource=”#guitarra” />
</rdfs:Class>

<rdfs:Property rdf:ID=”marca”>
      <rdfs:domain rdf:resource=”#guitarra” />
      <rdfs:range rdf:resource=”&rdf;literal” />
</rdfs:Property>
</rdf:RDF>
```

Listagem 2.2: Ontologia de guitarras em RDFS

Neste exemplo é definida uma classe **guitarra** através da classe essencial do RDFS: **rdfs:Class**. Em seguida são definidas mais duas classes da ontologia: **guitarra_blues** e **guitarra_jazz**. Essas classes são modeladas como subclasses da classe **guitarra** através da classe do RDFS **rdfs:subClassOf**. É definida também uma propriedade da ontologia: **marca** através da classe do RDFS **rdfs:Property**. Uma

propriedade em RDFS é definida em relação ao seu domínio e alcance, **rdfs:domain** e **rdfs:range**, respectivamente. No exemplo, a propriedade **marca** tem por domínio a classe **guitarra** e por alcance valores literais.

2.1.3. OWL

A OWL[24] é uma linguagem para descrição de ontologias utilizando a estrutura sintática do XML para denotar construtores de um dialeto de lógica de descrição. A OWL atende às necessidades de construção de ontologias (criação e explicitação de conceitos e propriedades), explicitação de fatos sobre um domínio e racionalização sobre as ontologias e fatos relacionados. A OWL é dividida em três linguagens que são classificadas de acordo com a capacidade expressiva de cada uma, conforme descrito a seguir.

A **OWL Lite** suporta a criação de ontologias mais simples com suas hierarquias de classificação e restrição. OWL Lite suporta a modelagem da cardinalidade entre elementos. A **OWL DL** suporta a implementação de lógica descritiva (DL é o acrônimo para "Descriptive Logic"). E **OWL Full** é a linguagem que suporta segundo McGuiness, citado por Breitman (2005) "o máximo de expressividade enquanto mantém completude computacional (para todas as computações se garante tempo finito)".

A listagem 2.3 mostra como é formalizada uma ontologia através da linguagem OWL.

```
<rdf:RDF
xmlns="http://www.w3.org/TR/2003/WD-owl-guide-00000000/band#"
xmlns:band ="http://www.w3.org/TR/2003/WD-owl-guide-
00000000/band#"
xml:base ="http://www.w3.org/TR/2003/WD-owl-guide-
00000000/band#"
xmlns:art ="http://www.w3.org/TR/2004/REC-owl-guide-
20040210/artist#"
xmlns:owl ="http://www.w3.org/2002/07/owl#"
xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs ="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">

<owl:Ontology rdf:about="">
<rdfs:comment>Ontologia simples de uma banda</rdfs:comment>
<owl:priorVersion>
      <owl:Ontology rdf:about="http://www.w3.org/TR/2003/WD-
owl-guide-00000000/band"/>
</owl:priorVersion>
```

[24] http://www.w3.org/2004/OWL/

```
<owl:imports rdf:resource="http://www.w3.org/TR/2003/CR-owl-
guide-00000000/artist"/>
<rdfs:label>Band Ontology</rdfs:label>
</owl:Ontology>

<owl:Class rdf:ID="Band">
      <rdfs:subClassOf rdf:resource="art:Group"/>
      <rdfs:subClassOf>
      <owl:Restriction>
            <owl:onProperty rdf:resource="#numberOfMembers"/>
            <owl:minCardinality
rdf:datatype="xsd:nonNegativeInteger">2</owl:cardinality>
      </owl:Restriction>
      </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="MusicalInstrument">
      <owl:disjointWith rdf:resource="#Band"/>
</owl:Class>

<owl:DatatypeProperty rdf:ID="numberOfMembers">
      <rdfs:domain rdf:resource="#Band"/>
      <rdfs:range rdf:resource="xsd:positiveInteger"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="yearOfCreation">
      <rdfs:domain rdf:resource="#Band"/>
      <rdfs:range rdf:resource="xsd:positiveInteger"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="style">
      <rdfs:domain rdf:resource="#Band"/>
      <rdfs:range rdf:resource="xsd:string"/>
</owl:DatatypeProperty>

<Band rdf:ID="The+Beatles">
      <yearOfCreation
rdf:datatype="xsd:positiveInteger">1960</yearOfCreation>
</Band>

</rdf:RDF>
```

Listagem 2.3: Ontologia de uma “Banda” em OWL

As duas primeiras declarações identificam o *namespace* da ontologia. A primeira declaração explicita o *namespace default* e na segunda observa-se o prefixo “band” que refere-se a uma ontologia sobre bandas.

O prefixo “base” identifica a URI base para a ontologia. O prefixo “art” referencia-se a outro vocabulário do qual conceitos foram importados, um vocabulário fictício de artistas. Em seguida aparece a declaração formal de OWL na linha em que se encontra o prefixo “owl”. Em seguida são definidos os prefixos “rdf”, “rdfs”, “xsd” que

referem-se às respectivas camadas inferiores que também são necessárias para a criação da ontologia. Após a definição dos *namespaces* inclui-se sob a etiqueta **owl:Ontology** alguns metadados referentes ao documento. Essa parte é chamada de cabeçalho da ontologia e contém alguns marcadores para controle de versão, inserção de comentários sobre a ontologia, importação de outras ontologias e ainda o nome da ontologia (BREITMAN, 2005).

Depois de todas as definições iniciais a ontologia começa a ser devidamente modelada com suas classes, propriedades e restrições. No exemplo verifica-se a criação de uma classe chamada “Band”. Essa classe é subclasse da classe “**art:Group**” da ontologia de artistas. Pode-se também definir restrições como a definição de uma cardinalidade mínima para o número de integrantes de uma banda (no caso, dois membros). São definidas três propriedades para essa ontologia: número de membros da banda, ano de criação e estilo da banda. Cada propriedade, assim como em RDFS, possui seu domínio (no caso, a classe “Band”) e seu alcance de valores (no caso, as duas primeiras propriedades com valores numéricos e a última com valor *string*).

A OWL aumenta a expressividade principalmente quanto à sua capacidade de anotar restrições como a restrição de cardinalidade utilizada no exemplo. Além disso, a OWL dá suporte à criação de inferências lógicas que implicam em uma maior corretude da modelagem da ontologia como, por exemplo, a propriendade **disjointWith** que explicitamente diz que um instrumento musical não pode compartilhar instâncias com uma banda pois são objetos completamente distintos. Isso evitaria conflitos futuros em que um desenvolvedor pudesse equivocadamente gerar uma relação de subclasse entre um instrumento e uma banda, por exemplo. Outro ponto interessante é que em OWL é possível explicitar indivíduos da ontologia como no caso da banda **The**+**Beatles**. Poderiam ser explicitadas ainda relações de transitividade e simetria entre classes, o que seria impossível com o RDFS.

2.2. PADRÕES PARA ANOTAÇÃO E MANIPULAÇÃO DE DADOS SEMÂNTICOS

A seguir está descrita a linguagem RDFa como a mais utilizada para anotar documentos XHTML semanticamente. A técnica propiciada pelo RDFa permite aos desenvolvedores de *websites* explicitarem relações semânticas entre o conjunto de

elementos contidos num documento XHTML dentro do próprio documento. Isso faz com que as páginas ganhem mais significado e consigam expressar sentido quando lidas principalmente por *crawlers* ou *robots* que buscam por informações semânticas na Web. Existem outras técnicas para anotar conteúdo semântico que não serão detalhadas. Também será descrita a principal linguagem para consulta de dados que se relacionam obedecendo o padrão de triplas semânticas, o SPARQL. Existem outras linguagens que assemelham-se ao SPARQL em estrutura mas não têm a mesma amplitude de disseminação por serem muito específicas a determinados *frameworks* que manipulam dados semânticos como RDQL, iTQL, Versa, XUL e Adenine, por isso não serão descritas.

2.2.1. RDFa

RDFa é uma especificação de um vocabulário XML criado para permitir adicionar anotações semânticas ricas a um documento XHTML[25] partindo do princípio de que o conteúdo só será publicado uma única vez reunindo códigos que podem ser interpretados por humanos e por máquinas. O RDFa reúne uma série de atributos em XML para especificar o significado do conteúdo existente na forma de XHTML.

Para a descrição de sujeitos, predicados e objetos o RDFa utiliza URIs compactas chamadas CURIEs. CURIEs permitem construções do tipo **example:cow** e **example:places/barn** para gerar as URIs completas **http://example.org/farm/cow** e **http://example.org/farm/places/barn**, respectivamente, algo que não poderia ser feito com os XML *Qualified Names* por causa da barra (*/*), proibida nessa parte da notação (expressão à frente dos dois pontos (**:**)). Uma característica que pode gerar um problema de ambiguidade acontece quando o interpretador de RDFa encontra em algum atributo uma construção do tipo **http:**, uma vez que o interpretador não consegue identificar se a referência é para o protocolo HTTP ou para um CURIE. Para que isso seja evitado, utiliza-se colchetes para envolver o CURIE, **[example:place/barn]**.

A listagem 2.4 exemplifica como é possível mesclar o conteúdo de uma página HTML com informações semânticas providas pelo RDFa. A listagem 2.4 possui a descrição de uma faixa de um determinado artista identificado pelo seu ID universal da base de dados **MusicBrainz** com a utilização da ontologia de música **Music Ontology**.

[25] http://www.w3.org/TR/xhtml11/

```
<ol xmlns:mo="http://purl.org/ontology/mo/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <li typeof="mo:Track"
about="http://www.bbc.co.uk/programmes/p002gpjx#track">
    <div class="summary">
      <h4>
        <span rel="foaf:maker">
          <a title="See more artist information for The Kinks"
href="http://www.bbc.co.uk/music/artists/17b53d9f-5c63-4a09-
a593-dde4608e0db9" typeof="mo:MusicArtist"
about="http://www.bbc.co.uk/music/artists/17b53d9f-5c63-4a09-
a593-dde4608e0db9#artist">
        <span class="artist" property="foaf:name">The
Kinks</span></a></span> – <span class="track"
property="dc:title">Waterloo Sunset</span>
      </h4>
      <span class="release-title" rev="mo:track"><span
typeof="mo:Record"><span property="dc:title">The Greatest Hits
Of 1967 (Various)</span></span></span>
     </div>
   </li>
</ol>
```

Listagem 2.4: Documento XHTML anotado com RDFa

Os *namespaces* são definidos dentro da *tag* de lista desordenada “<ol>”. A seguir define-se o tipo da lista, uma **Track** ou “**mo:Track**”, representando a faixa musical. Ainda é possível reconhecer as informações do artista, seu nome (The Kinks), a faixa musical (Waterloo Sunset) e o álbum a que pertence a faixa (The Greatest Hits Of 1967 (Various)).

A listagem 2.5 representa o mesmo exemplo utilizando a notação RDF para ilustrar a comparação das tecnologias.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:mo="http://purl.org/ontology/mo/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xhv="http://www.w3.org/1999/xhtml/vocab#"
  xmlns:xml="http://www.w3.org/XML/1998/namespace"
>

<mo:Record>
    <mo:track>
      <mo:Track
rdf:about="http://www.bbc.co.uk/programmes/p002gpjx#track">
```

```
        <foaf:maker>
          <mo:MusicArtist
rdf:about="http://www.bbc.co.uk/music/artists/17b53d9f-5c63-
4a09-a593-dde4608e0db9#artist">
            <foaf:name>The Kinks</foaf:name>
          </mo:MusicArtist>

        </foaf:maker>
      <dc:title>Waterloo Sunset</dc:title>
      </mo:Track>
    </mo:track>
    <dc:title>The Greatest Hits Of 1967 (Various)</dc:title>
  </mo:Record>
</rdf:RDF>
```

Listagem 2.5: Notação RDF para o mesmo exemplo mostrado em formato RDFa

2.2.2. SPARQL

SPARQL[26] é uma linguagem para realizar consultas sobre grafos em RDF, assim como o SQL é a linguagem que realiza consultas em bancos de dados relacionais. O SPARQL é capaz de encontrar padrões no grafo RDF, atribuir valores a variáveis usadas como coringas em alguma consulta, filtrar resultados e ainda construir novos grafos a partir do modelo disponível durante a pesquisa.

O SPARQL provê quatro tipos diferentes de consultas: SELECT, CONSTRUCT, ASK e DESCRIBE.

A listagem 2.6 é uma consulta em SPARQL que lista as bandas que possuem pelo menos 90% de grau de similaridade com a banda “The Beatles”.

```
PREFIX mo:   <http://purl.org/ontology/mo/>
PREFIX sim:  <http://purl.org/ontology/sim/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT *
WHERE {
   <http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-
e2c3e1d2600d.html> a mo:MusicGroup;
   <http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17-
e2c3e1d2600d.html> sim:link ?blank.
   ?blank sim:relation mo:similar_to;
          sim:level ?level;
          sim:to ?similar_group.
   FILTER (xsd:double(?level) >= 0.9). }
```

Listagem 2.6: Consulta em SPARQL

[26] http://www.w3.org/TR/rdf-sparql-query/

Uma consulta SELECT geralmente se inicia com a definição dos *namespaces* em um bloco de declarações inicializado pelo identificador PREFIX. No exemplo são definidos três prefixos, um para **Music Ontology** (**http://purl.org/ontology/mo/**), outro para uma ontologia de similaridade (**http://purl.org/ontology/sim/**) e o terceiro para a ontologia **FOAF** (**http://xmlns.com/foaf/0.1/**). "SELECT *" significa a seleção de todas as variáveis declaradas na consulta. Essas variáveis são preenchidas de acordo com a satisfação lógica da estrutura sob a cláusula WHERE. A palavra "a" é uma abreviação para "*typeof*" ou "tipo de" (no exemplo uma das regras é que o ID "**http://musicbrainz.org/artist/b10bbbfc-cf9e-42e0-be17**" seja tipo de "**mo:MusicGroup**").

A importância do SPARQL reside no fato de que, sabendo que existe uma série de projetos imbuídos em disponibilizar o conteúdo de seus domínios em formato semântico, não há a necessidade de criação de APIs exclusivas para a recuperação desses dados pois essas redes podem ser pesquisadas através da linguagem SPARQL.

2.3. PRINCIPAIS BIBLIOTECAS PARA MANIPULAÇÃO DE DADOS SEMÂNTICOS

A partir do momento em que houve uma maior consolidação dos padrões de representação semântica dos dados emergiram uma série de projetos que se dedicam continuamente a propor soluções baseadas nessa tecnologia. Os projetos que serão destacados a seguir, **SESAME**, **Jena**, **Redland** e **Mulgara**, têm a característica de serem *open source* garantindo que todo o *framework* possa ser obtido e modificado de acordo com as intenções de desenvolvimento.

2.3.1. SESAME

SESAME[27] é um *framework* de código-aberto escrito em Java que permite a manipulação de dados em formato RDF. A manipulação inclui consultas e construções através da linguagem SPARQL e a persistência desses dados é feita através de mecanismos de bancos de dados. O SESAME possui algumas características que favorecem a sua escolha por desenvolvedores dentre os vários *frameworks* que trabalham com dados

[27] http://www.openrdf.org/

semânticos. Essas características são alto desempenho, a facilidade de instalação, a facilidade relativa de aprendizado da ferramenta e a sua interface de configuração e administração.

Para a instalação do SESAME é necessário possuir o **Java Runtime Environment** (JRE) na versão 5 ou posterior.

O SESAME pode ser usado tanto em aplicações quanto ser configurado como um servidor que possui diversas aplicações conectadas a ele.

A camada de repositório do SESAME torna transparente a utilização do *framework* pelo desenvolvedor abstraindo uma série de operações de baixo nível realizadas pela camada de armazenamento e inferência do SESAME (**Sail**). A camada **Sail** gerencia a persistência e a manipulação dos dados e é configurável para oferecer estilos diferentes de persistência desses dados. A persistência mais básica é a de um grafo construído na memória dinâmica. Quando se deseja recuperar o estado do grafo na próxima instanciação da classe deve-se utilizar o construtor **MemoryStore** que oferece a opção de salvar o grafo interno em arquivo com a passagem de uma *string* com o endereço do arquivo. A arquitetura modular do SESAME ainda oferece componentes do tipo RIO (RDF I/O) que gerenciam a serialização dos arquivos RDF.

A aplicação criada para este trabalho utiliza o *framework* SESAME e estende o exemplo descrito detalhadamente em Segaran (2009). Na seção 4.2 alguns detalhes referentes a utilização do SESAME são oportunamente descritos.

2.3.2. JENA

Jena[28] é um *framework* desenvolvido em Java e embora tenha funcionalidades similares ao Sesame, o Jena possui APIs mais completas. Ele oferece suporte à criação de ontologias usando OWL, suporte a consultas em SPARQL e diversas maneiras de armazenar os dados em grafos internos. Uma característica negativa que o Jena possui em relação ao Sesame é a de ser um *framework* que demanda mais tempo para aprendizado aos desenvolvedores por ser mais complexo (SEGARAN, 2009).

A classe **ModelFactory** é usada para criar diferentes tipos de modelo para representação de dados semânticos. Para criar um modelo RDF simples, em memória, é utilizado o método **ModelFactory.createDefaultModel**. Este método retornará uma

[28] http://jena.sourceforge.net/

instância de um **Model** que será usada para a criação dos recursos que representarão os indivíduos acerca dos quais poderão se fazer asserções futuras. As propriedades podem ser criadas através do método **createProperty**. As asserções são criadas a partir dos objetos da classe **Resource** com o método **addProperty** que recebe como parâmetro um objeto da classe **Property** e outro recurso. Asserções também podem ser criadas a partir da instância do modelo com o método **addStatement**. Neste caso os três parâmetros a serem informados são os componentes de uma tripla semântica, sujeito, propriedade e objeto, instâncias das classes **Resource**, **Property** e **Resource**, respectivamente.

A listagem 2.7 cria um modelo de RDF simples de um relacionamento familiar[29].

```
String familyUri = "http://family/";
String relationshipUri = "http://purl.org/vocab/relationship/";

Model model = ModelFactory.createDefaultModel();

Resource adam = model.createResource(familyUri+"adam");
Resource beth = model.createResource(familyUri+"beth");
Resource chuck = model.createResource(familyUri+"chuck");
Resource dotty = model.createResource(familyUri+"dotty");

Property childOf =
model.createProperty(relationshipUri,"childOf");
Property parentOf =
model.createProperty(relationshipUri,"parentOf");
Property siblingOf =
model.createProperty(relationshipUri,"siblingOf");
Property spouseOf =
model.createProperty(relationshipUri,"spouseOf");

adam.addProperty(siblingOf,beth);
adam.addProperty(spouseOf,dotty);
adam.addProperty(parentOf,edward);

Statement statement =
model.createStatement(adam,parentOf,fran);

model.add(statement);
```

Listagem 2.7: Modelo RDF que representa um relacionamento familiar no Jena

A classe **ModelFactory** dispõe de métodos capazes de listar os componentes de uma asserção de acordo com os parâmetros especificados na chamada de cada um deles. Os métodos principais são **listSubjectsWithProperty** que lista todos os recursos que têm ligação com a propriedade especificada na chamada assim como o método **listObjectsOfProperty** encontra todos os objetos que possuem um sujeito e uma

[29] http://www.ibm.com/developerworks/xml/library/j-jena/

propriedade, informados como parâmetros. É possível também a listagem de todas as asserções que possuem determinada propriedade através da chamada do método **listProperties**. Os resultados são gerados na forma de especializações da classe Iterator do Java como **ResIterator**, **NodeIterator** e **StmtIterator** para recursos, objetos e asserções, respectivamente.

```
ResIterator parents = model.listSubjectsWithProperty(parentOf);

while (parents.hasNext()) {

  Resource person = parents.nextResource();

  System.out.println(person.getURI());
}

NodeIterator moreParents =
model.listObjectsOfProperty(childOf);

NodeIterator siblings = model.listObjectsOfProperty(edward,
siblingOf);

StmtIterator moreSiblings = edward.listProperties(siblingOf);
```

Listagem 2.8: Uso das especializações da classe Iterator no Jena

Uma desvantagem em utilizar o modelo de dados em memória é a perda de dados assim que o sistema é desligado. Por isso o Jena também oferece mecanismos de persistência dos modelos de dados em um sistema de arquivo ou na forma de banco de dados relacionais (o Jena suporta a integração com o PostgreSQL, Oracle e MySQL).

O Jena também oferece suporte a uma linguagem de consulta para RDF chamada RDQL para a construção de consultas complexas. O RDQL, apesar de não ser um padrão formal, é amplamente utilizado em frameworks RDF.

O Jena foi lançado pelo Laboratório de pesquisa em Web Semântica da empresa Hewlett-Packard.

2.3.3. REDLAND

Redland[30] é uma plataforma semântica de código-aberto desenvolvida em linguagem C. O Redland dispõe suas bibliotecas de forma modular, ou seja, o desenvolvedor pode integrar em sua aplicação os elementos do Redland de forma independente. As capacidades que o Redland oferece são: a transposição do formato RDF para diversas formas de serialização incluindo N-Triples, N3, Turtle, e RDFa, consultas aos dados e compatibilidade com as linguagens Perl, PHP, Python e Ruby. O Redland provê o acesso a essas operações através de um terminal de linha de comando chamado **Rapper**.

A arquitetura do Redland é composta de alguns módulos fundamentais para seu funcionamento. A classe **Model** faz o papel de prover a interface de comunicação entre a aplicação principal e o *framework* Redland. Apesar disso a maioria das funcionalidades é realizada por outras classes, principalmente a classe **Storage** que lida com a manipulação de asserções e a classe **Query** que interpreta consultas mais complexas aos dados armazenados. Outra classe muito importante do Redland é a classe **Parser** que traduz alguns tipos de sintaxe para o formato RDF (BECKET, 2002).

A listagem 2.9 mostra a criação de um perfil **FOAF** utilizando uma API do Redland estendida para a linguagem Ruby.

```
require 'rubygems'
require_gem 'rdf-redland'
require 'rdf/redland/constants'
include Redland

foaf = Namespace.new('http://xmlns.com/foaf/0.1')
model = Model.new()
res = model.create_resource()
res.add_property(Redland::TYPE, foaf['Person'].
add_property(foaf['name'],'Alan').
add_property(foaf['surname'],'Santos').
add_property(foaf['title'],'Estudante')
r2 = model.create_resource()
require 'rdf/redland/schemas/foaf'
r2.type = FOAF::PERSON
res.add_property(FOAF::KNOWS, r2)
serializer = Serializer.new()
serializer.to_file('foaf.rdf',model,Uri.new("http://www.alan.co
m/foaf#"))
```

Listagem 2.9: Criação de um perfil FOAF no Redland

[30] http://librdf.org/

Nas primeiras linhas são realizadas as importações necessárias ao projeto. A seguir é guardado o *namespace* do **FOAF** (a URL "**http://xmlns.com/foaf/0.1**") numa variável. Um modelo em memória é gerado através da instanciação da classe **Model**. A seguir um *blank node* é gerado pelo método **create_resource**. A partir desse momento inicia-se a criação de asserções pelo método **add_property**. O Redland dispõe da propriedade nativa **Redland::TYPE** e desse modo é definido que o nó é do tipo **Person** do **FOAF**. A seguir são definidas as outras asserções como nome, sobrenome e uma ocupação. A seguir é gerada uma relação de contato entre nós. Na última linha de código visualiza-se a possibilidade serialização do grafo interno definido com a instância **model** em um arquivo (**foaf.rdf**) sob uma URI (**http://www.alan.com/foaf#**).

2.3.4. MULGARA

Mulgara[31] é um *framework* baseado em linguagem Java desenvolvido pela empresa Kowari e sua funcionalidade é muito semelhante à do SESAME. Uma das características principais do Mulgara é o suporte a diversas APIs J2EE, incluindo a JTA.

O Mulgara possui algumas características importantes que estão descritas no site principal do *framework* dentre as quais são destacadas:

- uso da linguagem iTQL, muito similar a SQL, para a consulta dos dados internos;
- vários modelos ou banco de dados por servidor;
- ampla capacidade de armazenamento de dados;
- suporte a multiprocessadores;
- funciona em arquiteturas de 64 e 32 bits;
- otimizado para não consumir memória principal do sistema;
- suporte completo a transações;
- conectividade com Jena, JRDF, SOAP e SDK;
- portabilidade por funcionar em diversos sistemas operacionais (Windows NT, 2000 e XP, UNIX e Linux, Solaris, Mac OS X e IRIX.

O Mulgara também dispõe de um mecanismo de sessões que impede o bloqueio

[31] http://www.mulgara.org/

durante a leitura de alguns dados simultaneamente sendo que cada sessão, no escopo de uma consulta, é executada em sua própria *thread* após a obtenção de uma imagem inicial, antes da consulta, de todo o banco de dados. A operação de obtenção dessa imagem inicial não prejudica o desempenho do sistema sendo que custa, em tempo, menos de um milisegundo para ser feita, independentemente do tamanho do banco de dados.

A listagem 2.10 mostra como é feita a manipulação interna de um modelo RDF simples[32].

```
//criando o grafo
Graph graph = new GraphImpl();
GraphElementFactory elementFactory = graph.getElementFactory();
//criando elementos
BlankNode subject = elementFactory.createResource();
URIReference subject2 = elementFactory.createResource(new
URI("http://example.org#subject"));
URIReference predicate = elementFactory.createResource(new
URI("http://example.org#predicate"));
URIReference predicate2 = elementFactory.createResource(new
URI("http://example.org#predicate2"));
Literal object = elementFactory.createLiteral("object");
Triple triple = elementFactory.createTriple(subject, predicate,
object);
Triple triple2 = elementFactory.createTriple(subject2,
predicate, object);
Triple triple3 = elementFactory.createTriple(subject2,
predicate, subject);
Triple triple4 = elementFactory.createTriple(predicate,
predicate2, subject);
//inserindo triplas
graph.add(triple);
graph.add(triple2);
graph.add(triple3);
graph.add(triple4);
//consultando o grafo
Triple queryTriple = elementFactory.createTriple(subject2,
null, null);
ClosableIterator queryResult = graph.find(queryTriple);
//criando a consulta em iTQL
String queryText = "select $s $p $o from
<rmi://mysite.com/server1#testModel> where $s $p $o ; ";
ItqlInterpreter interpreter = new ItqlInterpreter(new
HashMap());
Query query = interpreter.parseQuery(queryText);
//executando a consulta
Answer queryResult = session.query(query);
```

Código 2.10: Manipulação de um modelo RDF simples no Mulgara

[32] http://docs.mulgara.org/system/jrdfexamples.html

3. Dados Ligados Abertos

A definição encontrada na página inicial do **Linked Data**[33] conceitua dados ligados como o uso da Web para conectar dados relacionados que não estavam previamente ligados ou usar a Web para diminuir as barreiras na ligação de dados já vinculados usando outros métodos. Mais especificamente, **Linked Data** pode ser compreendido como um subgrupo da Web Semântica em que os dados disponíveis na Internet podem ser codificados e agrupados de modo a prover informações mais complexas e significantes na forma de RDF e URIs (MACMANUS, 2010). As primeiras ideias básicas de dados ligados foram destacadas por Berners-Lee (2006). Ele descreve os quatro princípios fundamentais de dados ligados como:

- todos os itens devem ser referenciados por URIs;
- todas as URIs devem possuir um conteúdo relacionado ao objeto identificado pelas mesmas, comumente um documento RDF;
- quando observar uma propriedade de *link* dentro de um conteúdo RDF, ele deve referenciar outra URI confiável;
- *links* para outras URIs devem ser incluídos no documento RDF para propiciar a descoberta de novas informações pelo usuário.

O foco dos dados ligados é que sejam publicados dados estruturados com o uso de URIs que identificam objetos ou "coisas" sem a preocupação de elaborar ontologias e inferências. Atualmente os dados ligados são descritos utilizando o formato RDF. Isso faz com que o conceito de dados ligados seja mais acessível para sua própria produção e tenha um alcance mais generalizado entre provedores de dados pela Web. Quando se observa mais de perto os vocabulários que compõem a teia semântica dos dados ligados é inteligível a divisão dos tipos de *links* semânticos, que são as propriedades dentro do contexto de ontologia (HAUSENBLAS, 2009):

- grupos relacionados à relações pessoais: **foaf:knows** do **FOAF**;
- tipos de *links* espacias ou de localidade: **foaf:based_near** ou **geo:lat** do

[33] http://linkeddata.org/

vocabulário básico "geo";

- tipos de *link* temporais como o **dc:created** do **Dublin Core** e a propriedade **event:type** da **Event Ontology**;
- *links* que representam estruturas semânticas como **dc:isPartOf** do **Dublin Core**;
- outros tipos como o **scovo:dimension** do **Statiscal Core Vocabulary**.

3.1. PROJETO LINKING OPEN DATA

A tendência de crescimento de sistemas que utilizem Linked Data como base é um fato evidente devido ao contínuo surgimento de *datasets* ao redor da Web. Um *dataset* é um conjunto de dados na Web que possui dados estruturados sob a forma de uma ontologia e pode ser acessado por ferramentas desenvolvidas para inferência de informações como APIs de consulta. Normalmente, o mais interessante a fazer para os desenvolvedores sobre *datasets* é o *mashup* ou cruzamento de informações de *datasets* diferentes gerando uma terceira informação que não poderia ser descoberta caso as consultas fossem realizadas individualmente em cada *dataset*.

O projeto LOD (Linking Open Data) é um esforço conjunto da comunidade W3C SWEO[34] para que os dados da Web atual sejam publicados no contexto de *datasets*, ou centro de dados e com isso, possibilitar a interligação entre esses *datasets*. O projeto se iniciou em 2007 com um modesto conjunto de *datasets* e tem crescido substancialmente em número de participantes desde seu início.

Atualmente, o projeto comporta mais de cinquenta *datasets* diferentes com mais de 2 bilhões de triplas e mais de 3 milhões de *links* entre si. A última versão, mantida por Cyganiak (2009) fornece o seguinte esquema mostrado na figura 3.1:

[34] http://www.w3.org/2001/sw/sweo/

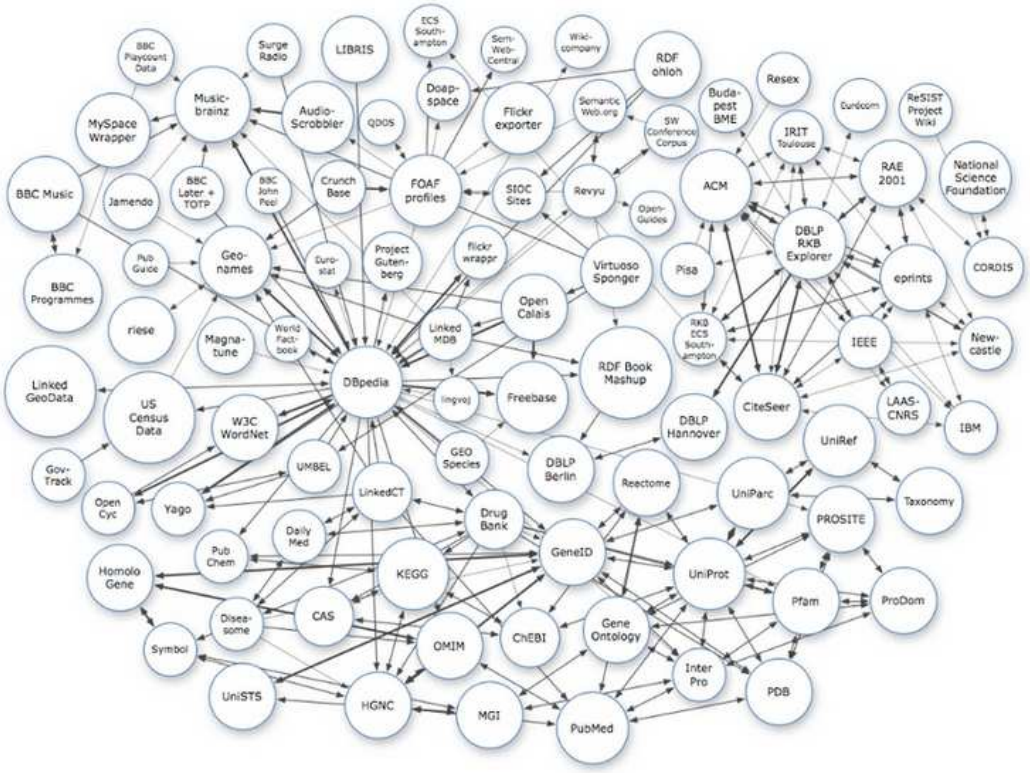

Figura 3.1: Nuvem de dados ligados ou LOD Cloud (CYGANIAK, 2009)

A **LOD Cloud** ou nuvem de dados ligados oferece a oportunidade de desenvolvimento de diversas aplicações utilizando os dados disponíveis por esses vários repositórios contidos na "Cloud", os quais podem ser encarados como diversos bancos de dados de informações públicas disponíveis na Web para serem consumidas por novas aplicações.

Um exemplo de aplicação que poderia ser feita utilizando um subgrafo da nuvem de dados ligados que pode ser observado na figura 3.2, é classificar grupos de pessoas correlacionando seus interesses literários e musicais levando em conta a localização geográfica de cada indivíduo.

A aplicação poderia ser iniciada com a coleta dos dados do repositório **Revyu** que contém críticas de pessoas sobre qualquer tipo de coisas como livros ou músicas. A partir dos perfis **FOAF** desses usuários poderia se obter as preferências literárias e musicais, além da localização espacial de cada pessoa. As informações literárias poderiam ser refinadas com o auxílio das ontologias **RDF Book Mashup**, **DBLP** e **Project Gutenberg** com a mediação do **DBpedia**. Para extrair as informações musicais dos usuários poderia se observar se ele possui algum perfil na rede social **Jamendo** utilizando o seu identificador universal e também refinar essas informações com o **DBTune** e o **MusicBrainz**. Após a

coleta desses dados é possível realizar uma busca por determinada informação estatística como a porcentagem de pessoas com determinada faixa etária em determinado local através da **US Census Data** e da **World Factbook** para verificar se a amostragem de dados da pesquisa está adequada. De posse desses dados poderiam ser feitas diversas consultas que cruzassem esse tipo de informação como “Qual o autor preferido das pessoas que vivem em Nova Orleans?” ou “Cidades próximas se assemelham quanto ao gosto musical de seus habitantes?” ou mesmo “Existe alguma relação entre um estilo musical e um estilo literário predominante em determinada cidade?”.

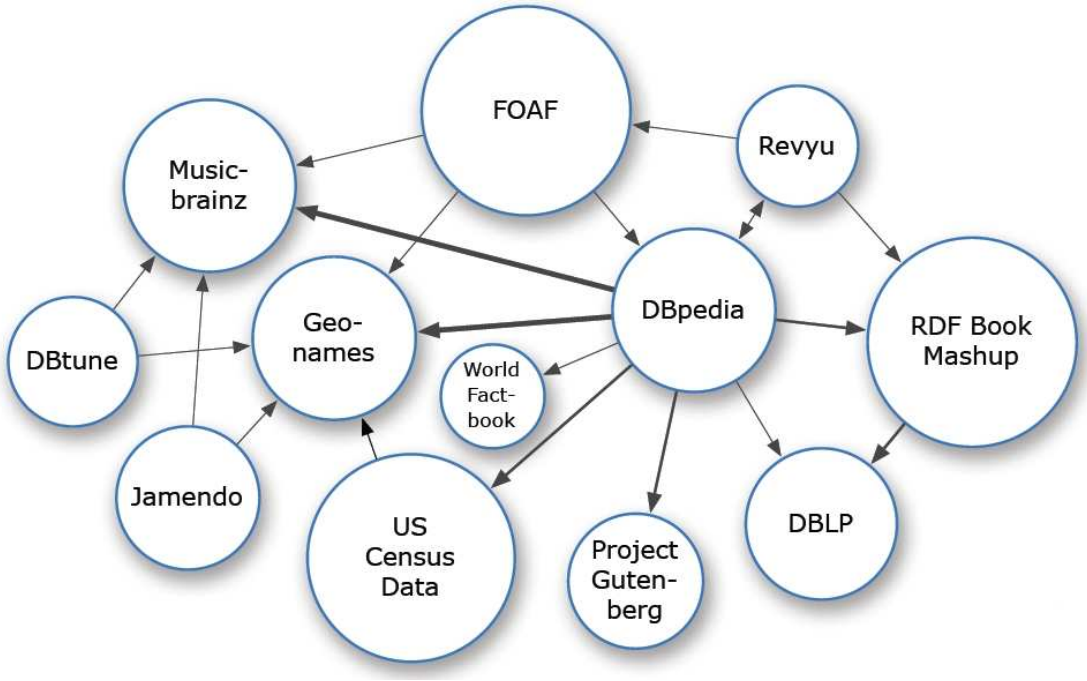

Figura 3.2: Subgrafo da LOD Cloud

3.2. ONTOLOGIAS UTILIZADAS COMO MODELOS PARA LINKED DATA

Segundo Guarino, citado por Carneiro (2005) uma ontologia é “uma maneira de se conceitualizar de forma explícita e formal os conceitos e restrições relacionados a um domínio de interesse”.

Existe uma série de ontologias públicas que foram criadas para descrever objetos do mundo real e seus relacionamentos e com isso, fornecer subsídios de manipulação de dados para aplicações da Web Semântica de forma consistente e desambígua. Essas ontologias, mais adequadamente classificadas como tesauros, modelos de representação

sem o envolvimento de lógica que oferecem somente a listagem dos termos, suas definições e relacionamentos hirárquicos, abrangem contextos de informação bem distintos e na atual configuração da nuvem de dados ligados pode-se observar ontologias referentes a redes sociais como a **FOAF**[35], o **Jamendo**[36], a **Myspace Wrapper**[37], a **flickr wrappr**[38] e a **Audio-Scrobbler**[39], ontologias que concentram informações sobre publicações literárias e/ou acadêmicas como o **RDF Book Mashup**[40], o **DBLP Berlin**[41] e o **Project Gutenberg**[42], ontologias que manipulam dados musicais como a **Surge Radio**[43], **MusicBrainz**[44] e **BBC Music**[45], ontologias que descrevem dados sobre localização geográfica como o **GeoNames**[46] e até ontologias que tratam de dados referentes à medicina como a **Daily Med**[47] e a **Drug Bank**[48].

Ainda existem outros cenários de informação que não foram descritos aqui. Algumas ontologias importantes estão descritas de forma mais detalhada a seguir. É importante destacar que esses *datasets*, além de geralmente possuírem uma ontologia própria, fazem o uso de outras ontologias para denotarem seus dados como, por exemplo, o uso do **FOAF** na descrição das propriedades da rede **Myspace Wrapper**.

3.2.1. FRIEND OF A FRIEND

O **Friend of a Friend** ou **FOAF** possui um extenso vocabulário para representação de informações sobre pessoas, como por exemplo, seu nome, endereço, cidade (localização), figuras, blogs ou páginas pessoais de outros tipos, data de nascimento e principalmente quais são os seus conhecidos.

Há uma série de redes sociais que disponibilizam arquivos **FOAF** de seus usuários para acesso público como **Hi5**, **LiveJournal**, **Flickr**, **LastFM** e **Myspace**.

[35] http://www.foaf-project.org/
[36] http://www.jamendo.com/
[37] http://dbtune.org/myspace/
[38] http://www4.wiwiss.fu-berlin.de/flickrwrappr/
[39] http://dbtune.org/last-fm/
[40] http://sites.wiwiss.fu-berlin.de/suhl/bizer/bookmashup/
[41] http://www4.wiwiss.fu-berlin.de/dblp/
[42] http://www4.wiwiss.fu-berlin.de/gutendata/
[43] http://www.surgeradio.co.uk/
[44] http://dbtune.org/musicbrainz/
[45] http://www.bbc.co.uk/music/
[46] http://www.geonames.org/ontology/
[47] http://www4.wiwiss.fu-berlin.de/dailymed/
[48] http://www4.wiwiss.fu-berlin.de/drugbank/

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">
<foaf:Person rdf:ID="me">
      <foaf:name>Alan Santos</foaf:name>
      <foaf:givenname>Alan</foaf:givenname>
      <foaf:family_name>Santos</foaf:family_name>
      <foaf:nick>alan</foaf:nick>
      <foaf:yahooChatID>shawwhip</foaf:yahooChatID>
</foaf:Person>
</rdf:RDF>
```

Listagem 3.1: Perfil FOAF

O *namespace* do **FOAF**, **http://xmlns.com/foaf/0.1/** define em torno de 50 propriedades diferentes que uma pessoa pode possuir explicitadas em um arquivo.

Quando se obtém um arquivo **FOAF** não se tem somente informação acerca de uma única pessoa, tem-se também as URIs de todos os conhecidos desta pessoa que por sua vez, referenciam os arquivos **FOAF** de cada indivíduo. Isso significa que, com a abordagem adequada, pode-se reproduzir uma teia de relações pela ligação desses documentos.

3.2.2. MUSIC ONTOLOGY

A **Music Ontology** é construída no topo de duas ontologias base: a **Timeline Ontology** e a **Event Ontology**. Esta ontologia também se apóia na ontologia **FRBR** e na ontologia **FOAF**.

3.2.2.1. TIMELINE ONTOLOGY

"Informação temporal é a primeira coisa que queremos expressar quando lidamos com conhecimento relacionado à música" (RAIMOND, 2007).

É uma ontologia capaz de expressar os tipos mais variados de expressão temporal, como datas “a música foi criada em 23 de Março de 1979” e valores do tempo de duração de uma música.

A **Timeline Ontology** define também o conceito **TimeLine** que permite representações do tipo “faixas de tempo”. Exemplo: “Do tempo 5 até o tempo 8 da faixa musical desejo que apareça alguma informação sobre o álbum do artista na forma de uma

caixa de texto.”. A **Timeline Ontology** é modelada com a sublinguagem da OWL, **OWL DL**.

3.2.2.2. EVENT ONTOLOGY

O conceito de Evento definido na **Event Ontology** é descrito como composto de um número de fatores (objetos inanimados como os instrumentos e o palco), agentes (como um artista) e produtos (os autores classificam o som produzido pelo artista como uma espécie de produto). A ontologia de eventos pode ser ligada com a ontologia **GeoNames** sendo que todo evento possui uma localização onde acontece (**event:place**) e também com a **Timeline Ontology** pois cada evento tem seu tempo particular de duração ou data de acontecimento (**event:time**).

A **Event Ontology** também abrange o conceito de subeventos, permitindo que um evento mais complexo, como um festival, possa ser detalhado ou dividido em eventos menores devido à complexidade de mapeamento desse tipo de evento em dados semânticos.

3.2.2.3. FRBR

A **Functional Requirements for Bibliographic Records Ontology** ou **FRBR** objetiva mapear todo o contexto que envolve a produção musical como a composição, performance de um artista, gravação em estúdio de um álbum, masterização de um álbum etc.). A figura 3.3 ilustra um subprocesso derivado desse contexto.

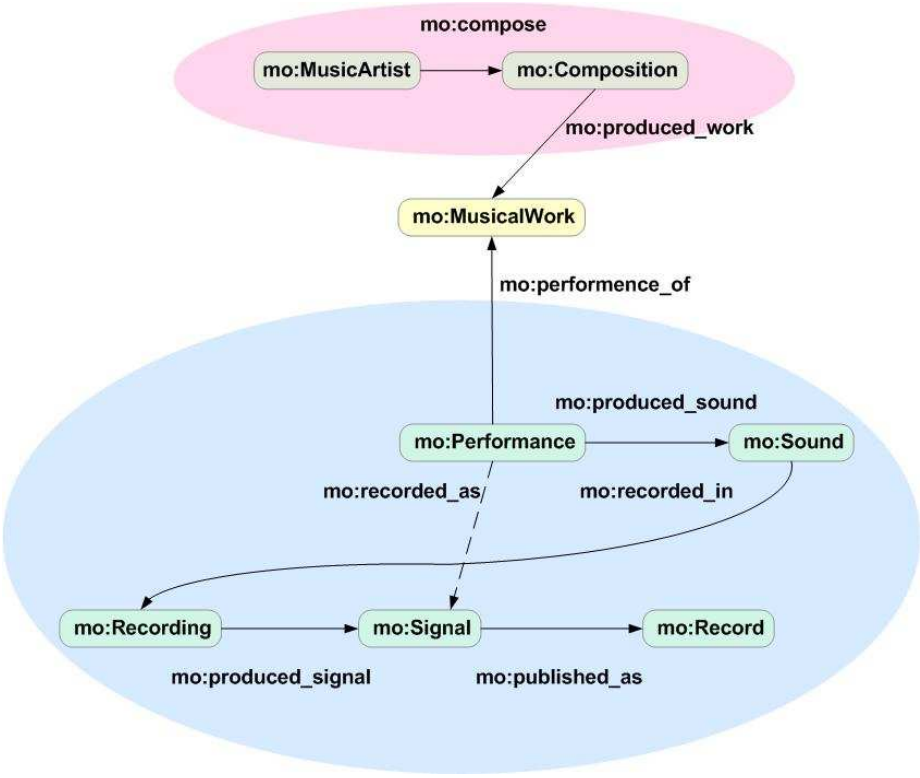

Figura 3.3: Workflow de uma produção musical (RAIMOND, 2007)

O exemplo mostrado pela figura 3.3 relaciona o processo de composição de uma música em que um artista (**mo:MusicArtist**) compõe a música (**mo:Composition**) e a disponibiliza como um trabalho musical (**mo:MusicalWork**) com o processo de gravação e performance dessa composição.

3.2.2.4. NÍVEIS DE EXPRESSIVIDADE

A **Music Ontology**, por ser muito flexível, é dividida em diferentes níveis de expressividade segundo Raimond (2007).

- o primeiro e mais simples somente lida com informações puramente editoriais expressando principalmente a relação entre faixas, álbuns e artistas;
- o segundo nível introduz o conceito de **Evento** que pode ser utilizado para expressar relações entre a composição de uma música, a produção de um álbum e a performance de um músico.
- o terceiro nível de expressividade permite o detalhamento de um evento em

uma forma como “neste show, esta tecla foi tocada nesse determinado tempo por essa determinada pessoa que estava tocando o piano”.

A ontologia oferece suporte aos três tipos de informação musical descritas segundo Pachet, citado por Raimond (2007), que são os metadados editoriais, os metadados culturais (envolve gêneros musicais e informações de redes sociais) e os metadados referentes ao conteúdo das mídias musicais.

3.2.3. MUSICBRAINZ

O **MusicBrainz**[49] é uma comunidade aberta dedicada a coletar e tornar públicos metadados relacionados à música. A coleção gerada pelo **MusicBrainz** ultrapassa os simples moldes de metadados do tipo artista/faixa/álbum. Por exemplo, os dados que podem descrever um artista dentro do **MusicBrainz** são: um **ID MusicBrainz** (**MBID**), o nome, um nome para ordenação, uma lista de nomes mal-digitados ou apelidos que se aproximam do nome original da banda (útil para a desambiguação de pesquisas pois é impossível prever o que será digitado pelo usuário), o tipo (banda/artista solo), a data de criação da banda, a data de término da banda, um campo de comentário, um campo de anotação (editável).

Todas as propriedades que podem ser aplicadas a recursos relacionados ao **MusicBrainz** estão descritos em uma página específica[50] da documentação do **MusicBrainz**. Usuários e desenvolvedores podem usufruir do **MusicBrainz** encontrando todas as músicas de um determinado artista, toda a discografia de um determinado artista e críticas de álbuns específicos

O **MusicBrainz** ainda provê um mecanismo interessante de busca e preenchimento de metadados ausentes em arquivos de música digitais. Qualquer tipo de aplicação com suporte a XML pode realizar consultas ao **MusicBrainz**, visto que ele possui um serviço Web que produz resultados nesse formato ao ser solicitado por uma biblioteca cliente compatível com várias linguagens de programação.

[49] http://musicbrainz.org/
[50] http://musicbrainz.org/doc/MusicBrainz_Database

3.3. APLICAÇÕES QUE MANIPULAM DADOS LIGADOS

Conforme os dados ligados vêm se tornado cada vez mais comuns na Internet, novas aplicação que exploram esses dados têm surgido. Esta seção descreve algumas dessas aplicações.

3.3.1. SISTEMA DE RECOMENDAÇÃO MUSICAL USANDO REDES SOCIAIS

Usando as redes sociais como primeiro passo para criar um método de recomendação musical, os autores propõem uma inferência ao associar ao usuário um artista através da consulta aos seus hábitos utilizando a **Event Ontology**. Esse algoritmo simples existe na maioria dos serviços sociais de música segundo Passant (2008). O *framework* MOAT permite que as pessoas marquem o conteúdo de suas redes sociais com URIs em vez de simplesmente utilizar palavras-chave do tipo *string* graças à aplicação **LODr** (**Linking Open Data Tagging System**). Uma vez que as pessoas façam isso é possível obter informações relacionadas àquele objeto marcado.

O exemplo utilizado no artigo mostra uma foto de um **Flickr** de um usuário qualquer. A partir do momento em que o usuário identifica a foto com uma tag URI, a imagem passa a expressar um significado mais consistente. No caso, a foto é de **Joe Strummer**, vocalista da banda **The Clash**, o que é inferido pelo **DBpedia** que relaciona uma banda e seus membros. Sendo assim, caso um outro usuário de outra rede, como a **Drupal**, por exemplo, tenha alguma referência sobre a banda **The Clash**, identifica-se uma relação de preferências entre os dois usuários.

Usar **Linked Open Data** para formular recomendações pode ser um caminho de várias estratégias. Os autores dizem que foi feita uma pesquisa entre 400 páginas de artistas randômicas para a observação de quais seriam as propriedades mais comuns de dados. As 20 propriedades mais populares são listadas na figura 3.4.

Rank	Property	Number of relationships
1	skos:subject	1930
2	rdf:type	882
3	dbpedia:reference	847
4	dbpedia:genre	450
5	dbpedia:page	400
6	dbpedia:hasPhotoCollection	400
7	dbpedia:origin	355
8	dbpedia:wikiPageUsesTemplate	333
9	dbpedia:label	265
10	dbpedia:wordnet_type	194
11	dbpedia:associatedActs	189
12	foaf:homepage	178
13	dbpedia:currentMembers	151
14	dbpedia:url	114
15	dbpedia:pastMembers	108
16	dbpedia:occupation	97
17	owl:sameAs	95
18	foaf:depiction	89
19	foaf:img	89
20	dpbedia:wikipage-de	85

Tabela 3.1: Propriedades mais populares do DBpedia (PASSANT, 2008)

Logo depois houve uma filtragem para decidir quais dessas propriedades seriam realmente relevantes e assim foram excluídas algumas propriedades como **dbpedia:wikiPageUsesTemplate**.

O principal desafio aqui é obter um caminho comum entre artistas diferentes e quando esse caminho for encontrado, decidir se ele é realmente um caminho marcante para definir que um artista é similar ao outro. Por exemplo, é possível sugerir uma banda semelhante a outra sendo que elas tocaram na mesma casa de *show* em datas recentes. Essa poderia ser uma regra dentre tantas outras.

Sistemas de recomendação musical podem ser agrupados em dois tipos: sistemas de filtragem colaborativa ou sistemas baseados no conteúdo. Os sistemas de filtragem colaborativa se apóiam no histórico de informações do usuário e da comparação entre essas informações sempre baseando-se no comportamento do grupo de usuários. O serviço **LastFM** é um desses sistemas. Ele pode, por exemplo, se basear na comparação entre *tags* que costumam ser utilizadas por determinado usuário e sugerir artistas que se aproximam do contexto musical descrito por essa *tag*. Já os sistemas baseados em conteúdos são de baixo nível, ou seja, eles são capazes de extrair informações dos conteúdos musicais de um usuário como, por exemplo, o timbre, o ritmo, a estrutura, a harmonia e a melodia de certa faixa que o usuário costuma ouvir e procurar por faixas semelhantes em seu banco de

dados para fornecer a recomendação.

Também existem os sistemas que agrupam características dos dois sistemas citados anteriormente como o **SIMAC**, projeto descrito a seguir na seção 3.3.2.

3.3.2. SIMAC

O projeto propõe gerar recomendações musicais baseadas tanto nos documentos **FOAF** do usuário quanto nas descrições extraídas dos próprios arquivos de áudio.

Embora exista uma grande variedade de informações musicais na Web como resenhas de álbuns, biografias de artistas, eventos musicais e sua localização, essa informação raramente está diretamente ligada aos objetos a que deveriam se associar, os arquivos de música. O principal objetivo do projeto **SIMAC** é procurar por descritores semânticos dos áudios e gerar uma recomendação musical baseada nessas informações da mesma maneira em que é possível capturar informações que dizem uma série de características dos usuários através do seu perfil **FOAF**.

Os autores citam as diferenças entre a filtragem baseada em colaboração e baseada no conteúdo. A filtragem baseada em colaboração consiste no uso das informações descritas pelos usuários de forma explícita (anotações e classificações) ou implícita (hábitos musicais do usuário, por exemplo). Já a filtragem baseada no conteúdo tenta extrair as informações inerentes aos itens gerados pelo usuário, nesse caso as faixas de áudio. Um grande desafio seria estabelecer medidas de distância entre as faixas de áudio para estabelecer o grau de similaridade entre elas e assim oferecer a recomendação adequada. Os autores afirmam que a adição de metadados culturais auxiliaria nessa proposta. A principal meta desse tipo de serviço é recomendar artistas desconhecidos e quando possível, até suas faixas disponíveis para o usuário final, de acordo com seu gosto musical.

Os autores citam que seria importante elaborar uma maneira de combinar aspectos dos usuários, como idade, gênero e ocupação com suas preferências musicais para melhorar o serviço de recomendação musical. Um serviço de recomendação musical também poderia, de acordo com o artigo, integrar tecnologias utilizadas para divulgação musical, assim como o **iTunes Music Store**[51] gera recomendações através de uma *newsletter* periódica e faz o uso do RSS.

[51] http://www.apple.com/itunes/

A maioria dos sistemas atuais de recomendação musical é baseada na filtragem colaborativa (alguns utilizam abordagens híbridas). Alguns exemplos de plataforma são citados como o **AudioScrobbler**, **iRate**, **Goombah Emergent Music** e **inDiscover**. A falha desses sistemas seria gerar sempre recomendações muito óbvias por se basear na similaridade de faixas escutadas entre usuários.

O protótipo do sistema do artigo propõe uma nova abordagem em que a recomendação é feita levando-se em conta também as informações de dados pessoais, através do **FOAF**. O sistema, desse modo, funciona capturando as informações de interesse do **FOAF** do usuário, no caso é oferecido um arquivo **FOAF** gerado a partir do perfil do **LiveJournal**. Logo a seguir o sistema filtra quais dentre os interesses são artistas ou bandas. No próximo passo o sistema busca por informações de similaridade entre os artistas encontrados para criar um banco de dados interno de recomendação. Esse passo usa um *crawler* que busca dados de portais como **allmusic.com**, **mp3.com** e **msn.music.com**. O sistema ainda procura por informações de artistas pouco conhecidos em alguns portais especializados como: **magnatune.org**, **garageband.com** e **vitaminic.com**.

Finalmente é gerada uma métrica de similaridade entre as faixas tocadas ou compostas pelos artistas encontrados nos interesses do perfil do usuário e as faixas dispostas no banco de dados gerado anteriormente. A métrica de similaridade é baseada em quantificadores como: descritores de ritmo, de tom e timbre (CELMA, 2002).

3.3.3. EASAIER

O **EASAIER** é um portal de recuperação de dados musicais que associa recursos individuais em um contexto semântico. O projeto se apóia no uso da **Music Ontology** e o **Hotbed Mapping**: um banco de dados que contém informações sobre a música tradicional escocesa de acordo com os autores. As informações variam entre os artistas, os instrumentos tocados por eles e as gravações feitas. O projeto gerou dois aplicativos principais descritos a seguir.

O **HOTBED Search** que é um aplicativo que provê uma representação gráfica dos resultados da busca na forma de uma árvore. É possível que o usuário refine constantemente a busca para obtenção de melhores resultados.

E o **MusicBrainz Search** que funciona da seguinte maneira: inicialmente o **MusicBrainz** utiliza o **Openlink Virtuoso RDFView** como a funcionalidade capaz de

mapear seu banco de dados relacional para o modelo de **Music Ontology**. A partir desse momento cada recurso do **MusicBrainz**, seja ele, um artista, um álbum ou uma faixa de áudio são especificados através de um identificador único do **MusicBrainz**, o **MBID**. O **MusicBrainz ID** recupera uma série de informações pertinentes que identificam um artista como: *websites* relacionados, imagens, informação discográfica etc. Adicionalmente o sistema mostra os artistas relacionados ao primeiro. A busca ainda percorre o **DBpedia** para a recuperação de informações que o **MusicBrainz** não possui como: anos em atividade do artista, gênero, a descrição da **Wikipedia** etc.

O **EASAIER** também incorpora conteúdo do **Google**, **Youtube**, **LyricWiki**, **Yahoo Images & Music**, **Amazon**, **eBay**, e **LastFM** (LUGER, 2007).

4. Estudo de caso: integração de redes sociais e informações musicais

A motivação para o desenvolvimento deste aplicativo é mostrar como as informações descentralizadas e variadas existentes na Web podem ser agrupadas e incluídas dentro de um contexto específico, determinado pelo interesse da aplicação desenvolvida. Este capítulo está organizado como segue.

4.1. DESCRIÇÃO DO CENÁRIO DA APLICAÇÃO

A aplicação construída coleta uma série de atributos referentes aos perfis de usuários da rede social **Myspace** e amplia as relações existentes com a exploração dos recursos encontrados no perfil desses usuários com dados de outras bases de dados disponíveis na Web como informações musicais e geográficas. A aplicação pode ser encarada como um *Web Crawler* de dados RDF e visa responder a consultas como “Quais são os próximos shows de determinada banda?”, “Quais os álbuns já lançados por determinado artista do Myspace?” ou mesmo “Onde foi criada determinada banda, geograficamente?”.

A aplicação procura disponibilizar informações extras sobre o grupo de usuários a que um usuário está ligado e permite o aprofundamento sucessivo em redes de outros usuários de acordo com a intenção de pesquisa do utilizador do programa. O sistema permite que o utilizador verifique informações importantes de seus artistas preferidos, geralmente tidos como contatos de **Myspace**, como os próximos eventos, classificados por ordem de aproximação de datas, e quais os álbuns já lançados por eles. Ainda são disponibilizados um *link* que denota a página do perfil desses artistas no **Myspace** e um *link* para um mapa que indica a localização geográfica de cada artista.

4.2. ARQUITETURA DA APLICAÇÃO

Foi utilizada como base de desenvolvimento do trabalho a API do *framework*

SESAME incluída no projeto através do arquivo **openrdf-sesame-2.2.4-onejar**[52]. A classe principal **SimpleGraph**[53] em que são desenvolvidas as operações gerais de manipulação de um grafo interno além do acesso às funções da API do SESAME foi modificada com a inclusão de algumas funções extras de acordo com o objetivo da aplicação.

Cada elemento que compõe a arquitetura da aplicação que pode ser visualizada na figura 4.1 será descrito a seguir.

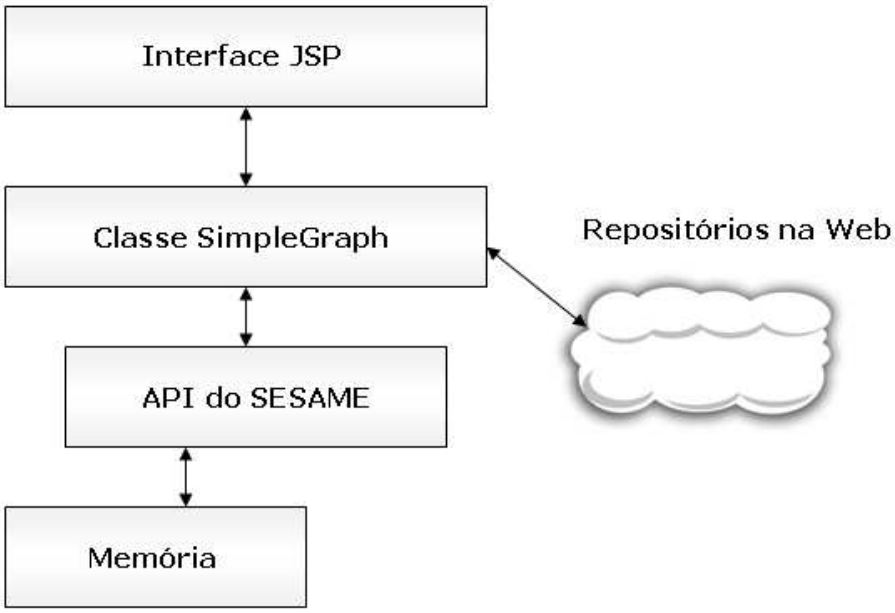

Figura 4.1: Arquitetura da aplicação desenvolvida

A primeira camada da aplicação é a que interage diretamente com o usuário final e foi modelada com a tecnologia *Java Server Pages* (JSP). As páginas JSP contêm, cada uma, uma referência para a classe principal do projeto, **SimpleGraph**, através de **JavaBeans** e possuem *scriplets* que fazem a chamada dos métodos principais da classe **SimpleGraph** para inserir dados no grafo interno e consultá-los posteriormente. Todos os dados consultados são dispostos de forma estruturada segundo o *layout* da aplicação que será mostrado com detalhes na seção 4.4.

A classe, num primeiro momento, recebe as URIs vindas da camada JSP, faz as requisições aos repositórios, recebe os arquivos RDFs provenientes dessas consultas e armazena em memória o conteúdo desses arquivos num grafo de triplas (sujeito,

[52] http://sourceforge.net/projects/sesame/files/Sesame%202/
[53] http://semprog.com/psw/chapter8/SimpleGraph.java

propriedade, objeto). O segundo momento acontece quando a classe recebe alguma consulta SPARQL. Nesse caso existe um método que faz a consulta no grafo interno e devolve uma lista (estrutura de dados **List** do Java) contendo todos os resultados obtidos. Essas são as duas funções principais da classe **SimpleGraph,** prover a negociação HTTP com os repositórios na busca pelos RDFs para inserí-los num grafo interno e fazer consultas SPARQL nesse grafo.

A API do SESAME faz as operações de baixo nível da classe **SimpleGraph** para armazenamento dos dados no grafo interno como reservar as áreas de memória de acordo com a quantidade de dados a ser inserida. No caso desta aplicação, não foram utilizadas as opções de inserção de dados em um banco de dados ou em arquivos, mas para aplicações de grande porte seria fundamental a preservação dos dados em função do desempenho do sistema. Os repositórios utilizados para as consultas são detalhados na seção 4.3.

4.3. INTERLIGAÇÃO DOS MODELOS DE DADOS

A aplicação se conecta a alguns repositórios especializados para a captura das informações mostradas ao usuário final. Esses repositórios são o **DBtune** para **Myspace**, também conhecido formalmente como **Myspace Wrapper** na nuvem de dados ligados, o **lastfm.rdfize** e o **MusicBrainz**.

O **Myspace Wrapper** contém as propriedades mais básicas de cada usuário do **Myspace**. Na aplicação foram usadas as propriedades: nome do usuário, *link* para um mapa na base de dados **GeoNames** segundo a localização do usuário, figura de perfil usada no Myspace, tipo de usuário para identificar quando ele é um artista e o número total de amigos desse usuário.

O **lastfm.rdfize** provê os **eventos** de cada artista com que têm suas próprias propriedades como título**,** data e *homepage* de cada evento. O **lastfm.rdfize** também disponibiliza um identificador para a rede **MusicBrainz**.

A rede **MusicBrainz**, por sua vez, mostra as características de cada artista e em especial os **álbuns** lançados por cada artista com suas propriedades de título e *link* para uma resenha no *site* do **BBC**. As informações sobre os álbuns são utilizadas na aplicação.

O identificador que interliga as redes **Myspace Wrapper** e **lastfm.rdfize** é o nome utilizado para identificação do perfil **Myspace** do usuário. A figura 4.2 mostra os repositórios, suas interligações e as propriedades que cada um fornece à aplicação.

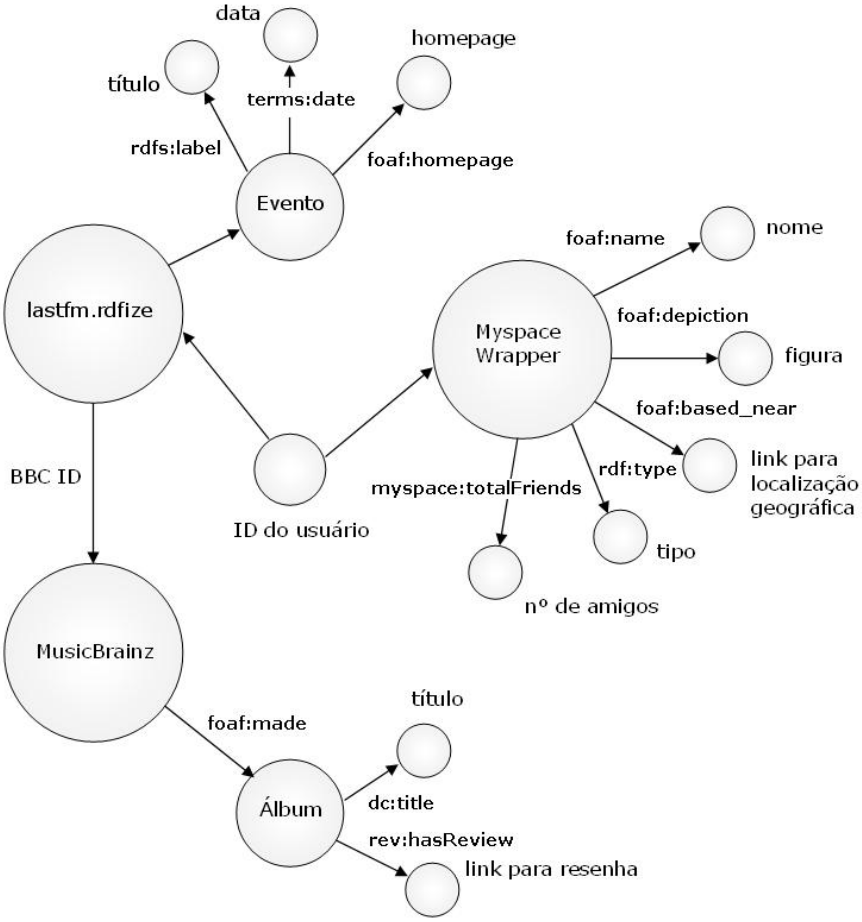

Figura 4.2: Repositórios e propriedades utilizados na aplicação

Na figura 4.2 aparece uma série de *namespaces* que são abreviações para o vocabulário ou sintaxe, usados para cada propriedade específica. Neste caso os vocabulários utilizados foram: **FOAF**[54], **myspace**[55], **dc**[56] ou **Dublin Core**, **rev**[57], e **terms**[58]. As linguagens de sintaxe foram **RDF**, **RDFS** e **OWL**, sendo que a última não aparece na figura 4.2 por não se referir diretamente a alguma propriedade de destaque.

4.4. CONSULTAS NO MODELO DE DADOS

Nesta seção são apresentadas algumas telas do sistema para a demonstração de

[54] http://xmlns.com/foaf/0.1/
[55] http://purl.org/ontology/myspace#
[56] http://purl.org/dc/elements/1.1/
[57] http://purl.org/stuff/rev#
[58] http://purl.org/dc/terms/

seu funcionamento. A tela de apresentação é mostrada na figura 4.3.

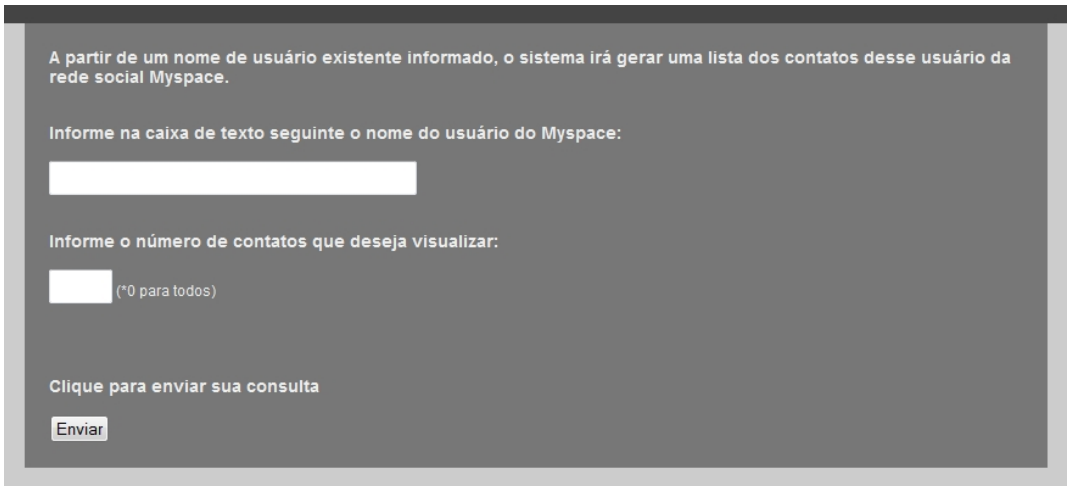

Figura 4.3: Tela inicial da aplicação

A partir desse momento o usuário visualiza um formulário onde pode incluir nos campos um nome de usuário do **Myspace** e o número de contatos que deseja visualizar como mostra a figura 4.4.

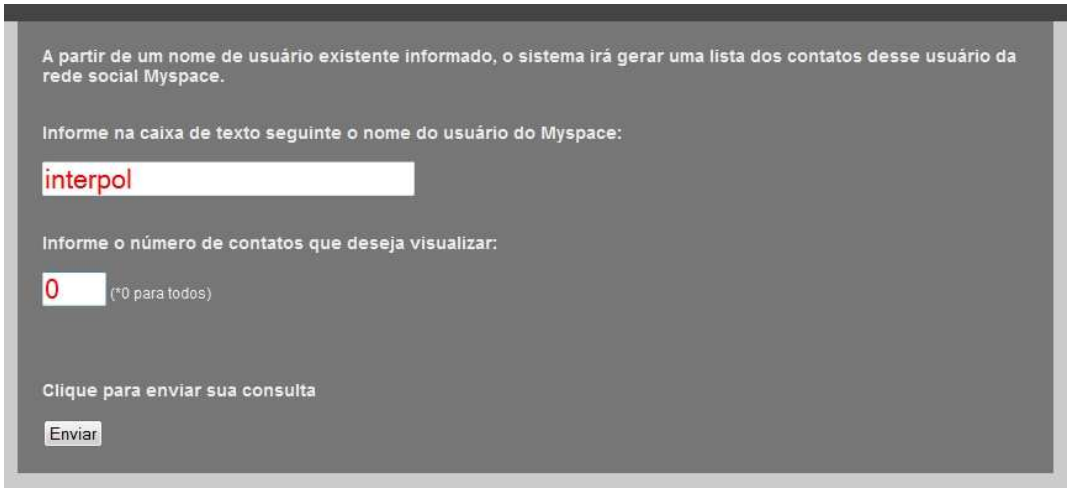

Figura 4.4: Tela com os campos preenchidos

Para este caso, o usuário objetiva visualizar a rede com todos os contatos da banda “Interpol” (o número “0” insere todos os contatos disponíveis).

A seguir, ao clicar em “Enviar” o usuário obtém o resultado mostrado na figura 4.5.

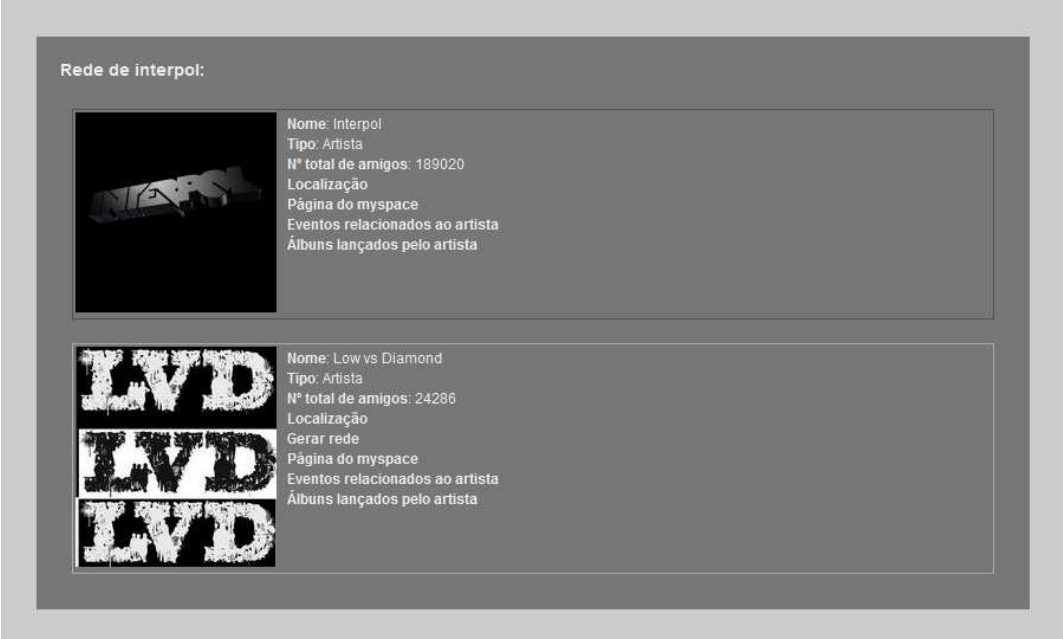

Figura 4.5: Tela que mostra os resultados ao usuário

A consulta retorna, além das próprias informações do artista, as informações de apenas um contato (“Low vs Diamond”) para este caso pois, apesar do usuário **interpol** possuir muitos amigos na rede **Myspace**, a aplicação somente realiza a busca nos **TopFriends** do usuário por questões de desempenho e, no caso, o usuário **interpol** possui um único **TopFriend**.

Para a manipulação do modelo RDF é realizada a consulta em linguagem SPARQL que pode ser vista na listagem 4.1.

```
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX myspace:<http://purl.org/ontology/myspace#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
SELECT ?name ?based_near ?depiction ?type ?rdf ?totalFriends
WHERE {
?x rdf:type foaf:PersonalProfileDocument
?x foaf:primaryTopic ?id .
?id foaf:name ?name .
optional { ?id foaf:based_near ?based_near . }
?id foaf:depiction ?depiction .
?id rdf:type ?type .
?id owl:sameAs ?rdf .
?id myspace:totalFriends ?totalFriends .
}
```

Listagem 4.1: Consulta SPARQL para recuperar os dados dos usuários do Myspace

A consulta se inicia com a definição dos prefixos a serem utilizados como bases das propriedades das relações que se deseja obter como resultado. Os prefixos, desse modo, podem ser vistos como as ontologias ou as sintaxes usadas nessa consulta específica: **FOAF**, ontologia do **Myspace**, a sintaxe RDF e a sintaxe OWL.

Para identificação do usuário é necessário buscar pelo sujeito da sentença em que o tipo do documento é um **PersonalProfileDocument** do **FOAF** e a partir desse sujeito buscar pelo ID do usuário através da propriedade **primaryTopic**, também do **FOAF**. Esse tipo de mecanismo foi utilizado especificamente para lidar com o tipo de arquivo RDF do **Myspace** do usuário gerado pelo **DBTune** e poderia ser diferente caso o tipo de arquivo RDF possuísse outra estrutura.

Nesse aspecto é de grande importância que um desenvolvedor observe muito bem os tipos de relações e padrões que existem nos arquivos que têm em mãos para não gerar resultados inconsistentes. Após a descoberta do ID do usuário é possível obter as propriedades que serão exibidas ao usuário final.

Na observação dos resultados é possível visualizar, além das informações geradas pelas consultas, algumas opções extras que fazem com que a aplicação fique mais rica. Essas opções são a geração de uma rede para determinado usuário encontrado na pesquisa de contatos, em que o mesmo procedimento inicial de pesquisa é utilizado com o ID do usuário escolhido, a visualização dos eventos ou *shows* a serem realizados pelos artistas e também dos álbuns lançados pelos artistas.

Caso clique na opção de visualizar os eventos da banda “Interpol” o usuário obtém o seguinte resultado, mostrado na figura 4.6.

Figura 4.6: Tela que mostra os próximos shows da banda “Interpol”

Cada evento é mostrado dentro de um quadro que contém as informações de **título**, **data** e ***homepage*** do evento. A consulta SPARQL para esse resultado pode ser visualizada na listagem 4.2.

```
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX terms:<http://purl.org/dc/terms/>
SELECT ?homepage ?date ?label
WHERE {
?x rdf:type <http://purl.org/ontology/mo/Performance> .
?x foaf:homepage ?homepage .
?x terms:date ?date .
?x rdfs:label ?label .
}
```

Listagem 4.2: Consulta SPARQL para recuperar os eventos de um artista do Myspace

O usuário pode voltar à página anterior (figura 4.5) e optar por visualizar os álbuns já lançados pela banda “Interpol”. Neste caso ele obtém o seguinte resultado, mostrado na figura 4.7.

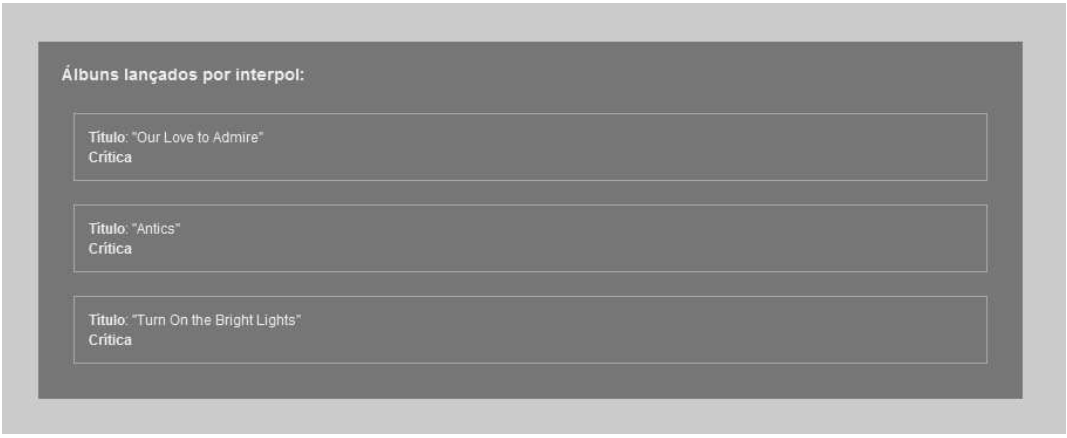

Figura 4.7: Tela que mostra os álbuns lançados pela banda “Interpol”

O usuário recebe como resultado os três álbuns já lançados pela banda, “Our Love to Admire”, “Antics” e “Turn On the Bright Lights” e um *link* para cada resenha feita no repositório musical **BBC**. A consultada usada para a obtenção desse resultado é exibida na listagem 4.3.

```
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX rev:<http://purl.org/stuff/rev#>
PREFIX dc:<http://purl.org/dc/elements/1.1/>
SELECT ?title ?review
WHERE {
?x foaf:made ?record .
?record dc:title ?title .
?record rev:hasReview ?review .
}
```

Listagem 4.3: Consulta SPARQL para recuperar os álbuns de um artista do Myspace

O usuário ainda pode retornar à sua busca inicial e gerar a rede da banda “Low vs Diamond”. Para este caso ele obtém o resultado mostrado na figura 4.8.

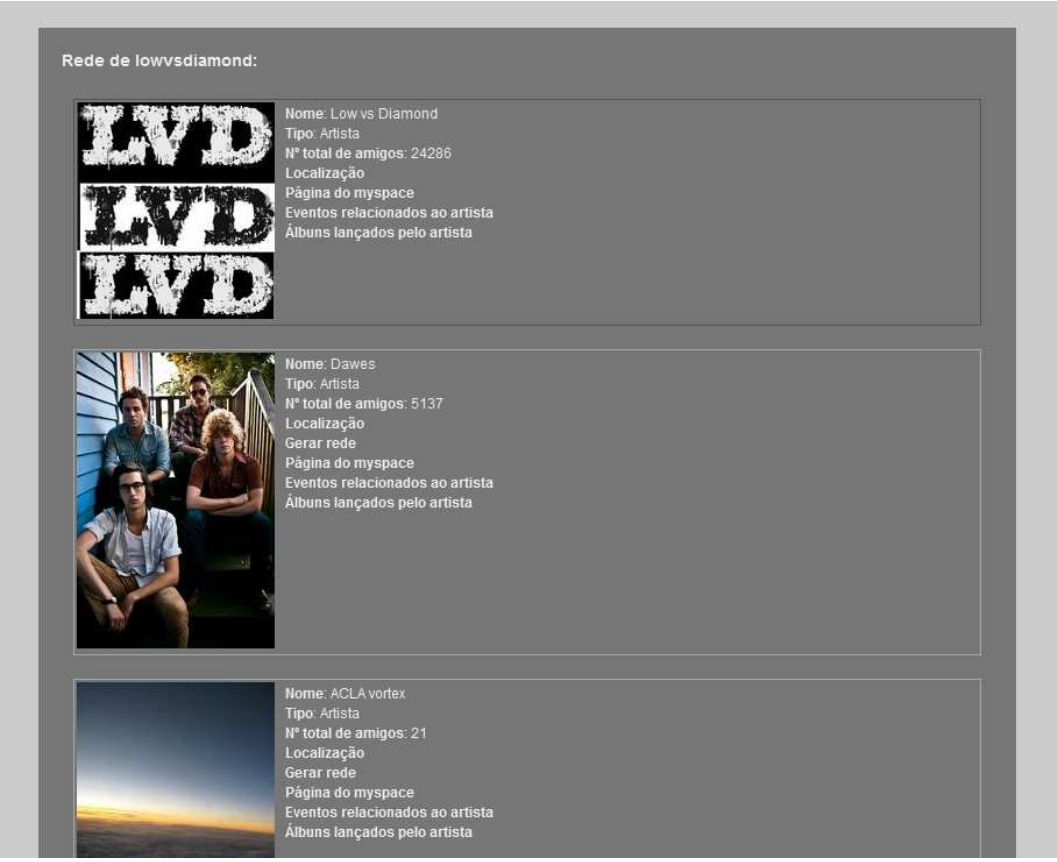

Figura 4.8: Tela que mostra a rede da banda “Low vs Diamond”

4.5. LIMITAÇÕES DA APLICAÇÃO

Verificou-se uma certa lentidão durante a inclusão dos dados no grafo interno. Isso se deve ao fato da grande quantidade de dados nos arquivos RDFs que devem ser capturados e transladados para o grafo interno. Muitas dessas informações são de certa forma inúteis, de acordo com a intenção da aplicação, e poderiam, desta forma, não serem incluídas no grafo interno.

Para suavizar essa limitação de desempenho, foi adicionada a opção de limitar o número de contatos a serem recuperados do **Myspace**. Esse aspecto deve ser muito melhorado quando se pensa em aplicações futuras na Internet que têm por obrigação possuir a característica de agilidade de resposta ao usuário final.

Outro ponto em que a aplicação demonstra falhas está fundamentalmente relacionado à dificuldade em se mapear e disponibilizar informações de artistas, principalmente os menos conhecidos, em arquivos RDFs. Assim uma série de consultas não retorna o resultado esperado por falta desse mapeamento, por exemplo, as informações de eventos e álbuns dependem do **lastfm.rdfize** que ainda é restrito a poucos artistas.

5. Considerações finais

A tendência de que mais aplicações utilizem o conceito de dados ligados e se apóiem não só nas tecnologias, como também em projetos já consolidados como o *Linking Open Data*, citado na seção 3.1, é grande pois os usuários finais estão se acostumando, cada vez mais, com a oportunidade de obterem informações variadas de forma automática e transparente e essa modificação já vem acontecendo dentro de centros de dados importantes como o **BBC** que intenciona disponibilizar todo o seu banco de dados interno em formato compatível com as tecnologias da Web Semântica. Os chamados "buscadores da nova geração", como o **Yahoo!SearchMonkey**, citado na seção 1.1.3, recomendam que os próprios desenvolvedores de *Websites* incluam mais semântica em suas páginas para que elas sejam indexadas por seus *robots*.

Outro exemplo da melhoria que a utilização de padrões da Web Semântica pode trazer ocorre na área de comércio eletrônico. Atualmente, os robôs de compras, chamados *shopbots*, tentam realizar buscas por produtos capturando informações que aparecem regularmente nas páginas dos portais de compra e venda. Desse modo, só as informações mais comuns são obtidas como nome e preço de um produto e, consequentemente, outras informações também úteis ao usuário final como o tipo de impostos e o custo do frete são desprezadas. A programação desses robôs está diretamente relacionada ao *layout* da página em que é feita a busca, havendo a necessidade de reprogramação sempre que há alguma modificação nesse *layout*, o que é um ponto negativo para a utilização dessa prática. A partir do momento em que as lojas começarem a publicar suas próprias ontologias de produtos e serviços, agentes de *software* poderão realizar buscas mais complexas que favoreçam a intenção de pesquisa do usuário (BREITMAN, 2005).

Para que um desenvolvedor possa integrar seu *Website* ao esquema proposto pelo conceito de dados ligados é recomendável que modele e publique os seus dados no formato RDF para que eles sejam interpretáveis pelos diversos agentes de *software* que podem realizar alguma busca sobre esses dados como os *Web Crawlers*. Outra forma de tornar as páginas de seu *Website* legíveis por esses agentes é anotá-las com a utilização de algum padrão de anotação semântica de dados como o RDFa, citado na seção 2.2.1. É necessário também que o desenvolvedor utilize padrões de URL para facilitar a inteligibilidade do conteúdo do seu *Website*. Portanto, uma entidade representada por sua URL

http://exemplo.com/entidade/ pode ter suas URLs **http://exemplo.com/entidade/rdf/** e **http://exemplo.com/entidade/html/** representando suas formas em RDF e em HTML, respectivamente.

Um desenvolvedor que deseja construir aplicações que envolvam os conceitos de dados ligados pode aprender a trabalhar com algum dos *frameworks* disponíveis para a manipulação de dados semânticos como SESAME, Jena, Redland ou Mulgara, citados na seção 2.3; determinar quais os repositórios que oferecem as informações segundo a necessidade da aplicação; estabelecer as políticas de armazenamento e manipulação dos dados internamente e aplicar técnicas que evitam redundâncias e ambiguidades entre esses dados.

O que foi observado é que ainda existem algumas falhas de certa forma pertinentes quando estamos falando de uma evolução gradativa do modelo da Web de documentos, quanto à disponibilização dos dados existentes na Web em formato interpretável por máquinas como, por exemplo, a limitada quantidade de informações que pode ser obtida do **lastfm.rdfize**, plataforma utilizada para o desenvolvimento da aplicação desenvolvida para este trabalho que disponibiliza informações musicais de apenas alguns artistas mais conhecidos. Conclui-se que haverá um certo período de tempo para que as plataformas semânticas estejam abastecidas com uma quantidade de informações compatível com a quantidade disposta em documentos XHTML ou em banco de dados relacionais sem estarem ligadas entre si.

Futuramente, além da maior difusão dos conceitos de dados ligados, o que pode ser realizado é a implementação dos serviços ainda não consolidados da Web Semântica, principalmente as camadas superiores da arquitetura da figura 2.1, ***Proof*** e ***Trust***.

Referências

BECKET, David. **The Design and Implementation of the Redland RDF Application Framework**, Computer Networks, n.5, v.39, p.577--588, 2002. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S1389128602002219>

BERNERS-LEE, Tim. **Linked Data**, Retrieved April, v.12, p.2008, 2006. Disponível em: <http://www.w3.org/DesignIssues/LinkedData.html>

BREITMAN, Karin. **Web Semântica: a Internet do futuro**. Rio de Janeiro: LTC Editora, 2005. 190p.

CARNEIRO, R.; BRITO, P. **Definição de uma Ontologia em OWL para Representação de Conteúdos Educacionais**, VII Encontro de Eestudantes de Informática do Estado do Tocantins, Palmas, 2005. Disponível em: <http://www.arquivar.com.br/espaco_profissional/sala_leitura/teses-dissertacoes-e-monografias/Definicao_de_uma_Ontologia_em_OWL_para_Representacao_de_Conteudos_Educacionais.pdf>

CELMA, Oscar, et al. **Getting music recommendations and filtering newsfeeds from FOAF descriptions**, Proceedings of Workshop on Scripting for the Semantic Web, Heraklion, Greece, 2002. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.59.8779&rep=rep1&type=pdf>

CYGANIAK, R.; JENTZSCH, A. **About the Linking Open Data Dataset Cloud**, 2009. Disponível em: <http://richard.cyganiak.de/2007/10/lod/>

HAUSENBLAS, Michael. **Exploiting Linked Data For Building Web Applications**, IEEE Internet Computing, p.80--85, 2009. Disponível em: <http://sw-app.org/pub/exploit-lod-webapps-IEEEIC-preprint.pdf>

HEATH, Tom. **An Introduction to Linked Data**, 2009. Disponível em: <http://tomheath.com/slides/2009-02-austin-linkeddata-tutorial.pdf>

LUGER, Michael, et al. **EASAIER Semantic Music Retrieval Portal**, The 2nd international conference on Semantics And digital Media Technologies (SAMT), Genova, Italy, p.5--7, 2007. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.143.1596&rep=rep1&type=pdf>

MACMANUS, Richard. **The State of Linked Data in 2010**. Post publicado em: <http://www.readwriteweb.com/>, 2010. Disponível em: <http://www.readwriteweb.com/archives/the_state_of_linked_data_in_2010.php>

PALMISANO, Davide. **RWW 2009 Top 10 Semantic Web products: one year later…**, Post publicado em: <http://davidepalmisano.wordpress.com/>, 2009.

Disponível em: <http://davidepalmisano.wordpress.com/2009/12/13/rww-2009-top-10-semantic-web-products-one-year-later/>

PASSANT, A.; RAIMOND, Y. **Combining Social Music and Semantic Web for music-related recommender systems**, Proceedings of the First Social Data on the Web Workshop, Karlsruhe, Germany, October, n.2008, v.27, p.1613--1673, 2008. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.142.6434&rep=rep1&type=pdf>

RAIMOND, Yves, et al. **THE MUSIC ONTOLOGY**, Proceedings of the International Conference on Music Information Retrieval, p.417--422, 2007. Disponível em: <http://wtlab.um.ac.ir/parameters/wtlab/filemanager/LD_resources/other/THE%20MUSIC%20ONTOLOGY%202007.pdf>

SALES, Alberto. **Web Semântica: O estado da arte**. 2008. 60p. Trabalho Individual (Mestrado em Ciência da Computação) - Faculdade de Informática, Pontifícia Universidade Católica do Rio Grande do Sul.
Disponível em: <http://www.scribd.com/doc/7253958/Web-Semantica-Estado-Da-Arte>

SEGARAN, Toby, et al. **Programming the Semantic Web**. 1.ed. United States: O'Reilly Media, 2009. 280p.