Pontifícia Universidade Católica
do Rio de Janeiro

Helena Serrão Piccinini

W-Ray - Uma abordagem para publicação de dados da Deep Web

Tese de Doutorado

Tese apresentada como requisito parcial para obtenção do título de Doutor pelo Programa de Pós-Graduação em Informática da PUC-Rio.

Orientador: Prof. Marco Antonio Casanova

Rio de Janeiro

Junho de 2013

Pontifícia Universidade Católica
do Rio de Janeiro

Helena Serrão Piccinini

W-Ray – Uma abordagem para publicação de dados da Deep Web

Tese apresentada como requisito parcial para obtenção do título de Doutor pelo Programa de Pós-Graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Marco Antonio Casanova
Orientador
Departamento de Informática - PUC-Rio

Prof. Antonio Luz Furtado
Departamento de Informática - PUC-Rio

Prof. Helio Côrtes Vieira Lopes
Departamento de Informática - PUC-Rio

Prof. José Antonio Fernandes de Macêdo
Universidade Federal do Ceará

Prof. Luiz André Portes Paes Leme
Universidade Federal Fluminense

Prof. José Eugênio Leal
Coordenador(a) Setorial do Centro Técnico Científico - PUC-Rio

Rio de Janeiro, 21 de junho de 2013

Helena Serrão Piccinini

Graduou-se em Engenharia Civil pela Universidade Federal Fluminense (UFF) em 1982. Trabalha com Administração de Banco de Dados desde 1984. Funcionária do Instituto Brasileiro de Geografia e Estatística (IBGE) desde 1986. Concluiu o Mestrado em Informática na área de Banco de Dados pela Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio) em 1998.

Ficha Catalográfica

Piccinini, Helena Serrão

W-Ray - Uma abordagem para publicação de dados da Deep Web / Helena Serrão Piccinini ; orientador: Marco Antonio Casanova. – 2013.

195 f. : il. (color.) ; 30 cm

Tese (doutorado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2013.

Inclui bibliografia

1. Informática – Teses. 2. Banco de dados. 3. Deep Web. 4. Linguagem natural. 5. Dados ligados. 6. Web semântica. 7. Mapeamento RDB_to_RDF. I. Casanova, Marco Antonio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Ao grande amor da minha vida,
Marcos de Carvalho Machado
pela grande ajuda, compreensão e carinho.

Agradecimentos

Aos meus pais, pelo incentivo durante toda a minha vida.

Aos meus queridos filhos Rafael e Juliana, por terem suportado quatro longos anos marcados por muitos momentos de tensão e ausência.

Ao meu orientador prof. Marco Antonio Casanova, pela dedicação, estímulo e paciência. Pela confiança depositada ao sugerir um tema tão envolvente e voltado para os problemas do IBGE.

Ao prof. Antônio Luz Furtado, a quem devo os primeiros passos desta tese.

À grande amiga Elvira Maria Antunes Uchoa, pelas contribuições e revisão do texto.

À PUC-Rio pelos auxílios concedidos, sem os quais este trabalho não seria possível.

Ao IBGE, pela oportunidade e suporte oferecidos. Em particular ao ex-diretor executivo do IBGE, Sergio da Costa Côrtes, e à ex-coordenadora da Coordenação de Metodologia e Banco de Dados do IBGE, Maria Célia Pelisson Jacon, pela concessão da licença e confiança a mim depositada.

Aos colegas e amigos do IBGE, que contribuíram com o desenvolvimento dos experimentos seja com dados, conhecimento, discussões ou tempo. Em particular: Cláudio Mariano Fernandes, Carlos Alberto dos Santos, Luiz Antônio Figueredo, Luiz Antônio Louzada, Luiz Paulo Nascimento, José Masello, Pedro Paulo Ribeiro Kappaum, Sônia Vasques Nogueira.

À grande amiga Alice Maria Barreto Vieira, pelos incentivos durante todo o curso.

Ao grande amigo Adriano Francisco Branco, pelo apoio e ajuda em momentos difíceis deste curso.

A todos os colegas, professores e funcionários do Departamento de Informática da PUC-Rio, pelo companheirismo, aprendizado e auxílio.

À todos os amigos que de uma forma ou de outra me estimularam ou me ajudaram.

Resumo

Piccinini, Helena Serrão; Casanova, Marco Antonio. **W-Ray - Uma abordagem para publicação de dados da Deep Web.** Rio de Janeiro, 2013. 195p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A Deep Web é composta por dados armazenados em bases de dados, páginas dinâmicas, páginas com scripts e dados multimídia, dentre outros tipos de objetos. Os bancos de dados da Deep Web são geralmente sub-representados pelos motores de busca, devido aos desafios técnicos de localizar, acessar e indexar seus dados. A utilização de hyperlinks pelos motores de busca não é suficente para alcançar todos os dados da Deep Web, exigindo interação com interfaces de consultas complexas. Esta tese apresenta uma abordagem, denominada W-Ray, capaz de fornecer visibilidade aos dados da Deep Web. A abordagem baseia-se na descrição dos dados relevantes através de sentenças bem estruturadas, e na publicação dessas sentenças em páginas estáticas da Web. As sentenças podem ser geradas com RDFa embutido, mantendo a semântica do banco de dados. As páginas da Web assim geradas são passíveis de ser indexadas pelos motores de coleta de dados tradicionais e por motores mais sofisticados que suportam busca semântica. É apresentada também uma ferramenta que apóia a abordagem W-Ray. A abordagem foi implementada com sucesso para diferentes bancos de dados reais.

Palavras-chave

Banco de dados; deep web; linguagem natural; dados ligados; mapeamento RDB-to-RDF; Web Semântica.

Abstract

Piccinini, Helena Serrão; Casanova, Marco Antonio (Advisor). **W-Ray - An approach to the Deep Web data publication.** Rio de Janeiro, 2013. 195p. DSc Thesis - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The Deep Web comprises data stored in databases, dynamic pages, scripted pages and multimedia data, among other types of objects. The databases of the Deep Web are generally underrepresented by the search engines due to the technical challenges of locating, accessing and indexing them. The use of hyperlinks by search engines is not sufficient to achieve all the Deep Web data, requiring interaction with complex queries interfaces. This thesis presents an approach, called W-Ray, that provides visibility to Deep Web data. The approach relies on describing the relevant data through well-structured sentences, and on publishing the sentences as static Web pages. The sentences can be generated with embedded RDFa, keeping the semantics of the database. The Web pages thus generated are indexed by traditional Web crawlers and sophisticated crawlers that support semantic search. It is also presented a tool that supports the W-Ray approach. The approach has been successfully implemented for some real databases.

Keywords

Database; deep web; natural language; linked data; RDB_to_RDF mapping; Semantic Web.

Sumário

Lista de Figuras

Lista de Tabelas

"A vantagem de se ter péssima memória é
divertir-se muitas vezes com as mesmas coisas boas como se fosse a primeira vez."

Friedrich Nietzsche (1855-1900)

1
Introdução

Neste capítulo é apresentada a motivação para este estudo, os objetivos do trabalho e como o documento da tese está organizado.

1.1. Motivação

Ao contrário da Web convencional que contém apenas páginas estáticas, a *Deep Web* (Bergman, 2001; Madhavan et al., 2009) inclui bases de dados, páginas dinâmicas e dados multimídia, dentre outros tipos de objetos. Os bancos de dados da *Deep Web* são normalmente sub-representados nos motores de busca, devido aos desafios técnicos de localização, acesso e indexação que envolvem estes dados. De fato, desde que os dados da *Deep Web* não estão disponíveis como páginas estáticas da Web, os motores de busca tradicionais não podem descobrir os dados armazenados nos bancos de dados através do rastreamento dos hiperlinks tendo que interagir com complexas interfaces de consultas.

O cenário da *Deep Web* também inclui diferentes tipos de dados como os dados multimídia, dados geográficos em formato vetorial, que geralmente são disponibilizados na Web através de ferramentas que oferecem facilidades para converter dados geográficos em páginas Web dinâmicas (ArcGis, 2012, MapServer, 2011), dados geográficos em formato *raster*, que incluem as imagens de satélite, e dados estatísticos, que geralmente são muito volumosos.

Duas abordagens básicas para acessar os dados da *Deep Web* têm sido propostas. A primeira abordagem, conhecida como *Surfacing* ou *Deep Web Crawl* (Cafarella et al., 2011; Madhavan et al., 2009; Maiti et al., 2009; Madhavan et al., 2008; Raghavan & Garcia-Molina, 2001), tenta preencher automaticamente os formulários HTML de consulta aos bancos de dados para executar as consultas *offline* e traduzir os resultados para páginas HTML estáticas, que finalmente podem ser indexadas. A segunda abordagem, conhecida como *Federated Search*, or *Virtual Integration* (Kabisch et al., 2010; Cafarella et al., 2009; He et al., 2005a; He et al., 2005b; Callan, 2002), sugere o uso de mediadores para um

domínio específico com o objetivo de facilitar o acesso às bases de dados. Estratégias híbridas, que são extensões das abordagens anteriores, também têm sido propostas (Rajaraman, 2009).

Apesar dos progressos recentes, o acesso aos dados da *Deep Web* ainda é um desafio, por várias razões:

- A completa indexação da *Deep Web* pode ser um problema não escalável. Uma vez que a *Deep Web* é significativamente maior do que a Web convencional, não é viável indexá-la completamente. Segundo Dragut et al. (2012) mesmo que os tamanhos da *Surface Web (*ou Web convencional*)* e da *Deep Web* não sejam conhecidos, existe um consenso na comunidade da Web de que a *Deep Web* é muitas vezes maior do que a *Surface Web*.
- As interfaces da Web, que interagem com os bancos de dados relacionais da *Deep Web*, são projetadas para humanos, o que complica o desenvolvimento de agentes de software para entendê-las e aplicar estratégias de preenchimento automático.
- Os dados multimídia e os dados geográficos, não possuem, além de seus metadados, descrições que possam ser indexadas.
- Por razões de segurança e desempenho, muitos donos de dados bloqueiam a entrada dos motores de coleta que utilizam a abordagem *Surfacing* como uma tentativa de indexação dos dados da *Deep Web*.

Em paralelo aos esforços de aproveitamento da *Deep Web*, pode-se observar que nos últimos anos muitos dados estruturados foram mapeados para o formato RDF[1] *(Resource Description Framework)* (Manola & Miller, 2004) e disponibilizados na Web. Entretanto, durante o processo de mapeamento, os donos dos dados não se preocuparam com o reuso de ontologias já disponíveis para o domínio dos seus dados e menos ainda em não replicar indivíduos[2] já publicados no formato RDF por outros donos de dados. O resultado disso foi um número elevado de dados isolados, cuja serventia se restringia aos seus próprios donos, além da grande quantidade de informação replicada e de difícil integração (Bizer et al., 2009).

[1] RDF - Conceito disponível no Apêndice A
[2] Ver explicação inserida no conceito RDF - Apêndice A

Com objetivo de organizar este cenário, surgiu um conjunto de boas práticas para publicar e conectar dados estruturados na Web, denominado *"dados ligados" (linked data)* (Berners-Lee, 2006).

Uma vez que as unidades primárias da Web são documentos HTML conectados por hiperlinks não tipados, os *dados ligados* dependem de documentos que contenham dados em formato RDF. No entanto, ao invés de simplesmente conectar esses documentos, os *dados ligados* usam o RDF para fazer declarações tipadas que ligam qualquer coisa no mundo. O resultado disto é conhecido como *Web de dados* (Bizer et al., 2009).

A Web de dados não é indexada por mecanismos de busca tradicionais, no entanto, em 2011 foram divulgadas algumas estatísticas feitas pelo Yahoo! que mostraram que o número de páginas com RDFa[3] embutido demonstrou um crescimento em torno de 510% entre março de 2009 e outubro de 2010 (Mica, 2011). Esse crescimento explosivo foi creditado ao fato de que, desde 2008, os motores de coleta de dados na Web, da Yahoo! começaram a avaliar as *tags* RDFa (Yahoo!, 2008). O mesmo se deu com o Google, a partir de 2009 (Google, 2009).

Por sua vez, o objetivo do RDFa é estruturar informações contidas em páginas HTML estáticas, tendo sido desenvolvido para mapear os dados de bancos de dados relacionais para o RDF.

1.2. Objetivo

O objetivo desta tese é propor uma abordagem diferente, denominada W-Ray, para tornar visível, na superfície da Web convencional e na Web de dados, os diferentes tipos de dados que compõem a *Deep Web*. O nome W-Ray foi escolhido porque assim como o RaioX (X-Ray) permite a visualização das estruturas ósseas do interior do corpo humano, a abordagem W-Ray permite a visualização dos bancos de dados subjacentes à Web.

A ideia básica consiste na criação de um conjunto de sentenças em linguagem natural (LN), com uma estrutura simples, para descrever os dados da *Deep Web* e que são publicadas em páginas HTML estáticas, que podem ser

[3] RDFa (RDF-em-atributos) (Adida et al., 2012) é uma recomendação do W3C para agregar um conjunto de atributos RDF em documentos XHTML, permitindo assim a extração de triplas RDF por agentes de software.

facilmente indexadas, como de costume, pelos mecanismos de busca. O uso de sentenças em linguagem natural é interessante por três motivos:

– Tornam as páginas geradas aceitáveis aos motores de coleta da Web. De outra forma, os motores de busca consideram palavras distribuídas aleatoriamente em uma página como uma tentativa de manipulação do *pageRank*, fato que pode ter como consequência a não indexação do conteúdo da página HTML pelos mecanismos de busca.
– Como uma alternativa para a síntese de sentenças em LN, pode-se simplesmente formatar os dados das views materializadas como tabelas HTML. No entanto, esta não é uma estratégia razoável, pois alguns mecanismos de busca ainda consideram tabelas como objetos visuais e os que conseguem indexá-las ainda não resolvem todos os problemas que envolvem este tipo de objeto HTML (Veneti et al., 2011; Madhavan et al., 2009; Cafarella et al., 2008), tais como tabelas com cabeçalhos mais complexos. Além disso, tabelas grandes podem ser difíceis de ler para o usuário comum e, completamente impossível, para os usuários com deficiência visual.
– As descrições assim geradas são minimamente aceitáveis para os humanos. Para que páginas HTML estáticas sejam indexadas pelos mecanismos de busca elas devem estar visíveis na Web também para humanos. Por este motivo, na abordagem proposta as páginas da Web são geradas seguindo as diretrizes de acessibilidade do consórcio W3C (Caldwell et al., 2008) e as recomendações publicadas pela Google para otimizar a indexação de sites (Google-OptimizationGuide, 2012).

A abordagem é dividida em três etapas.

1. A primeira etapa consiste na especificação de *views* materializadas (*views* de banco de dados) que descrevem os objetos da *Deep Web*.
2. A segunda etapa mapeia o conjunto de *views* materializadas para um esquema RDF, seguindo os princípios de *dados ligados*.
3. A terceira pode incluir uma ou as duas alternativas descritas a seguir:
   3.1. O administrador dos dados pode decidir trazer, para a superfície, os dados das views como sentenças em linguagem natural, organizadas em páginas da HTML estáticas com RDFa embutido, o que preserva a estrutura dos dados. As páginas da Web serão então indexadas pelos motores de busca

tradicionais e por aqueles que suportam busca semântica com base em RDFa. Sob este aspecto, W-Ray pode ser entendido como uma alternativa para a abordagem Surfacing porque consegue combinar uma estratégia para descrever os dados da Deep Web com o objetivo de torná-los visíveis aos mecanismos de coleta da Web convencional. A estratégia do W-Ray transfere, para a figura do administrador de dados, a responsabilidade de decidir quais dados devem ser expostos na Web e como eles devem ser publicados. Isto desobriga o trabalho de sondagem feito, por motores de coleta em banco de dados, através de tentativas de preenchimento dos formulários HTML. Essa mudança de paradigma representa a maior diferença entre W-Ray e a abordagem Surfacing.

3.2. O administrador pode decidir gerar triplas RDF a partir dos dados das views, com base no esquema RDF e seguindo os princípios de dados ligados. As triplas RDF podem ser armazenadas em um repositório ou expostas como um SPARQL endpoint[4] para permitir a consulta do RDF gerado através da linguagem SPARQL.

A abordagem W-Ray fornece uma atenção especial aos dados geográficos porque esses dados geralmente são publicados na Web convencional através de ferramentas como ArcGis e MapServer (ArcGis, 2012, MapServer, 2011) que são capazes de converter dados geográficos, em formato vetorial, para páginas Web dinâmicas. No entanto, tais páginas Web dinâmicas não são indexadas pelos motores de busca. Portanto, é necessária uma estratégia específica para estes dados, que seja capaz de descrevê-los, de forma a atrair os mecanismos de coleta para a indexação dos dados. Observações semelhantes aplicam-se a dados geográficos em formato *raster*. Os dados de cubos estatísticos também requerem uma atenção especial para evitar a geração de grandes descrições (isto é, sentenças ou triplas). A abordagem W-Ray sugere a criação de descrições de sub-cubos, em vez de células, combinados com as descrições dos domínios.

Para viabilizar a abordagem W-Ray foi desenvolvida uma ferramenta, denominada W-RayS, que apoia o projeto. Usando W-RayS, quatro estudos de caso foram realizados para avaliar a eficácia da abordagem. Os estudos de caso

[4] SPARQL - Linguagem de consulta de manipulação de dados no formato RDF. Recomendada pelo W3C em http://www.w3.org/TR/rdf-sparql-query/

foram baseados em dados reais mantidos pelo IBGE - Instituto Brasileiro de Geografia e Estatística e pelo INPE - Instituto Nacional de Pesquisas Espaciais. Os resultados obtidos se mostraram promissores.

A ideia da abordagem W-Ray foi publicada inicialmente em: Piccinini et al. (2010a) e Piccinini et al. (2010b) e Furtado et al. (2010). Desde então, evoluiu e passou a oferecer um suporte melhor para o projeto de ontologias, em consonância com os preceitos de *dados ligados*, embutindo RDFa nas páginas HTML, o que consequentemente permite que a estrutura dos dados seja preservada. A ferramenta W-RayS também passou a oferecer serviços de triplificação e um suporte melhor para a síntese da linguagem natural. O site da abordagem W-Ray com sua descrição e os projetos implementados é: http://www.inf.puc-rio.br/~hpiccinini/

1.3.Contribuições

As principais contribuições que este trabalho oferece são:

- Apresentação de uma abordagem sistemática capaz de tornar visíveis na superfície da Web convencional e na Web de dados, diferentes tipos de dados ocultos na *Deep Web*. Na Web convencional tira proveito da tecnologia já existente de indexação, busca e localização através dos mecanismos de busca tradicionais. Na Web de dados, tira proveito de agentes de software que conseguem extrair triplas RDF a partir do RDFa com o objetivo de fornecer buscas mais ricas e específicas para os dados.
- Liberdade e segurança que o dono do dado adquire através da garantia, fornecida pela abordagem W-Ray, de que apenas os dados relevantes e públicos de seus bancos de dados serão indexados pelos mecanismos de busca.
- Oportunidade de indexação, através de mecanismos de busca, de outros tipos de dados, tais como, os dados geográficos, em formato *raster* e vetorial, e dados estatísticos;

1.4. Organização da Tese

A tese é composta por sete capítulos. O capítulo 2 resume alguns conceitos e trabalhos relacionados. O capítulo 3 apresenta a abordagem W-Ray. O capítulo

4 descreve o kit de ferramentas desenvolvido para dar suporte à abordagem W-Ray. O capítulo 5 descreve quatro estudos de caso realizados para avaliar a abordagem W-Ray. O capítulo 6 discute os resultados obtidos nos estudos de caso. E, por fim, no capítulo 7 são apresentadas as conclusões.

2
Fundamentação

A abordagem W-Ray, descrita no capítulo 3, foi elaborada com base nos estudos relacionados a *dados ligados* na Web e à geração de linguagem natural. No entanto, a sua motivação está concentrada nos problemas enfrentados pela área de Recuperação de Informações na Web, mais especificamente nos problemas que os mecanismos de busca enfrentam na *Deep Web*.

Este capítulo está organizado da seguinte maneira:

- No item 2.1 é apresentado um resumo sobre *Mecanismos de Busca na Web,* com o enfoque na *Deep Web,* as principais abordagens propostas para o problema de localização de dados da *Deep Web* e uma discussão sobre estas abordagens;
- O item 2.2 inclui uma revisão de *Dados Ligados na Web,* com destaque para *Publicação dos Dados Ligados na WEB* e *Vocabulários na Web de Dados*;
- O item 2.3 discute o *Mapeamento de Bancos de Dados Relacionais para RDF* (RDB-to-RDF) destacando pontos relevantes de alguns trabalhos envolvendo mapeamentos RDB-to-RDF; e finalmente é apresentado um resumo sobre *RDFa*;
- A seguir, no item 2.4 pode ser encontrada uma breve revisão sobre *Geração de Linguagem Natural,* com destaque para a *Geração de Sentenças a partir da Linguagem OWL (OWL-to-RDF).* São apresentados os trabalhos relacionados ao mapeamento de OWL-to-LN e uma discussão sobre as estas abordagens de mapeamento de OWL-to-RDF;
- No item 2.5 é apresentado um resumo do capítulo;

2.1. Mecanismos de busca na Web

Os mecanismos de busca (*Search Engines*) são sistemas de recuperação de informações que buscam grande quantidade de informações acessíveis via

navegadores Web. Os principais módulos dos mecanismos de busca tradicionais são (Castillo, 2004):

- O módulo de coleta de páginas (*Web crawler*) que é responsável por navegar periodicamente pela Web, visitando e selecionando documentos de acordo com a política de relevância do mecanismo de busca, para montar um repositório composto por um subconjunto da Web.
- O módulo de indexação que analisa o conteúdo de cada página armazenada no repositório criado no módulo de coleta, gera um índice associando a URL a cada palavra-chave das páginas e armazena o índice em um grande banco de dados. Os métodos de indexação empregados variam de um mecanismo de busca para outro.
- O módulo de busca e ordenação que recebe as requisições dos usuários, pesquisa o índice gerado no módulo de indexação, seleciona alguns endereços de documentos candidatos à resposta e classifica-os de acordo com uma estimativa de relevância que pode atender a necessidade do usuário.

O módulo de coleta é especial interesse para esta pesquisa devido ao seu tratamento em relação às páginas HTML dinâmicas. Por isso, a seguir é apresentado um resumo sobre ele, destacando seus principais problemas, principalmente no que se refere à *Deep Web* em mais detalhes.

2.1.1.Módulo de Coleta

O módulo de coleta (*Web crawler*) (Pant et al., 2004), também conhecido como *bots, spiders, Web robots, worms* ou simplesmente *crawler*, inicia seu processo de navegação na Internet através de um conjunto inicial de URLs armazenado em uma lista chamada "sementes" (*seeds)* previamente definidas pela política do mecanismo de busca. Quando uma URL da lista *seeds* é acessada pelo robô, ele faz o download da página, analisa todas as URLs inseridas nessa página a fim de selecionar e armazenar essas novas URLs em uma lista de páginas a visitar chamada "limite" (*frontier)*. Esse processo é repetido até satisfazer a condição de parada do robô.

Em Castillo (2004), pode ser encontrada uma classificação para os módulos de coleta de acordo com o uso de políticas de implementação que visam melhorar o seu rendimento. São elas:

Segundo Castillo (2004) o comportamento de um motor de coleta é o resultado de uma combinação de políticas:

- uma política de seleção que define que páginas devem ser baixadas;
- uma política de revisita, que verifica se houveram alterações nas páginas;
- uma política de ajuste que define como evitar sobrecarrega nos sites;
- uma política de paralelização que define como coordenar a coleta distribuída.

De acordo com a política de seleção podem ser identificados tipos diferentes de motores de coleta:

– Os que buscam somente *links* nas páginas HTML. Basicamente, este tipo de módulo de coleta tenta encontrar o máximo de referências possíveis usando estratégias como procurar apenas por URLs que terminam com .html, .htm, .asp, .aspx, .php ou com "/". Este tipo de módulo de coleta é bastante usado para verificar se os *links* das páginas continuam funcionando.

– Os que tentam encontrar todos os recursos de um determinado *site*. Basicamente, utiliza um *link* inicial passado como referência e tenta extrair o máximo de páginas navegando pelos diretórios da URL. Este tipo de módulo de coleta pode ser usado quando se deseja indexar todo o conteúdo de um *site*.

– Os que buscam páginas cujo conteúdo se encaixa dentro de um ou vários tópicos previamente determinados. Trabalha com abordagens que usam apenas os nomes dos *links* para decidir se a página será baixada ou não, bem como abordagens que empregam uma medida de similaridade entre o conteúdo HTML das páginas baixadas com os conteúdos das páginas ainda não visitadas para decidir se irá baixar a página ou não. Este tipo de busca, focada em um determinado domínio, é conhecida como busca vertical.

Todo módulo de coleta deve seguir o "protocolo de exclusão", que consiste em examinar o arquivo robots.txt definido pelo administrador de cada servidor Web. É neste arquivo que são incluídos os comandos que permitem ou restringem a entrada do módulo de coleta.

Segundo Castillo (2004) um módulo de coleta enfrenta dois problemas principais:

– o grande número de páginas na Web - a maioria dos rastreadores tenta contornar este problema estabelecendo políticas próprias de prioridades para a seleção das páginas que serão coletadas.

– o tempo de atualização do conteúdo das páginas - a rapidez com que as páginas são atualizadas diminui a probabilidade do módulo de coleta rastrear o conteúdo atualizado. Políticas que definem o tempo de retorno do módulo de coleta em cada site ou a criação de *sitemaps*[5] contendo a frequência de atualização do site são algumas soluções adotadas.

Outro grande problema também identificado em Shestakov (2008) e Castillo (2004) é a geração dinâmica de páginas. Na maioria dos casos, os rastreadores não conseguem coletar dados destas páginas. Este problema representa o maior desafio para os motores de busca no contexto da *Deep Web*, por isso é detalhado a seguir.

As páginas dinâmicas compõem a Web dinâmica. Na Web dinâmica, o conteúdo de uma página é gerado em tempo de execução, ou seja, é gerado somente após um pedido feito a um programa, que esteja executando no servidor ou no cliente (Shestakov, 2008). No que diz respeito à Web estática ou tradicional, as páginas já existem no servidor e possuem conteúdo pré-definido, prontas para serem transmitidas a um cliente quando o pedido é recebido.

Em Shestakov, (2008) e Castillo, (2004) podemos encontrar descrições sobre dois tipos de páginas dinâmicas que foram resumidas da seguinte maneira:

- O primeiro tipo inclui parâmetros utilizados em suas URLs para a geração de novas páginas. Estas páginas podem ser detectadas pelos rastreadores porque suas URLs contêm os símbolos '?' e '&'. Tecnicamente, essas páginas são indexáveis, porque são acessíveis através de hiperlinks. No entanto, tais páginas geram um problema. Uma escolha errada na estratégia de mapeamento dos *links* das páginas pode levar o módulo de coleta a infinitas requisições de páginas, ou seja, uma página gerada dinamicamente pode ter um link para outra página dinâmica que pode levar à página seguinte e assim por diante.
- O segundo grupo de páginas dinâmicas é composto por páginas geradas com base em parâmetros fornecidos por um usuário através de interfaces de busca ou formulários HTML. Essas interfaces fornecem aos usuários da Internet o

[5] *Sitemap* é um arquivo XML que relaciona as URLs de um site e os metadados adicionais sobre cada URL para que os mecanismos de busca possam indexar o site de maneira mais inteligente (ex.: quando o site foi atualizado pela última vez; com que freqüência ele é alterado; qual a importância de cada URL no site).

> acesso *online* a inúmeros bancos de dados na Web. A fim de obter alguma informação a ser extraída do banco de dados de interesse, o usuário formula a sua consulta, especificando os termos dessa consulta em um formulário HTML e recebe como resultado da consulta, um conjunto de páginas dinâmicas que incorporaram as informações necessárias, extraídas de um banco de dados. No entanto, a formulação de uma consulta arbitrária via interface de busca é uma tarefa extremamente complexa para os *Web crawlers*.

Quando um formulário é submetido (Cafarella et al., 2011; Madhavan et al., 2007), os valores preenchidos nas entradas do formulário são enviados para o servidor através de uma solicitação HTTP que pode usar um dos dois métodos: *get* ou *post*. Com o *get*, os parâmetros são acrescentados à ação e incluídos como parte da URL na solicitação HTTP (por exemplo, http://jobs.com/find?src=hp&kw=chef&st=Any&sort=salary&s=go), que formam as páginas dinâmicas do primeiro tipo. Com o método *post*, os parâmetros são enviados no corpo do pedido HTTP e a URL é simplesmente a ação, em outras palavras, todas as submissões de um formulário têm a mesma URL e a consulta do usuário está embutida na solicitação HTTP (por exemplo, http://jobs.com.br/find). Assim, as URLs obtidas nos formulários que usam *get* são únicas e dependentes dos valores submetidos, enquanto que as obtidas com *post* não são únicas. Estas páginas fazem parte do segundo tipo de páginas dinâmicas. Uma vez que os motores de busca identificam páginas da Web com base em suas URLs, as páginas de resultado de um método *post* são indistinguíveis e, portanto, não são indexáveis.

O conjunto de dados, que não é acessível pelos mecanismos de busca, é conhecido como *Deep Web*. Os problemas relativos à indexação de páginas dinâmicas estão enumerados na próxima seção.

2.1.2. Deep Web

De acordo com Bergman (2001), as pesquisas na Web através de mecanismos de busca tradicionais podem ser comparadas a um arrastão na superfície do oceano. Enquanto uma grande quantidade de informações cai na rede, uma incalculável riqueza de informações ainda continua inalcançável nas

suas profundezas. O principal motivo disso é que os motores de busca tradicionais, na sua maioria, conseguem indexar apenas documentos contidos em páginas HTML estáticas e que estejam ligados a outras páginas através de *hiperlinks*.

Esta parte da Web acessível aos mecanismos de busca compõe a Web superficial (*surface Web*) ou Web convencional, enquanto que a não acessível é conhecida como Web profunda (*Deep Web*).

Após se passarem 12 anos desde que Bergman (2001) expôs a importância da *Deep Web*, as soluções apresentadas para este problema ainda não são efetivas. Segundo Dragut et al. (2012), mesmo que os tamanhos da *surface Web* e da *Deep Web* não sejam conhecidos, existe um consenso na comunidade da Web de que a *Deep Web* é muitas vezes maior do que a *surface Web*. Se considerarmos o conceito mais abrangente de *Deep Web*, como em Rajaraman (2009), seu tamanho pode ser imensurável. De acordo com Rajaraman (2009), além dos dados estruturados, a *Deep Web* inclui outros tipos de objetos, tais como os dados multimídia e documentos com alta frequência de atualização, como é o caso do Twiter, sites de notícias financeiras relativas ao mercado de ações, sites de vendas com promoções por tempo limitado, dentre outros.

As principais abordagens que tentam resolver o problema de acesso aos bancos de dados da *Deep Web* são: Pesquisa Federada (*Federated Search*), Rastreamento da Web Profunda (*Deep Web Crawl)* e Busca por Produtos (*Product Search*). Estas abordagens são apresentadas a seguir, e depois comparadas, ressaltando suas vantagens e desvantagens.

2.1.2.1. Federated Search

A abordagem *Federated Search* (Cafarella et al., 2009; Callan, 2002; He et al., 2005a; He et al., 2005b; Kabisch et al., 2010), que também é conhecida como *Virtual Integration*, é basicamente uma solução de integração de dados aplicada aos dados da *Deep Web*. Em outras palavras, esta solução se baseia na construção de mediadores, potencialmente um para cada domínio de informação, tais como carros, imóveis, oferta de empregos. Inicialmente é feita uma análise de vários formulários disponíveis na internet para cada domínio em questão e o esquema mediado de cada domínio é criado semi-automaticamente ou manualmente. Em

seguida, são gerados os mapeamentos entre as entradas dos formulários e o esquema mediado. As consultas efetuadas pelo usuário podem então ser executadas pelo mecanismo de busca sobre o esquema mediado. A seguir, o mecanismo de busca reformula as consultas e as transfere para cada fonte de dados subjacente. O resultado de cada fonte é então combinado e apresentado para o usuário.

Um exemplo de um motor de busca que implementa a abordagem *virtual integration* e que pode ser encontrado na Web é o Mobissimo (Mobissimo, 2004). O Mobissimo agrega sites voltados para o domínio de viagens. Ele permite que seus usuários comparem os preços de passagens aéreas, hotéis e aluguel de carros através da agregação de resultados de mais de 180 sites de agências de viagens, companhias aéreas e transportes. A agregação é mostrada através de um quadro para a comparação das melhores ofertas encontradas na web. Quando o usuário do site faz a sua seleção ele o redireciona para o respectivo site de vendas ou reservas.

Outro exemplo da abordagem *virtual integration* pode ser encontrado em Nguyen et al. (2010) onde é apresentado o sistema PruSM (Prudent Schema Matching). Nguyen et al. (2010) detalha como o alinhamento[6] de esquemas de formulários Web de múltiplas fontes de informação pode ser feito por similaridade. Um resumo deste sistema pode ser encontrado no apêndice A.

2.1.2.2. Surfacing

A abordagem *Surfacing* ou *Deep Web crawl* (Cafarella et al., 2011; Madhavan et al., 2009; Madhavan et al., 2008; Raghavan & Garcia-Molina, 2001) pode ser considerada uma extensão dos mecanismos de busca tradicionais. A abordagem enfoca a indexação prévia dos resultados relevantes de execuções de formulários HTML. Os formulários HTML são preenchidos automaticamente pelos mecanismos de busca e as consultas são executadas *offline*. Os resultados destas execuções são traduzidos para páginas HTML estáticas, que são então indexadas pelos mecanismos de busca.

[6] O alinhamento de dados nesse contexto corresponde ao casamento de esquemas de formulários Web. Este problema é tratado em diversos processos de gerenciamento de dados, como por exemplo, integração de dados, consulta a diferentes fontes de dados e pesquisa por similaridade.

A abordagem *surfacing* envolve dois desafios técnicos (Cafarella et al., 2011):

- a definição de quais valores devem ser selecionados, para preencher as caixas de texto em um formulário; e
- a definição do número máximo de combinações de valores, para o preenchimento dos formulários com entradas múltiplas, de forma que as consultas executadas retornem um número finito de dados que sejam úteis e distintos - sem respostas em branco - para serem indexados.

Madhavan et al. (2008) descreve a solução implementada pela Google. Os autores abordam o primeiro problema mencionado acima, através do preenchimento automático das caixas de texto reservadas para palavras-chave - que existem na maioria dos formulários HTML - com um conjunto de palavras que o mecanismo de busca identifica como parte de um determinado domínio. Os resultados retornados são analisados pelo mecanismo de busca e novas palavras são extraídas, identificadas em um domínio, e então diferentes palavras-chave são selecionadas para serem testadas novamente.

Com relação ao segundo problema, os autores apresentam um algoritmo capaz de selecionar, dentre o produto cartesiano dos valores candidatos ao preenchimento do formulário com múltiplas entradas, somente um subconjunto que provê um número razoável de resultados distintos e úteis.

2.1.2.3. Busca por Produtos

A abordagem "busca por produtos" (*product search*) é aquela em que o usuário submete previamente seus dados estruturados a um site capaz de integrar informações de um determinado domínio. Esses sites não usam a abordagem *virtual integration* porque a integração só é feita sobre as informações que são previamente submetidas pelo usuário e que seguem um conjunto de regras exigidas pelo site.

Um exemplo típico desta abordagem é o Google Shopping antigo Google Base (Madhavan et al., 2007; Google-Shopping, 2013). No Google Shopping, os donos dos dados devem submeter seus dados através de um arquivo XML, onde cada item deve conter uma identificação única dentro do site, ser classificado segundo uma taxonomia da Google e ser descrito através de um conjunto de

metadados também predefinido pela Google. A taxonomia é bastante simples e contempla diferentes domínios de vendas. O usuário continua fazendo a pesquisa por palavra-chave, mas o módulo de busca tira proveito do dado estruturado.

Os conceitos descritos no item 2.1 constituem a motivação deste trabalho e são importantes para avaliação do status das pesquisas recentes no contexto da *Deep Web*. Estamos propondo uma abordagem diferente que envolve os conceitos apresentados nos itens 2.2 e 2.3.

2.1.2.4. Comparação das Abordagens

As abordagens para o problema de localização de dados da *Deep Web* resumidas nesta seção, possuem uma série de vantagens e desvantagens.

A abordagem *Virtual Integration* que tenta resolver o problema de localização de informações no âmbito da *Deep Web* através da integração de fontes de dados, é uma boa escolha para mecanismos de busca vertical com foco em um determinado domínio. Ela consegue manter a semântica dos dados estruturados e, por isso, tira proveito desta estrutura fornecendo ao usuário consultas ricas e específicas. *Virtual Integration* também consegue agregar resultados de várias fontes de dados e executar buscas mais profundas do que um mecanismo de busca tradicional.

Entretanto, esta abordagem possui os seguintes problemas:

- existem milhões de fontes de dados pertencentes a incontáveis domínios na Web. Construir e gerenciar mapeamentos em tal escala pode ser um desafio, além de ter que ser feito em centenas de idiomas;
- a criação e manutenção de cada mapeamento requerem um grande envolvimento humano;
- o tempo de resposta das consultas pode ser longo.

Apesar do progresso das ferramentas que seguem a abordagem *Surfacing,* considerada uma extensão dos mecanismos de busca tradicionais, elas ainda apresentam as seguintes limitações:

- não abrangem todos os tipos de dados existentes na *Deep Web*, tais como dados geográficos e dados multimídia;

- a semântica dos dados estruturados é perdida quando estes são publicados em páginas HTML estáticas;
- os usuários não podem definir quais dados do seu banco de dados podem ser indexados pelos mecanismos de coleta. Este fato pode obrigar o dono do dado, por razões de segurança, o bloquear a entrada dos motores de coleta no site.

A principal vantagem da abordagem *Surfacing* é que ela tira proveito da tecnologia dos mecanismos tradicionais, usando o ambiente que a grande maioria dos usuários da Web está habituada a trabalhar.

A abordagem *Busca por Produto*, possui vantagens semelhantes às da *Virtual Integration*, com a desvantagem de que o dono dos dados é quem deve adequar seus dados às regras de integração. É importante ressaltar que, nestas soluções, os dados da *Deep Web* só passam a ser visíveis dentro do próprio site. Nem mesmo o Google Shopping oferece a visibilidade dos seus dados através do Google Web.

A tabela 1 resume e compara as abordagens descritas para o problema de visibilidade dos dados da *Deep Web*.

Tabela 1 – Comparação das abordagens apresentadas para a *Deep Web*

	Virtual Integration	**Surfacing**	**Busca por produto**
Abordagem	Integração de dados	Extensão dos mecanismos de busca tradicionais	Integração de dados
Abrangência na *Deep Web*	Limitada (Focada em domínios)	Limitada (Não abrange todos os tipos de dados)	Limitada (Focada em domínios)
Semântica dos dados estruturados	Preservada	Perdida	Preservada
Consultas	Palavra-chave + semântica através da estrutura do dado	Palavra-chave	Palavra-chave + semântica através da estrutura do dado
Atualização	Difícil (pode existir alteração nos esquemas)	Fácil	Difícil (via usuário)
Controle sobre a exposição dos dados	Nenhum	Nenhum	Excelente

A abordagem W-Ray também busca resolver o problema de localização de dados de *Deep Web*, no entanto ela é uma proposta diferente das abordagens *virtual integration* e *surfacing* que utiliza conceitos de dados ligados e linguagem natural descritos a seguir.

2.2. Dados Ligados

A principal finalidade dos *dados ligados* é permitir o compartilhamento de dados estruturados na Web, tão facilmente como os documentos são compartilhados atualmente (Heath & Bizer, 2011).

Os *dados ligados* se baseiam na criação de links tipados entre diferentes fontes de dados publicadas na Web. Isto deve ser feito de tal forma, que o conjunto de dados seja entendido por máquina, que o seu significado seja explicitamente definido e que esteja ligado a outros conjuntos de dados externos que, por sua vez, podem estar ligados com outros conjuntos de dados externos e assim sucessivamente, formando uma grande rede de dados.

Uma vez que as unidades primárias da Web são documentos HTML conectados por hiperlinks não tipados, os dados ligados dependem de documentos que contenham dados em formato RDF[7]. No entanto, ao invés de simplesmente conectar esses documentos, os dados ligados usam o RDF para fazer declarações tipadas que ligam qualquer coisa no mundo. O resultado disto é conhecido como *Web de dados* (Bizer et al., 2009).

Conforme descrito em Bizer et al. (2009), a meta das pesquisas continua sendo a Web Semântica[8], mas os meios para alcançar essa meta são os *dados ligados*.

Sob uma perspectiva de desenvolvimento de aplicações, *Dados Ligados* possuem as seguintes características (Bizer et al., 2009).:

- Os dados são estritamente separados da formatação e aspectos de apresentação.
- Os dados são auto-descritivos. Se um aplicativo que consome *Dados Ligados* encontrar estes dados descritos com um vocabulário desconhecido, o

[7] Uma revisão do RDF*(Resource Description Framework)* pode ser encontrada no Apêndice A

[8] Web Semântica - é uma extensão da Web atual, que permitirá que humanos e computadores trabalhem em cooperação. A ideia principal é categorizar a informação de maneira padronizada, facilitando o seu acesso através de computadores (Breitman, 2005).

aplicativo pode dereferenciar os URIs (Uniform Resource Identifier) que identificam os termos do vocabulário a fim de encontrar a sua definição.

- O uso de HTTP como mecanismo de acesso a dados padronizados e RDF como um modelo de dados padrão facilita o acesso aos dados, se compararmos com APIs da Web que dependem de modelos de dados heterogêneos e de interfaces de acesso.
- A Web de dados é aberta, o que significa que os aplicativos não têm que ser implementados apenas para um conjunto fixo de fontes de dados, mas eles podem descobrir novas fontes de dados em tempo de execução, seguindo os links RDF.

No que se refere ao contexto de *dados ligados*, há alguns aspectos especialmente relevantes como a questão da sua publicação na Web, o uso de vocabulários na Web de dados e o RDFa usado para fornecer uma estrutura aos documentos da Web. Cada um desses tópicos é descrito a seguir.

2.2.1. Publicando dados ligados na Web

Berners-Lee (2006) define os "princípios dos dados ligados", que fornecem uma receita básica para a publicação e conexão de dados usando a infraestrutura da Web, ou seja, mantendo a sua arquitetura e padrões:

1. Usar URIs como nome para as coisas;
2. Usar HTTP URIs de modo que as pessoas possam procurar esses nomes (dereferenciar);
3. Quando alguém procurar um URI, fornecer informações úteis, usando padrões (RDF, SPARQL[9]);
4. Incluir links para outros URIs, para que mais coisas possam ser descobertas;

Cada um desses passos é importante para a abordagem W-Ray, uma vez que ela publica dados estruturados na Web.

Primeiro princípio: *URIs para identificar coisas*

[9] SPARQL - Linguagem de consulta de manipulação de dados no formato RDF. Recomendada pelo W3C em http://www.w3.org/TR/rdf-sparql-query/

De acordo com o primeiro princípio, devem ser identificados não apenas documentos, mas também objetos do mundo real e conceitos abstratos, tais como: pessoas, lugares, carros, relacionamento entre pessoas, uma cor, etc.

Para identificar univocamente objetos do mundo real, pode-se usar padrões de códigos que já são adotados em vários domínios (Bizer et al., 2009). No domínio das publicações, existem os números ISBN[10] e ISSN[11]; para o domínio financeiro, os identificadores ISIN[12]; para produtos ou mercadorias, são frequentemente usados os códigos EAN[13] e EPC[14]; nas ciências biológicas, existem identificadores para os genes, moléculas e substâncias químicas.

Somente quando não existir um padrão definido para o domínio da fonte de dados a ser publicada, o dono da fonte, que é o responsável pela sua geração e manutenção, deve definir o seu próprio identificador.

Segundo princípio: *HTTP URIs para permitir o acesso*

Como o protocolo HTTP é o mecanismo de acesso da Web, consequentemente, os URIs utilizados para identificar univocamente um objeto qualquer, neste contexto, devem segui-lo para que este objeto possa ser localizado na Web. Duas soluções atendem os requisitos de identificação dos objetos do mundo real no protocolo HTTP (Sauermann & Cyganiak, 2008): *303 URIs* e *Hash URIs*.

A primeira solução utiliza o código de retorno HTTP - *303 see others* - para indicar que o recurso solicitado não é um documento típico da Web. Na arquitetura da Web, o URI de um objeto do mundo real não deve retornar o código *200 (OK)* porque não há, de fato, qualquer representação adequada, em HTML, para um objeto do mundo real. No entanto, para fornecer informações sobre esses objetos, o W3C propõe, como uma solução, o redirecionamento para um documento que contenha informações sobre o objeto que está sendo

[10] ISBN - International Standard Book Number - sistema identificador único para livros e publicações não periódicas. http://www.isbn-international.org/

[11] ISSN - International Standard Serial Number - identificador de publicações seriadas aceito internacionalmente. Seu uso é definido pela norma técnica ISO 3297:2007. http://www.issn.org/

[12] ISIN - International Securities Identification Number - sua estrutura é definida pela ISO 6166 para identificar títulos, papéis comerciais e ações. http://isin.org/

[13] EAN-13 European Article Number - padrão europeu de código de barras para a identificação de produtos no varejo e atacado.

[14] EPC - Electronic Product Code - identificador único universal para cada objeto físico em qualquer lugar do mundo.

procurado. Uma vez que 303 é um código de redirecionamento, o servidor poderá fornecer a localização de um documento que representa o objeto. Ao fazer isso, evitamos ambiguidades entre o objeto e o recurso que o representa.

A segunda solução utiliza "*hash URIs*" para os recursos que não são documentos. URIs podem conter um fragmento, que é uma parte especial separada do resto do URI por um caractere tralha ("#"). Quando um cliente quer recuperar um *hash URI*, o protocolo HTTP retira a parte do fragmento antes de solicitar o URI do servidor. Isto significa que, um URI que inclui um caractere tralha, não pode ser recuperado diretamente e, por conseguinte, não necessariamente identifica um documento da Web. Mas podemos usá-los para identificar recursos que não são documentos, sem criar ambiguidade.

Terceiro princípio: *URI com informações úteis utilizando RDF*

O padrão utilizado para publicar dados estruturados na Web é o RDF (Manola & Miller, 2004). O formato para a serialização dos dados ligados é o RDF/XML (Becktt, 2004). Em situações onde os seres humanos necessitam manipular os dados descritos em RDF, pode-se usar o formato Turtle (Beckett & Berners-Lee, 2011).

Uma alternativa que está sendo muito utilizada é serialização dos dados ligados através do formato RDFa (Adida et al., 2012). Neste caso, as triplas RDF são embutidas na linguagem HTML e o atributo *about* do RDFa pode ser utilizado para atribuir URIs para entidades, permitindo assim que outros provedores de dados liguem seus dados aos publicados em RDFa.

Essa é alternativa usada pela abordagem W-Ray proposta nesta tese.

Quarto princípio: *Links entre URIs para que mais coisas possam ser descobertas*

Os links RDF permitem que as aplicações de clientes naveguem entre os dados para descobrir novos dados. A ideia é formar uma rede de dados da mesma forma que a Web convencional é uma rede de documentos. Três tipos de link podem ser usados (Heath & Bizer, 2011):

– Links de relacionamento - os recursos de uma fonte de dados podem ser relacionados a recursos em outra fonte de dados.

– Links de identidade - quando o mesmo recurso é definido por duas fontes de informação na Web, deve-se criar um link de identidade entre eles. Neste caso, os recursos são *aliases*. Exemplo de um link de identidade em OWL[15] (Ontology Web Language) (McGuinness & Harmelen, 2004):

<http://www.dave−smith.eg.uk#me><http://www.w3.org/2002/07/**owl#sameAs**><http://biglynx.co.uk/people/dave−smith>

– Links entre vocabulários - são links que podem ser criados entre os vocabulários para proporcionar o reuso de vocabulários conhecidos e confiáveis. Na próxima seção é apresentado um resumo sobre vocabulários na Web de dados.

2.2.2. Vocabulários na Web de dados

Comunidades diferentes têm preferências específicas sobre os vocabulários para publicação de dados na web. A Web de dados aceita diferentes vocabulários que podem ser utilizados em paralelo. No entanto, é considerada uma boa prática a reutilização de termos de vocabulários bem conhecidos e confiáveis (Bizer et al., 2009). O reuso aumenta a probabilidade de localização e consumo dos dados por aplicativos que podem estar ajustados para vocabulários bem conhecidos, sem a necessidade de pré-processamento dos dados ou de modificação na aplicação.

De acordo com Heath & Bizer (2011), na seleção de vocabulários para reutilização, os seguintes critérios devem ser aplicados:

1. Uso e consumo – o vocabulário está difundido? Ao usar este vocabulário, a fonte de dados vai ficar mais, ou menos, acessível através das aplicações existentes de dados ligados?
2. Manutenção e administração – o vocabulário será ativamente mantido de acordo com um claro processo de administração? Quando e como as atualizações são feitas?
3. Cobertura – o vocabulário tem cobertura suficiente dos dados para justificar a adoção dos seus termos e compromissos ontológicos?

[15] Ver conceito Apêndice A.

4. Expressividade – o grau de expressividade no vocabulário é adequado ao conjunto de dados e ao cenário da aplicação? É muito expressivo, ou não é expressivo o suficiente?

Abaixo estão relacionados alguns exemplos de vocabulários que devem ser reutilizados sempre que possível. Estes vocabulários foram indicados em Heath & Bizer (2011) porque possuem uma utilização genérica ou porque, mesmo que desenvolvidos para domínios específicos, são altamente conhecidos:

– Dublin Core Metadata Initiative (DCMI) - define atributos como *title, creator, date* e *subject* para descrever documentos, artigos, livros;
– Friend-of-a-Friend (FOAF) - define termos para descrever pessoas, suas atividades e relações com outras pessoas e objetos;
– Semantically-Interlinked Online Communities (SIOC) - descreve aspectos de sites de comunidades *online* como *users, posts* e *forums*;
– Music Ontology - define termos relacionados com música, como artistas, álbuns, trilhas, interpretações e arranjos;
– Bibliographic Ontology (BIBO) - fornece conceitos e propriedades para descrever citações e referências bibliográficas;
– Geonames Ontology - fornece conceitos e propriedades para descrever características geográficas;
– WordNet Ontology - descreve os termos do banco de dados WordNet[16] em OWL.

Se os vocabulários existentes não fornecem os termos necessários para os dados que estão sendo publicados, uma nova terminologia específica deve ser definida. Esta nova terminologia deve ser feita utilizando-se HTML URIs para a identificação dos termos de forma a permitir que outros clientes possam criar mapeamentos para este novo vocabulário.

Para a criação de um vocabulário, é necessário a utilização de uma linguagem que consiga expressar alguns conceitos, tais como, descrever classes de coisas do mundo e como estas coisas estão relacionadas umas com as outras em domínios específicos. O padrão para descrição dos dados ligados na Web é o RDF. No entanto, o RDF não oferece funcionalidades para tal tarefa. Esta

[16] WordNet é um banco de dados léxico onde substantivos, verbos, adjetivos e advérbios são organizados em conjuntos de sinônimos, cada um representando um conceito léxico.

funcionalidade é fornecida pelas taxonomias, vocabulários e ontologias que podem ser expressas em RDFS (RDF Vocabulary Description Language Schema, também conhecido como *RDF Schema* ou esquema RDF) (Brickley & Guha, 2004) e em OWL (McGuinness & Harmelen, 2004).

RDFS e OWL são usados quando existe a necessidade de uma expressividade maior nos relacionamentos entre os dados, como por exemplo, quando é necessário representar um relacionamento de herança entre os termos do vocabulário (Heath & Bizer, 2011). Quando combinados com um motor de raciocínio adequado, os esquemas de dados em RDFS e OWL permitem que sejam feitas inferências sobre o dado através de relacionamentos implícitos.

No contexto dos dados ligados, muitas vezes o RDFS não é suficiente para expressar vocabulários. No entanto, a linguagem OWL oferece primitivas, que no contexto dos dados ligados, devem ser usadas com frequência, como *owl:sameAs*, usada para afirmar que duas URIs identificam o mesmo recurso. Outras primitivas da OWL também são importantes para aumentar a interoperabilidade entre conjuntos de dados diferentes (Heath & Bizer, 2011):

- as primitivas *owl:equivalentClass* e *owl:equivalentProperty*, quando combinadas com *rdfs:subClassOf* e *rdfs:subPropertyOf*, fornecem um poderoso mecanismo para a definição de mapeamentos entre termos de vocabulários diferentes.
- as primitivas de modelagem OWL *owl:InverseFunctionalProperty* e *owl:inverseOf* também são úteis no contexto da Web de dados. No caso da propriedade *owl:inverseOf*, o criador de um vocabulário pode afirmar que uma propriedade é a inversa ou simétrica.

Os seguintes aspectos devem ser levados em consideração na definição de vocabulários (Heath & Bizer, 2011):

- Reutilize vocabulários existentes ao invés de reinventar termos.
- Defina novos termos apenas dentro de um *namespace* onde você tenha o controle.
- Use primitivas de RDFS e OWL para relacionar termos novos com termos de vocabulários já existentes.
- Aplique os "princípios de dados ligados" de forma rigorosa tanto na geração dos vocabulários como nas fontes de dados. As URIs dos termos do seu

vocabulário devem ser dereferenciáveis para que os aplicativos de dados ligados possam localizar a sua definição.

– Documente cada termo novo com etiquetas e comentários. As tags *rdfs:label* e *rdfs:comment* são projetadas para este fim.
– Defina apenas as coisas que realmente possuem importância e que podem ser úteis para outros consumidores de dados.

Dentre os vocabulários disponíveis, há um de especial interesse que é o Dublin Core. Ele pode ser usado para a definição de metadados que devem acompanhar a publicação dos dados ligados.

Embora a criação de metadados não faça parte explicitamente dos "princípios dos dados ligados" definidos em Berners-Lee (2006), ela é recomendada em Bizer et al. (2009).

Os metadados devem ser inseridos nos dados publicados para que os consumidores de dados possam avaliar a qualidade do dado e decidirem se podem confiar no dado publicado. Um conjunto mínimo de metadados deve acompanhar os dados publicados, tais como, sua data de criação e o método de criação. Os termos do vocabulário Dublin Core podem ser utilizados para esta finalidade.

O processo de mapeamento de um banco de dados relacional em triplas RDF é particularmente relevante no contexto deste trabalho porque é um dos procedimentos que compõem a abordagem W-Ray.

2.3. Mapeamento de Bancos de Dados Relacionais para RDF (RDB-to-RDF)

O mapeamento de um banco de dados relacional em triplas RDF, que inclui o esquema e os dados, é conhecido como RDB-to-RDF ou triplificação (Prud'hommeaux & Hausenblas, 2010).

Processos de triplificação seguem: uma abordagem de mapeamento direto, também conhecida como mapeamento sintático, ou uma abordagem de mapeamento para ontologias de domínio, também conhecida como mapeamento semântico.

Os processos de triplificação que seguem a primeira abordagem criam uma nova ontologia baseada no esquema de banco de dados relacional e, portanto, são fáceis de automatizar, porque envolvem regras de transformação diretas, tais como tabelas-para-classes e colunas-para-propriedades.

Os processos que seguem a segunda abordagem mapeiam o esquema de banco de dados para uma ontologia de domínio já existente. Nesse caso, o mapeamento semântico pode ser automatizado ou manual. Se automático, se baseia na suposição de que os dados podem fornecer informações importantes sobre o conteúdo e significado do esquema. Apesar de várias tentativas de automatização, estes processos ainda requerem intervenção humana. Este tipo de mapeamento é importante porque gera dados ligados.

De acordo com Ghawi & Cullot (2007), os processos de triplificação incluem duas etapas: (1) mapear o esquema de banco de dados para um esquema RDF; (2) Mapear o conteúdo do banco de dados para triplas. A etapa (1) consiste em representar os conceitos do esquema de banco de dados em termos de classes e propriedades RDF, e em definir um conjunto de regras que mapeiam os dados relacionais em triplas RDF. Este mapeamento pode ser implementado como um processo *batch*, que gera um *dump* de toda a base de dados em triplas RDF e as armazena em um repositório próprio, ou como uma interface que suporta pesquisas no banco de dados subjacente e que retorna triplas como respostas para as consultas, conhecida como mapeamento *virtual*.

Uma linguagem de mapeamento padrão, R2RML, foi publicada recentemente em (Das et al., 2012).

2.3.1. Ferramentas de mapeamento RDB-to-RDF

Atualmente, existem várias ferramentas de mapeamento RDB-to-RDF. Em Sahoo et al., (2009) pode ser encontrada uma revisão detalhada. Dentre as ferramentas, as seguintes ferramentas são especialmente interessantes sob a perspectiva de comparação ou mesmo como inspiração para a abordagem W-Ray: D2R Server que é uma ferramenta com uma abordagem clássica com mapeamento semântico manual; RDBToOnto que é capaz de gerar um RDFS mais preciso no que se refere a hierarquia de classes e, portanto, parte do seu mapeamento semântico é automático e parte manual; e Triplify, que executa o mapeamento

através de *views* sobre o banco de dados usando o mapeamento semântico manual. Cada uma delas é resumida a seguir.

2.3.2. D2R Server

D2R Server é uma ferramenta de mapeamento de dados de banco de dados relacionais para RDF que usa a linguagem declarativa denominada D2RQ (Bizer & Seaborne, 2004) para a geração automática dos arquivos de mapeamento. Foi criada por pesquisadores da Universidade Livre de Berlim (em alemão: *Freie Universität Berlin*, "FU Berlin") e possui código aberto.

A ferramenta continua disponível para download a partir do site http://d2rq.org. Uma recente versão (Eisenberg & Kanza, 2012) inclui a possibilidade de atualização dos dados através da SPARQL. O mapeamento RDB-to-RDF pode ser executado a partir de dados armazenados nos sistemas gerenciadores de bancos de dados Oracle, MySQL, PostgreSQL, SQL Server, HSQLDB e Interbase/Firebird.

A abordagem oferece tanto a possibilidade de geração de triplas RDF virtuais, ou seja, o mapeamento é feito, mas as instâncias do Banco de Dados não são materializadas em triplas RDF, quanto à possibilidade de geração de um *dump* de toda a base de dados em triplas RDF, se necessário. Para o acesso às triplas RDF virtuais, o arquivo de mapeamento é usado para traduzir o SPARQL em consultas SQL. O arquivo de mapeamento pode ser customizado pelo usuário para permitir o reuso de ontologias no processo de mapeamento.

Ao executar o serviço D2R Server, são passados os parâmetros da base de dados que se deseja triplificar ou visualizar virtualmente, tais como endereço, porta, nome, usuário etc. A partir daí, o D2R Server acessa o esquema do banco e constrói um arquivo de mapeamento, convertendo tabelas em classes, colunas em propriedades e, nas tabelas que possuem chave estrangeira, são realizados *joins*, de maneira que seja possível visualizar estes relacionamentos na forma de *object properties*.

O arquivo de mapeamento gerado inclui todas as tabelas e colunas da base, ou seja, o usuário não pode selecionar previamente os dados que deseja mapear para RDF. Consequentemente, nem todas as informações mapeadas são úteis para visualização e para a geração da ontologia. Além disso, todas as classes e

propriedades são nomeadas pelo vocabulário criado para a base em questão. Por isso, torna-se necessário "enxugar" o mapeamento, ou seja, alterar o vocabulário utilizado para as tabelas e colunas do banco de dados para nomes com mais semântica ou alinhar com os termos de vocabulários já existentes.

A Figura 1 mostra a comparação de um pequeno trecho de um mapeamento original executado pelo D2RServer com o mesmo mapeamento já modificado pelo usuário. Os textos em azul indicam as principais diferenças entre os mapeamentos. No mapeamento original, cada *PropertyBridge* corresponde a uma coluna na tabela e equivalem a uma propriedade RDF. Assim, no mapeamento original foram criadas duas propriedades, uma para a coluna *firstname*, outra para a coluna *lastname*. Para se efetuar um alinhamento destas propriedades com a propriedade *name* da ontologia FOAF (Friend-of-a-Friend), as seguintes modificações foram adicionadas:

- Criação de apenas uma *PropertyBridge*, com nome *map:name*
- Alteração da propriedade para *foaf:name*
- Atribuição da propriedade como *rdfs:label*
- Formatação do valor para exibir nome e sobrenome.

Para executar estas alterações o usuário é obrigado a conhecer a linguagem de mapeamento da abordagem.

Mapeamento original

```
map:users_firstname a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:users;
  d2rq:property vocab:users_firstname;
  d2rq:propertyDefinitionLabel "users
      firstname";
  d2rq:column "users.firstname";
  .
map:users_lastname a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:users;
  d2rq:property vocab:users_lastname;
  d2rq:propertyDefinitionLabel "users
      lastname";
  d2rq:column "users.lastname";
  .
```

Mapeamento modificado

```
map:name a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:Users;
  d2rq:property foaf:name;
  d2rq:property rdfs:label;
  d2rq:pattern "@@users.firstname@@
      @@users.lastname@@";
  .
```

Figura 1 - Alteração do mapeamento para o nome do usuário

2.3.3. RDBToOnto

RDBToOnto é uma ferramenta que permite mapear automaticamente dados de bancos de dados relacionais para RDF. A ferramenta possui seu código aberto

e foi desenvolvida através do projeto *Transitioning Applications to Ontologies (TAO)* que é formado por um grupo de universidades e empresas européias.

O mapeamento RDB-to-RDF pode ser executado a partir de dados armazenados em sistemas gerenciadores de bancos de dados (MySQL, Oracle, Microsoft Access) ou de dados armazenados em tabelas Excel. O mapeamento é feito através de *dump* de toda a base de dados em triplas RDF. A ferramenta é oferecida no site: http://www.tao-project.eu/researchanddevelopment/demosanddownloads/RDBToOnto.html

A ferramenta foi apresentada em Cerbah, (2008) juntamente com uma discussão sobre como, automaticamente, tirar proveito dos dados armazenados em banco de dados com o objetivo de gerar uma ontologia mais precisa. Mais especificamente, a ferramenta tenta explorar a capacidade de se identificar o conceito de hierarquia de classes "escondido" na estrutura dos dados através da categorização automática de instâncias das colunas do BD. Por exemplo, considere uma tabela de *Produtos* que contenha as instâncias sal e feijoada. A instância **sal** pode pertencer à categoria *condimentos* e **feijoada** à categoria *comidas típicas*.

RDBtoOnto também fornece ao usuário a possibilidade de melhorar o projeto de mapeamento através da sugestão de novos nomes para as tabelas e colunas. As sugestões podem ser automáticas ou manuais. As automáticas são fruto de métodos de aprendizado de máquina que são executados sobre os dados do BD. Estas sugestões geradas automaticamente e denominadas *constraints* (que não são as *constraints* comumente conhecidas em um banco de dados) são oferecidas aos usuários que podem manualmente alterá-las antes do mapeamento. A Figura 2 mostra a interface do usuário preenchida com um exemplo. Repare que na segunda linha do combobox ***Local Constraints*** podemos ver que o nome original da tabela é ***movi*** e que o nome sugerido para a classe mapeada é ***Inspection_Type***.

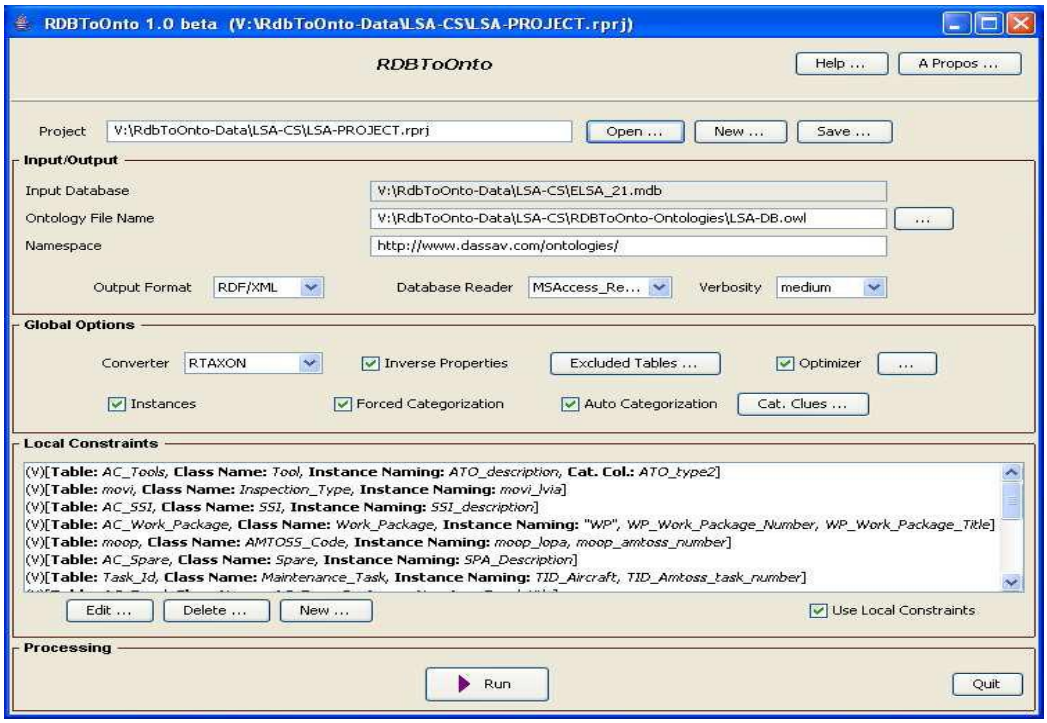


Figura 2 – Interface do usuário da abordagem RDBtoOnto

2.3.4.Triplify

Triplify é uma ferramenta de mapeamento RDB-to-RDF que possui código aberto e que foi desenvolvida pelo grupo de pesquisa *Agile Knowledge Engineering and Semantic Web (AKSW)* da Universidade Leipzig - Alemanha.

A ferramenta Triplify (Auer et al., 2009) converte os resultados de *views*, criadas pelo dono do dado, sobre um banco de dados relacional para o formato RDF. A ferramenta continua disponível para download a partir do site http://triplify.org.

O Triplify não possui uma linguagem de mapeamento proprietária. A abordagem tira proveito de dicas passadas através da linguagem SQL para executar o mapeamento. O processo de mapeamento pode ser executado sob demanda ou antecipadamente criando-se um *dump* das *views* sobre o banco de dados.

A seguinte estratégia de mapeamento é utilizada para transformar os resultados da consulta SQL para o modelo de dados RDF:

– **Mapeamento para vocabulários existentes** – O Triplify utiliza "*ALIASES*" criados no SQL para fazer o mapeamento com vocabulários existentes. O seguinte exemplo de uma consulta SQL:

"SELECT id, name AS 'foaf:name' from users;"

O "*ALIAS*" usado para a coluna *name* da tabela *users* sugere o alinhamento com a propriedade *name* da ontologia FOAF.

- **Object Properties** - O mapeamento para *object property* é possível quando o usuário insere explicitamente na consulta SQL que uma determinada coluna será mapeada como *object porperty* em RDF. Para isso o Triplify usa uma técnica de dicas através da cláusula **"AS"** do SQL para receber informações adicionais de mapeamento. A dica no caso de *object property* é usar o separador '->'. A Figura 3 apresenta um exemplo de uma consulta SQL de entrada para o Triplify. Neste exemplo, será criada uma *object porpety* denominada ***belongsToCategory*** cujo domínio será a classe **Produtos** (criada pelo mapeamento da tabela **Produtos)** e cujo range será definido pela classe **Categoria** (criada pelo mapeamento da tabela **Categoria)**.
- **Datatype Properties** - Para recuperar o tipo de dados de uma determinada coluna e criar as *datatype properties* do RDF com os tipos de dados utilizados em XML Schema Definition (XSD), a mesma técnica de dicas é acrescentada na cláusula **"AS"** do SQL para os que os nomes de coluna possam instruir o Triplify sobre o mapeamento dos tipos de dados. No exemplo da Figura 3 o separador '^^' é usado com este objetivo.
- **Tags de identificação de idioma** – Em todas as cláusulas **"AS"** do SQL podem ser inseridas dicas com o separador "@" para marcar o idioma.

```
SELECT id,
     price AS 'price^^xsd:decimal',
     desc AS 'rdfs:label@en',
     cat AS 'belongsToCategory->category'
FROM products;
```

Figura 3 – Trecho de um SQL de criação de uma *view*

2.3.5. Comparação das Ferramentas

As ferramentas de triplificação devem considerar pelo menos três problemas:

- o primeiro envolve quais dados devem ser triplificados dentre todos os dados armazenados no banco de dados. As ferramentas apresentadas, com exceção

do Triplify que trabalha com o conceito de *views* de banco de dados, mapeiam todo o BD usando uma linguagem própria de mapeamento. Ao final do processo, o usuário pode corrigir mapeamentos incorretos ou retirar dados que não devem ou não podem ser mapeados. Esta situação inclui um trabalho manual, que nem sempre é explicitado pelos autores das ferramentas, que é a edição do mapeamento gerado. Além disso, obriga o usuário a aprender uma nova linguagem que não é SQL nem RDF para executar a tarefa de edição.

– o segundo problema é como criar um esquema RDF para o dado. Todas as ferramentas apresentadas fazem o mapeamento *table-to-class* e *column-to-property*. Os relacionamentos n-m identificados são mapeados em forma de *object properties*. Para os relacionamentos n-m com atributos são criadas: uma classe referente à tabela que representa o relacionamento e *object properties* referentes às chaves estrangeiras. Nem sempre propriedades inversas são criadas.
– o terceiro se resume em como suportar o reuso de vocabulários. A ferramenta Triplify permite que o usuário faça o alinhamento manual durante o processo de geração da *view*. No RDBtoOnto, o alinhamento pode ser feito através de restrições definidas pelo usuário. Nas outras ferramentas, o alinhamento só pode ser executado alterando-se a linguagem de mapeamento.

A Tabela 2 oferece um resumo para comparação das ferramentas de mapeamento RDB-to-RDF.

Tabela 2 – Comparação das abordagens RDB-to-RDF

	Triplify	D2RQ	RDBToOnto
Tipo de Mapeamento (Virtual ou Dump)	Ambos	Ambos	Dump
Linguagem de Consulta (SPARQL, SPARQL to SQL)	Nenhuma	Ambas	SPARQL
Linguagem de Mapeamento	SQL (com dicas)	Própria	Regras de restrições
Seleção dos dados para triplificação	Via SQL	Via alteração da linguagem de mapeamento	Via alteração das regras de mapeamento
Alinhamento automático de ontologias	Nenhum	Nenhum	Nenhum
Abordagem de Mapeamento	table-to-class	table-to-class	table-to-class
Inovação	Trabalha com *views* sobre o banco de dados para permitir a seleção, alinhamento e mapeamento dos dados	Mapeamento tradicional	Cria hierarquia de classes através da categorização dos dados

2.4. RDFa

Conforme mencionado anteriormente, uma alternativa que está sendo muito utilizada para a serialização dos dados ligados é o uso do formato RDFa.

Nos últimos anos, o conteúdo da web tem sido cada vez mais consumido por máquinas que esperam certa quantidade de dados estruturados. Os mecanismos de busca, tais como o Yahoo! (Yahoo!, 2008) e o Google (Google-WebmasterBlog, 2009), começaram a fornecer resultados mais ricos, extraindo detalhes estruturados a partir das páginas Web. A tecnologia chave por trás deste crescimento é a capacidade de adicionar dados estruturados diretamente em páginas HTML.

RDFa (Resource Description Framework in Attributes) (Adida et al., 2012) consiste em embutir informações estruturadas em arquivos HTML, através de triplas RDF, possibilitando semântica para as máquinas em documentos destinados a humanos.

O RDFa pode ser usado para expressar qualquer tipo de informação dentro de um documento HTML. Se o documento possui informações provenientes de um banco de dados, então estas informações também podem ser expressas em RDFa.

Os atributos relevantes na utilização do RDFa em HTML são os seguintes:

- *about*: deve conter o URI de um indivíduo. Caso não esteja presente, subentende-se que o URI é o do próprio documento HTML;
- *rel*: deve conter o URI de uma *object property* do RDF que relaciona dois indivíduos;
- *href, src* ou *resource*: especificam o URI do objeto RDF. Neste caso, também é válido o atributo *about*;
- *property*: define a *datatype property* do RDF associada a um indivíduo;
- *content*: atributo opcional que permite definir o objeto RDF, no caso de tipos literais;
- *datatype*: atributo opcional que permite definir o tipo do dado associado a um indivíduo.

2.5. Sistemas de Geração de Linguagem Natural

A abordagem W-Ray se baseia na criação de um conjunto de sentenças para auxiliar o processo de tornar visíveis dados da *Deep Web*.

Os Sistemas de Geração de Linguagem Natural (LN) recebem como entrada um conjunto de informações e/ou um conjunto de consultas e produzem automaticamente uma linguagem compreensível ao ser humano. Como aplicações de geração automática de textos, podemos citar: a documentação automática de sistemas; a elaboração automática de cartas; a geração de relatórios; a tradução de linguagens formais de criação de ontologias para linguagem natural.

Os sistemas de geração de LN possuem três fases principais (Matthiessen & Bateman, 1991):

1. Determinação do conteúdo (“o que dizer?”) – consiste em selecionar os elementos de conteúdo relevantes para satisfazer os objetivos da comunicação. Na maioria das aplicações práticas, a função do texto deve ser o aspecto mais importante. Por isso, a maior preocupação concentra-se em “o que dizer”.
2. Organização do texto (“onde dizer?”) – consiste em agrupar os elementos de conteúdo selecionados em unidades linguísticas, definindo também as dependências hierárquicas e lineares entre essas unidades dentro de uma estrutura maior. Esta etapa se decompõe em:
   a. organização do discurso – onde a unidade linguística é a oração e a estrutura maior é o texto;
   b. organização das frases, onde a unidade linguística é o constituinte sintático e a estrutura maior é a oração.
3. Produção do texto (“como dizer?”) – que se subdivide em:
   a. Lexicalização - consiste em selecionar as palavras raízes e as estruturas temáticas para expressar cada elemento de conteúdo: escolha de palavras de classes abertas, tais como verbos, substantivos, adjetivos e advérbios; e escolha de estruturas temáticas, tais como agente, paciente e instrumento;
   b. Realização Sintática (etapa final) – inclui o tratamento sintático e morfológico. É subdividida em:
      i. Mapeamento dos papéis temáticos de cada frase para os elementos de superfície da frase como sujeito, objeto, complemento, adjunto;
      ii. Aplicar regras de sintaxe, como concordância entre o sujeito e o verbo;
      iii. Escolher as palavras de classes fechadas (pronomes, artigos, conjunções);
      iv. Flexionar as palavras raízes de classes abertas (ex. flexionar o verbo “comer” para a sua forma no passado “comeu”);
      v. Linearização - linearizar a árvore sintática em uma cadeia de palavras, atravessando esta árvore em profundidade e da esquerda para a direita.

Normalmente, um gerador se concentra em um subconjunto das tarefas listadas acima, implementando-as em arquiteturas com vários graus de modularidade. Um gerador completo, com a mais simples das arquiteturas totalmente modular, teria um componente especializado para cada uma dessas

tarefas e esses componentes seriam organizados em uma estrutura de níveis, onde o fluxo de informação seria unidirecional, na seguinte ordem:

- determinador de conteúdo;
- planejador de discurso;
- planejador de frases;
- lexicalizador;
- realizador sintático;

Nessa estrutura, cada componente recebe sua entrada do componente anterior a ele e passa sua saída para o componente seguinte.

A seguir será discutido o mapeamento de um arquivo RDF para LN e alguns trabalhos relacionados à nossa solução proposta.

2.5.1. Geração de LN a partir de um arquivo RDF

A tarefa de geração de LN a partir de um arquivo RDF ou de um arquivo OWL (RDF-to-LN ou OWL-to-LN) pode ser vista como uma simplificação de um sistema gerador de LN, onde as informações de entrada não são consultas ou perguntas, mas um conjunto de dados estruturados em RDF.

Na verdade, o uso de RDF facilita a geração de texto. Considerando-se que a entrada de um gerador de textos é uma fonte de dados em RDF, pode-se observar os seguintes aspectos:

– Existem semelhanças entre a linguagem do RDFS e a linguagem natural;
– Existem textos embutidos no RDFS;
– Pode-se formar frases, combinando os textos embutidos no RDFS com os indivíduos.
– uma tripla RDF (S, P, O) pode gerar uma sentença da forma "S tem P com valor O".

Com relação aos esquemas RDF, são fontes de texto:

– nomes de classes - que podem ser interpretados como substantivos, que funcionam como sujeitos ou objetos na frase.

– nomes de *object properties* - que funcionam como verbos transitivos;
– nomes de *datatype properties* - que funcionam como adjetivos, advérbios ou substantivos.

Em suma, o próprio esquema RDFS pode ser considerado como uma tarefa de redação onde, agregando-se os indivíduos, podemos criar novas sentenças, particularizando as sentenças relativas aos metadados. É possível observar isso no exemplo da Figura 4 e Figura 5.

....
O bioma abrange um percentual da área localizada na unidade da federação.
....

Figura 4 - Frase formada apenas com metadados do RDFS

....
O bioma "*Caatinga*" abrange *48* por cento da área localizada na unidade da federação "*Alagoas*".
....

Figura 5 - Frase formada com metadados e indivíduos

2.5.2. Trabalhos relativos à Linguagem Natural

Três abordagens envolvendo a verbalização de dados estruturados em OWL inspiraram as soluções adotadas na abordagem W-Ray. São eles: ACE (Attempto Controlled English), NIBA (Natural Language Information Requirements Analysis) e Swoop.

2.5.2.1. ACE

A ACE (Attempto Controlled English) (Fuchs et al., 2008) é uma linguagem natural controlada que foi desenvolvida através do projeto de pesquisa denominado *Attempto* da Universidade de Zurique – Suíça. Mais precisamente, a ACE é um subconjunto da língua inglesa, onde cada sentença deste subconjunto é interpretada sem ambiguidade, ou seja, uma sentença ACE pode ser traduzida para lógica de primeira ordem.

Com o objetivo de fornecer interoperabilidade entre ACE e linguagens da Web Semântica, o projeto *Attempto* desenvolveu uma ferramenta *"OWL Verbalizer"* que traduz a linguagem OWL para a linguagem ACE. O principal

objetivo desta ferramenta é simplificar a visualização e edição de bases de conhecimento em OWL que possuem sintaxe complexa oferecendo expressividade em LN e ao mesmo tempo mantendo seu entendimento através de máquinas.

No verbalizador de OWL em linguagem ACE todos os axiomas OWL são avaliados, as *object properties* são mapeadas para verbos e as classes para nomes próprios. Sua implementação foi feita em Prolog e executa o mapeamento OWL-to-ACE em três passos: (1) Os axiomas são reescritos enxugando a OWL. Por exemplo, são removidos os axiomas que definem o domínio e o range. (2) A estrutura dos axiomas *SubClassOf* é levemente alterada. Por exemplo, operadores tais como *IntersectionOf* e *UnionOf* são reordenados para que elementos estruturalmente mais simples sejam tratados pela ferramenta em primeiro lugar. (3) Os axiomas modificados são diretamente mapeados para ACE através da aplicação de regras DSG (Definite Clause Grammar)[17]. O propósito dos dois primeiros passos é aumentar a legibilidade das sentenças. Uma demonstração desta ferramenta pode ser encontrada em: *http://attempto.ifi.uzh.ch/site/docs/owl_to_ace.html*

O verbalizador ACE não utiliza o recurso das anotações existentes para as propriedades e classes em *rdfs:label*. O exemplo a seguir foi retirado de Fliedl et al. (2010) e mostra uma *object propety* definida em uma ontologia criada para o domínio de *Vinhos*

```
<owl:ObjectProperty rdfid:ID=”madeFromGrape”
      <rdfs:domain rdf:resource= “#Wine”/>
      <rdfs:range rdf:resource=”#WineGrape”/>
</owl:ObjectProperty>
```

e o seu uso com a cardinalidade mínima entre o domínio “Wine” e o range “WineGrape”

```
 <rdfs:subClassOf>
     <owl:Restriction>
         <owl:onProperty rdf:resource=”#madeFromGrape”/>
         <owl:minCardinality
               rdf:datatype=http://www.w3.org/2001/XMLSchema#nonNegativeInteger>1
         </owl:minCardinality>
     </owl:Restriction>
```

[17] É uma maneira de expressar uma gramática, tanto para as linguagens naturais como para as formais, numa linguagem de programação lógica como o Prolog.

```
</rdfs:subClassOf>
```

resulta na seguinte tradução em ACE:

"Every Wine madeFromGrape at least 1 thing".

A frase acima não possui uma legibilidade correta, uma vez que o termo *madeFromGrape foi* utilizado exatamente como descrito no fragmento OWL.

A principal vantagem do mapeamento de OWL-to-ACE é fornecer uma sintaxe alternativa para a OWL. Uma sintaxe legível em LN pode ser atraente para muitas pessoas que desejam entender o que uma ontologia representa, mas não são especialistas na tarefa de geração e manipulação de OWL. Sua sintaxe é destinada principalmente para bases estruturadas e semanticamente complexas, como OWL, em que métodos visuais e sintaxes tradicionais não conseguem proporcionar interfaces amigáveis para os usuários.

2.5.2.2. NIBA

NIBA (Natural Language Information Requirements Analysis) é um projeto de pesquisa patrocinado pela *Klaus-Tschira-Foundation* em Heidelberg - Alemanha cujo objetivo é trabalhar com a extração de modelos conceituais de textos em LN.

O projeto NIBA propõe uma abordagem de mapeamento OWL-to-LN (Fliedl et al., 2010) cujo objetivo é tratar os problemas linguísticos existentes nestes mapeamentos. A abordagem é focada nos seguintes pontos: o desenvolvimento de um conjunto de diretrizes para a padronização de *labels* em OWL; a filtragem de padrões linguísticos; e a criação de um conjunto de regras para a geração de sentenças em LN que explicitam os conceitos da OWL.

O conjunto de diretrizes para a criação padronizada de anotações em *rdfs:label* para classes e propriedades são:

- Todos os nomes devem estar na língua inglesa;
- Se um *label* possui mais de um termo, uma letra maiúscula deve ser usada como delimitador entre os termos;
- Não são permitidas abreviações;
- Nomes devem estar no singular;
- Siglas devem ser escritas como substantivos normais começando com uma letra maiúscula.

- Nomes de propriedade devem começar com letras minúsculas;
- Nomes de classes e indivíduos devem começar com letras maiúsculas.

Para a filtragem de padrões linguísticos são definidas cinco regras:

- Se os labels começam com *"has"* e tem a forma = *"has" [+ Adjetivo] + Substantivo"* então são mapeados como propriedade;
- Se os labels forem especificados por: substantivos compostos, [adjetivo] + substantivo, URLs ou nomes próprios (simples ou compostos) são mapeados como classes ou indivíduos;
- Se um label começa com *"is"* e tem a forma = *"is" [+ Adjetivo/Particípio/substantivo] [+Preposição] [+Substantivo]* então são mapeados como propriedade;
- Se os labels começam com o verbo na terceira pessoa e tem a forma = Verbo [+Preposição]+ Substantivo [+Preposição] então são mapeados como propriedade;
- Se os labels começam com o verbo no particípio e têm a forma = verbo + "from" | "By" [+Substantivo] são mapeados como propriedades;

Na etapa de geração das sentenças, a ferramenta assume que já foram executados os passos de análise das diretrizes e que as regras e diretrizes já foram confrontadas. Então após estes passos, a etapa de geração das sentenças é executada com base nas categorias gramaticais *Natural Theoretic Morphosyntax (NTMS)* e em regras *DCG (Definite Clause Grammar)* em Prolog. Onde NTMS é um modelo para geração de gramática que usa estruturas de categorização específicas para sentenças e que foi criado pelo grupo de pesquisa NIBA. Algumas das categorias NTMS usadas na gramática proposta são: v3(=sentencenode), n3(=nominalphrase), a0(=adjective), n0(=noum), vo(=verb), aux0(=auxiliary), <pass>(=passivation), <tvag2>(transitive,agentverb) e PP(=pastparticiple).

A Figura 6 ilustra como uma sentença é gerada através da gramática NIBA e a partir do fragmento de OWL apresentado abaixo:

```
<owl:Class rdf:about=''#WhiteLoire"">
        <rdfs:subClassOf>
        <owl:Restriction>
        <owl:onProperty rdf:resource=''#madeFromGrape"/>
        ...
```

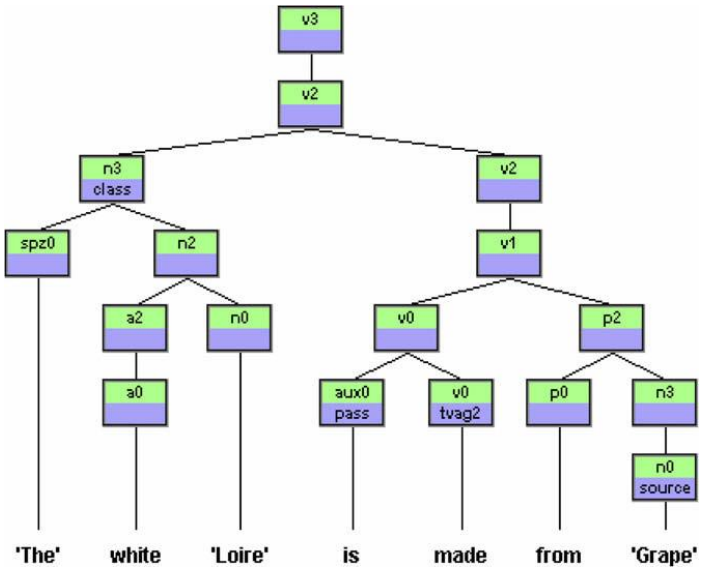


Figura 6– Exemplo de uma árvore da gramática NIBA (Fliedl et al., 2010)

Como esta abordagem espera que os nomes dos recursos sejam criados seguindo um conjunto de recomendações, as frases geradas possuem uma legibilidade melhor do que as obtidas na abordagem ACE.

2.5.2.3. Swoop

SWOOP é um projeto da Universidade de Maryland, EUA, que inclui o desenvolvimento de uma ferramenta para criação, edição e depuração de ontologias definidas na linguagem OWL. Como parte deste projeto foi apresentado em Hewlett et al., 2005 uma abordagem para a geração de LN a partir de arquivos OWL.

Esta abordagem é voltada para a língua inglesa e utiliza o processo de marcação de unidades de um texto, denominado *Part-Of-Speech Tagger*[18], para detectar automaticamente as categorias linguísticas dos termos utilizados na definição dos *labels* de uma ontologia em OWL. Além disso, a abordagem propõe um conjunto fixo de regras de expansão para a aplicação de *templates* linguísticos

[18] Part-Of-Speech (POS) - é a categoria gramatical de uma palavra, ou de itens lexais, que é geralmente definida pelo comportamento sintático ou morfológico do item lexical em questão. Categorias linguísticas comuns incluem substantivo e verbo, dentre outros.

Part-Of-Speech Tagger (POS Tagger) é uma parte de um software que lê o texto em um determinado idioma e que através de um processo de marcação atribui classes gramaticais a cada palavra ou unidade (*tokens*) presente no texto. As classes gramaticais são substantivo, verbo, adjetivo, etc. Geralmente os softwares mais refinados usam etiquetas POS mais específicas, como o conjunto de etiquetas POS para língua inglesa proposto no projeto Penn Treebank da Universidade Pensilvânia -EUA (ex.: NP = substantivo próprio no singular; VB = verbo na forma básica; RBS = advérbio, superlativo).

e, assim como a abordagem do projeto NIBA, também tira proveito da definição de *labels* padronizados.

Para ilustrar como a ferramenta pode executar o mapeamento de uma propriedade, suponha que uma ontologia possui uma *object property* cujo *rdfs:label* seja igual a "`hasColor`". Neste caso o algoritmo utilizado na ferramenta identifica que:

- *o label está na categoria:* `(has)+(NP)` *(onde NP=substantivo Próprio)*
- *e que os templates candidatos para o mapeamento são:*
  1. `X has a color Y`
  2. `X has Y color` *(se a regra de expansão "**se Y é um AdjP**" for satisfeita)*

Um aspecto interessante é que esta abordagem é capaz de concatenar todas as *datatype properties* e *object properties* de um determinado sujeito numa mesma sentença. Este fato diminui o número de sentenças para o mesmo sujeito, mas torna a sentença muito longa e, portanto, pouco inteligível. Para resolver esta situação, a abordagem oferece uma forma de apresentação da sentença num formato de uma lista enumerada aninhada no sujeito da sentença. No exemplo a seguir de uma sentença onde o sujeito é o vinho *Beaujolais* e este possui cinco propriedades:

"A Beaujolais is a Wine that:

-- is made from at most 1 grape, which is Gamay Grape

-- has Delicate flavor

-- has Dry sugar

-- has Red color

-- has Light body"

Deste modo, a sentença se tornou simples e sem a necessidade do uso de pronomes pessoais.

2.5.2.4. Anotações semânticas para imagens

Em Hollink et al. (2003) é apresentada uma abordagem diferente onde, sentenças estruturadas e apoiadas por vocabulários controlados podem ser geradas com o objetivo de descrever imagens. Esta abordagem foi desenvolvida como parte de um projeto de pesquisa intitulado *I'mIk: Interactive Disclosure of*

Multimedia, Information and Knowledge (2002-2006) da Universidade de Amsterdã, Holanda.

A abordagem apresenta uma ferramenta que auxilia a geração manual de descrições de imagens utilizando, sempre que possível, uma ou mais ontologias ligadas entre si. As descrições são compostas por conteúdo estruturado, ou seja, ao se descrever, por exemplo, um quadro de arte, o usuário emprega uma coleção de declarações do tipo *"agente+ação+objeto+destinatário"*. Cada declaração deve ter pelo menos um agente, uma ação e um objeto ou um destinatário. Os termos usados nas sentenças são selecionados a partir de termos de diferentes tesauros ou ontologias. Várias frases podem ser usadas para descrever um único quadro de arte. Por exemplo, uma pintura de Chagall, onde Chagall beija a esposa e recebe flores dela, poderia ser descrita com duas declarações:

Agente: "Chagall, Marc" (Ulan[19])
Ação: "kiss" (WordNet[20])
Destinatário: "wives" (AAT[21])

Agente: "woman" (WordNet)
Ação: "give" (WordNet)
Objeto: "flower" (WordNet)
Destinatário: "Chagall, Marc" (Ulan)

Segundo os autores, este esquema evita os problemas de análise sintática de descrições em LN, mantendo alguma naturalidade e riqueza nas descrições. O uso de tais conceitos para descrever a imagem permite fazer uma correspondência semântica durante a pesquisa. Por exemplo, pode-se encontrar esta imagem usando um sinônimo ou hiperônimo (por exemplo, "touch" em vez de "kiss"), uma vez que estes termos pertencem a uma ontologia criada para o WordNet.

Além disso, a ferramenta trabalha com o conceito de "*cenário*", ou seja, as características da cena como um todo. Para descrever o cenário, são usados os

[19] Union List of Artist Names (ULAN) é um thesaurus que contém informações sobre 220.000 artistas.

[20] WordNet é uma base de dados léxica na qual os substantivos, adjetivos, verbos e advérbios são organizados como conjuntos de sinônimos que representam um determinado conceito.

[21] Art and Architecture Thesaurus (AAT) é um grande thesaurus contendo aproximadamente 125.000 termos relevantes para o domínio da art. Os termos são organizados em uma hierarquia simples.

slots: *evento, local e hora*. Estes três *slots* também são preenchidos usando termos de tesauro. Por exemplo, a pintura de Chagall pode ser descrita com o evento localização igual a "local de trabalho do artista" (conceito da WordNet).

Para auxiliar o alinhamento dos termos que compõem as sentenças a ferramenta oferece a facilidade de busca por palavra chave em ontologias de domínio e de alto nível previamente carregadas na ferramenta. Também oferece um campo de texto livre, que pode ser usado quando a informação não se encaixa em um dos *slots* ou não está presente em uma das ontologias.

2.5.3. Comparação das Abordagens

Todas as abordagens de verbalização são focadas no aspecto de construção de uma boa LN a partir de modelos bem definidos em OWL. Isso depende da consciência do usuário ao nomear os termos utilizados para denominação das classes e propriedades da ontologia.

O maior problema enfrentado pelas abordagens de mapeamento OWL-to-LN reside em encontrar um meio-termo entre a acurácia e a legibilidade das sentenças. A abordagem ACE, por funcionar também como uma linguagem de primeira ordem, não possui ambiguidades em suas sentenças e, por isso, é pouco maleável no que se refere à legibilidade humana. Soluções mais flexíveis são apresentadas em Swoop, que garante que o sujeito esteja sempre presente, mesmo agregando todas as propriedades em uma única sentença, o que permite uma melhor legibilidade para humanos.

Outro problema importante é que as abordagens são voltadas apenas para a língua inglesa, ou seja, usam particularidades desta língua para a formação de suas frases.

A tabela 3 apresenta um resumo para a comparação das abordagens de mapeamento OWL-to-LN apresentadas.

Tabela 3 – Comparação das abordagens OWL-to-LN

	ACE	NIBA	SWOOP	Anotações semânticas para imagens
Legibilidade humana	média	alta	alta	média
Legibilidade máquina	alta	Nenhum comprometimento	Nenhum comprometimento	alta
Abordagem usada	Conjunto de regras próprias em DCG	Recomendações para padrão de *labels;* filtragem de padrões linguísticos; conjunto de regras próprias em DCG	Part-Of-Speech Tagger e conjunto de regras próprias	Apoio à geração de descrições estruturadas para imagens com reuso e alinhamento de vocabulário
Usa *rdfs:label*	não	sim	sim	sim
Geração da sentença	automática	automática	automática	manual
Idioma	Inglês	Inglês	Inglês	Inglês
Inovação	Funciona também como Linguagem de Primeira Ordem	Recomendações para padronização de labels	Concatena *datatype properties* e *object properties* de um determinado sujeito na mesma frase	Geração de descrições estruturadas a partir de várias ontologias

2.6. Resumo

Neste capítulo, foi mostrado como trabalha um motor de coleta de um mecanismo de busca tradicional e que suas limitações geram um ambiente conhecido como *Deep Web*. Foram apresentadas as soluções mais conhecidas para a localização de dados da Deep Web e discutiu-se as vantagens e desvantagens de cada uma delas.

Também apresentou-se um resumo sobre os conceitos que envolvem a publicação de fontes de dados ligados na Web. Foi discutido como os bancos de dados podem ser mapeados para RDF e foram apresentados os trabalhos relacionados ao mapeamento de RDB-to-RDF, seguido de uma análise comparativa destes trabalhos. Foi descrita a importância do reuso de vocabulários e da ligação entre os dados de diferentes vocabulários a fim de aumentar a

localização dos dados na Web de dados. Diferentes maneiras de se publicar dados estruturados na Web de dados foram discutidas e, dentre elas, abordou-se como usar o RDFa.

Um resumo sobre geração de linguagem natural foi incluído, seguido de uma pequena descrição sobre geração de LN a partir de RDF. Finalmente foram apresentados os trabalhos relacionados que envolvem mapeamentos OWL-to-LN e uma discussão sobre estes trabalhos.

3
Abordagem W-Ray

Este capítulo descreve a abordagem W-Ray, proposta nesta tese, seus objetivos e suas vantagens. O capítulo está organizado da seguinte maneira: inicialmente é apresentada uma breve introdução detalhando os pontos relevantes que inspiraram a criação da abordagem W-Ray. A seguir, a abordagem é detalhada através da descrição de suas etapas: *Projeto das views; Projeto da Ontologia* e *Publicação.* Finalmente a abordagem W-Ray é comparada com outras abordagens existentes.

3.1. Motivação

W-Ray é uma abordagem para o problema de localização de dados no contexto da *Deep Web,* que propõe um enfoque diferente dos encontrados na literatura, retirando a responsabilidade dos mecanismos de busca, movendo-a para os bancos de dados.

A ideia da abordagem W-Ray é resultado de um conjunto de observações feitas na literatura. Primeiro, observou-se que apesar dos recentes progressos, o acesso aos dados da *Deep Web* continua sendo um desafio porque, conforme discutido no capítulo 2, as abordagens existentes ainda não resolvem o problema de maneira efetiva. Em seguida, foi constatado que, dentre as abordagens existentes, a abordagem *surfacing* destaca-se por aproveitar o avanço da tecnologia dos mecanismos de busca tradicionais mantendo, desta forma, o ambiente que o usuário está acostumado a trabalhar na Web. Contudo, esta abordagem apresenta três deficiências importantes que ainda não foram resolvidas. São elas:

1. a falta de privacidade e segurança por parte dos donos dos dados, uma vez que não possuem a liberdade para escolher quais dos seus dados podem ser visualizados pelos mecanismos de busca;

2. a perda da semântica dos dados estruturados quando são trazidos para a Web convencional;
3. a falta de abrangência em relação a diferentes tipos de dados, como os dados multimídia, geográfico e estatísticos.

Diante desse cenário, surge a seguinte pergunta: *Como resolver as deficiências existentes na abordagem surfacing mantendo suas vantagens?*

Buscando respostas para tal pergunta, foram observados os seguintes pontos:

- Atualmente a forma mais indicada de publicar dados estruturados na Web é através de dados ligados. No entanto, porque estão em RDF, tais dados não são rastreados pelos motores de coleta tradicionais.
- Para publicar dados na Web de dados, são utilizadas ferramentas de mapeamento RDB-to-RDF. No entanto, tais ferramentas requerem um trabalho adicional do administrador de dados para as tarefas de seleção dos dados que serão publicados, além do alinhamento de dados e de vocabulários, conforme discutido no capítulo 2. Mesmo assim, nos últimos anos, cada vez mais dados estruturados têm sido disponibilizados na Web de dados através de ferramentas deste tipo.
- Ainda no contexto da Web de dados, o RDFa tem sido muito utilizado para estruturar textos contidos em documentos HTML.
- Em paralelo, pode-se constatar que muitos donos de dados desejam aumentar a visibilidade dos seus dados, uma vez que submetem seus dados estruturados em sites como o Google Shopping, mesmo sabendo que esse procedimento envolve uma grande quantidade de trabalho adicional e que o Google Shopping não promete que seus dados serão contemplados nos resultados de busca do Google Web.

Analisando todos estes pontos, constatou-se que uma forma interessante de resolver a primeira deficiência da abordagem *surfacing* seria retirar dos mecanismos de busca a responsabilidade de fornecer a visibilidade aos dados da *Deep Web* e transferi-la para os bancos de dados. Desta forma, a abordagem W-Ray passa a incluir a figura do *administrador de dados* ou *projetista W-Ray*, que

tem a responsabilidade de decidir quais dados podem ser expostos e como eles devem ser publicados na Web.

A segunda deficiência da abordagem *surfacing* - *a perda da semântica dos dados estruturados* – é solucionada através da publicação de uma parte significativa dos dados armazenados em um banco de dados, utilizando sentenças em linguagem natural, organizadas em páginas HTML estáticas com RDFa embutido. Desta forma, é possível atrair os mecanismos de busca e, ao mesmo tempo, manter a semântica dos dados.

Quanto à terceira deficiência - *falta de abrangência com relação a diferentes tipos de dado*s – a abordagem W-Ray é capaz de dar visibilidade aos dados convencionais, dados estatísticos provenientes de um *data warehouse* e dados geográficos em formato vetorial ou *raster*. A solução define, para cada tipo de dado, um conjunto de diretrizes para a criação de *views* materializadas. A partir das *views*, descrições automáticas podem ser geradas para tornar os diferentes tipos de dados visíveis aos motores de coleta. O problema dos dados multimídia, mais especificamente vídeos, é tratado em Nunes et al. (2012). Sentenças são geradas a partir de um serviço de reconhecimento de voz, publicadas em páginas HTML estáticas com RDFa embutido e ligadas a dados publicados na DBPedia.

A abordagem W-Ray é detalhada a seguir.

3.2. Etapas da abordagem W-Ray

A abordagem W-Ray consiste na execução de uma metodologia para publicar dados de bancos de dados relacionais em páginas Web. Esta metodologia é executada em três etapas:

1. O projetista W-Ray define manualmente um conjunto de *views* materializadas sobre o banco de dados que resumem os dados que devem ser publicados. O conjunto das views determina os *templates* que indicam como as sentenças devem ser geradas. Esta etapa é denominada **Projeto das Views**.
2. A segunda etapa, denominada **Projeto da Ontologia**, consiste no mapeamento RDB-to-RDF do conjunto de *views* (denominado na abordagem W-Ray como um "esquema das *views*") para um esquema RDF (denominado na abordagem W-Ray como "*ontologia da aplicação*"). Nesta etapa o projetista W-Ray deve seguir as diretrizes de dados ligados.

3. A última etapa, denominada **Publicação do Site**, depende da escolha dentre duas alternativas não excludentes:
   3.1. A primeira alternativa consiste na triplificação dos dados das *views*, que é feita com base no mapeamento do esquema das *views* para a *ontologia da aplicação*. Esta etapa é denominada **Triplificação**.
   3.2. A segunda alternativa consite na publicação dos dados na Web. Esta se utiliza dos *templates* para gerar sentenças bem estruturadas em LN, que descrevem os dados das *views* e, em seguida, publica-as em páginas da Web com RDFa embutido. Esta etapa é denominada **Publicação do Web Site** e é composta por três sub-etapas:
      3.2.1. A primeira, denominada **Projeto dos Templates**, consiste na seleção automática dos *templates* que orientam a geração das sentenças.
      3.2.2. A segunda, denominada **Projeto do Web Site**, consiste em como organizar as sentenças em páginas HTML estáticas.
      3.2.3. A terceira, denominada **Geração do Web Site**, consiste na geração de páginas HTML com ou sem RDFa embutido.

A etapa de **Publicação do Web Site** e a de **Triplificação** são diferentes uma da outra, mas não são mutualmente exclusivas, uma vez que o projetista W-Ray pode optar por publicar um site que descreve as *views*, ou triplificar as *views*, ou ambos.

A Figura 7 mostra mais detalhadamente as etapas da abordagem W-Ray. Na coluna da esquerda se encontram as etapas do projeto e na da direita os resultados de cada etapa.

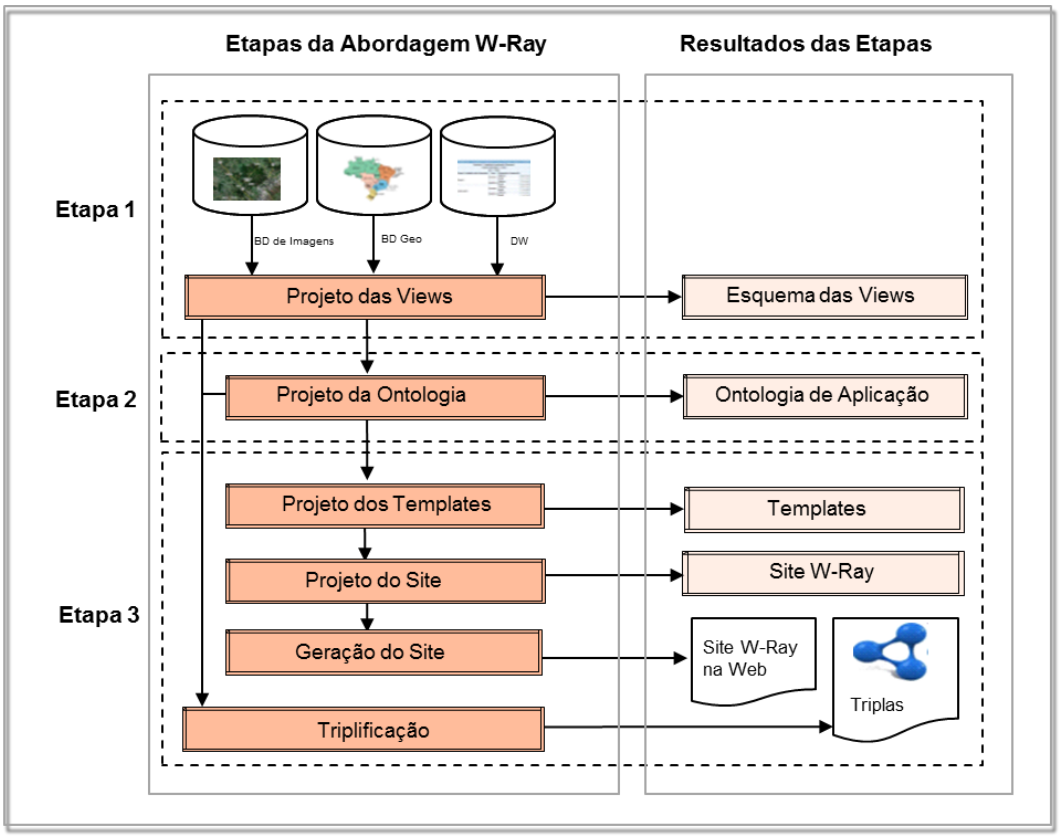


Figura 7 - Etapas da abordagem W-Ray e resultados de cada etapa

Conforme descrito no capítulo 2, na visão de geração de LN, as etapas da abordagem W-Ray podem ser interpretadas da seguinte maneira:

- Determinação de conteúdo (“o que dizer?”): Corresponde à etapa de **Projeto das Views**.
- Organização do Discurso (“onde dizer?”): Corresponde à etapa de **Projeto da Ontologia** e sub-etapa de **Projeto dos Templates**.
- Produção do texto (“como dizer?”): Corresponde às sub-etapas de **Projeto dos Web Sites** e **Geração dos Web Sites**.

Por outro lado, pode-se considerar a abordagem W-Ray sob a perspectiva de publicação de dados ligados na Web, conforme descrito no capítulo 2. Neste sentido, as etapas da abordagem W-Ray podem ser interpretadas da seguinte forma:

- Mapeamento do esquema de banco de dados relacional para uma ontologia: o W-Ray divide este mapeamento em duas etapas: Etapa de Projeto das Views e Etapa de Projeto da Ontologia.

- O mapeamento do conteúdo do banco de dados relacional para triplas RDF corresponde à fase Triplificação e à de Publicação dos Web Sites com RDFa.

As próximas seções descrevem cada etapa do projeto da abordagem W-Ray detalhadamente.

3.2.1. Projeto das Views

O projetista W-Ray deve primeiro selecionar quais dados serão publicados com a ajuda de *views* materializadas de banco de dados relacional. Para maior clareza as diretrizes genéricas para o projeto de *views* que são aplicadas a qualquer tipo de dado são tratadas de forma separada das diretrizes que se aplicam especificamente aos dados geográficos e estatísticos.

Dados Convencionais

O projeto das *views* deve obedecer às seguintes diretrizes genéricas no momento da definição das *views*:

– Assumindo que existem estatísticas sobre o número de acesso aos dados, as *views* devem refletir dados que são frequentemente solicitados.
– *Views* não devem violar as regras de privacidade estipuladas pelos donos dos dados.
– Atributos cujos valores não têm semântica fora do banco de dados não devem ser diretamente publicados, tais como tabelas criadas para fins administrativos do próprio banco de dados.
– Chaves primárias geradas artificialmente, chaves estrangeiras que fazem referência a essas chaves primárias, atributos com domínios que codificam as classificações ou artefatos semelhantes, se selecionados para publicação, devem ter seus valores internos substituídos por suas respectivas definições externas. Por exemplo, um código de classificação deve ser substituído pelo termo da classificação correspondente.
– As *views* não devem conter muitos atributos. Apenas devem ser selecionados os atributos e seus relacionamentos que são relevantes ou que ajudam na localização dos objetos.

Em outras palavras, as *views* devem exteriorizar dados que fornecem um resumo dos objetos mais importantes, dentro dos limites da privacidade.

Dados Geográficos:

Ao definir *views* para os dados geográficos, além das diretrizes genéricas, o projetista deve selecionar os dados que externalizem:

- valores dos atributos convencionais como, por exemplo, os nomes oficiais de objetos geográficos;
- metadados de conjuntos de dados geográficos, como escala geográfica e projeção cartográfica;
- relacionamentos topológicos[22] entre os objetos geográficos descritos.

Dados geográficos podem ser representados em formato vetorial[23] e em formato raster[24]. Cada um destes formatos implica em diretrizes específicas para a definição das *view.*

Dados Geográficos em formato Vetorial:

Considere que os dados geográficos em formato vetorial estão organizados em camadas, tais como, divisão política, hidrografia e relevo, o projetista W-Ray deve obedecer às seguintes diretrizes adicionais ao definir as *views*:

- A definição da *view* deve combinar um pequeno número de camadas que contenham objetos geográficos interligados;
- Para cada camada, a definição da *view* deve incluir uma restrição que filtre objetos geográficos sem importância;
- Para cada camada, a definição da *view* deve selecionar alguns atributos relevantes dos objetos geográficos;
- Quando a *view* combina várias camadas, sua definição deve:
  - especificar a prioridade entre as camadas;
  - especificar quais relacionamentos topológicos entre os objetos geográficos de diferentes camadas devem ser materializados;
  - indicar qual a ordenação topológica dos objetos que serão descritos. Por exemplo, lugarejos ou povoados podem ser descritos de norte para o sul e

[22] Conceito disponível no Apêndice A.
[23] Conceito disponível no Apêndice A.
[24] Conceito disponível no Apêndice A.

de oeste para leste. Assim, na *view*, os dados devem estar organizados nesta ordem.

Como exemplo, considerando a *Base Cartográfica Vetorial Contínua do Brasil ao Milionésimo* (BCIM), que é um produto disponível do site do Instituto Brasileiro de Geografia e Estatística (IBGE), observa-se que ela é composta por várias camadas geográficas. Supondo que as *views* construídas sobre a BCIM utilize apenas as camadas Divisão Política, Localidades e Hidrovias do Brasil, pode-se criar os seguintes filtros:

- Divisão Política: apenas os Estados localizados na região norte devem ser selecionados, com seu nome, nome abreviado, área e população;
- Localidades: apenas as capitais estaduais da região norte devem ser selecionadas, com o nome, sigla, área e população;
- Hidrovias: apenas hidrovias da região norte devem ser selecionadas, com seus nomes e tipo de navegabilidade;

Além disso, considera-se que a relação topológica entre Capitais e Estados seja *"está localizado em"*, que entre Hidrovias e Estados seja "*cruza*", que as cidades têm prioridade sobre os estados e que cidades devem estar ordenadas de norte para o sul e de leste para o oeste. A Figura 8 mostra as *views* que refletem estes requisitos. A Figura 9 mostra o resultado do esquema das *views*, incluindo os relacionamentos topológicos.

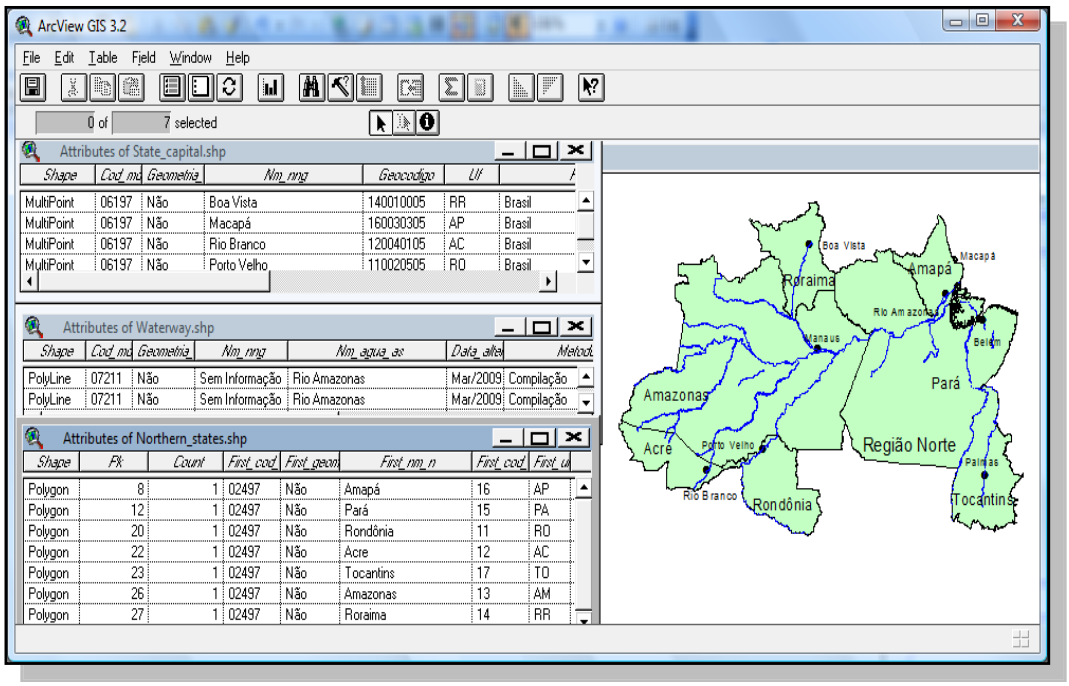


Figura 8 - Resultado das *views* sobre as camadas da BCIM – fonte: IBGE.

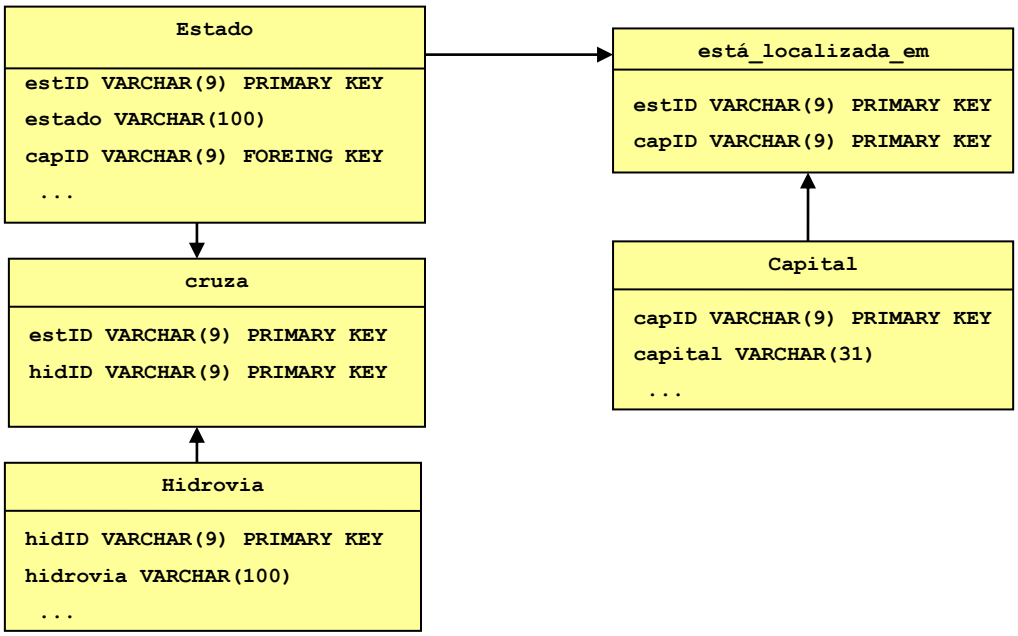


Figura 9 - Exemplo de esquema da *views* para dados geográficos em formato vetorial

A seguir, são mostrados exemplos de sentenças que podem ser geradas a partir das *views* da Figura 9 para descrever o mapa da Figura 8.

> **Amazonas** é um *rio* que *cruza o estado* **Pará.**
> **Pará** é um *estado* que possui sigla **PA**, população igual a **7.581.051**, área em km **1.247.954,666** e *capital* igual a **Belém.**

Dados Geográficos em formato Raster:

Leme et al. (2007) indica que, para descrever dados em formato raster, o projetista primeiro deve selecionar bancos de dados geográficos, como os *gazetteers*, que cobrem a mesma área geográfica dos dados, em formato raster, que serão descritos. Então, seguindo as diretrizes de dados vetoriais, ele deve definir *views* que selecionem os objetos geográficos contidos dentro de um *bounding box* desenhado sobre os dados raster.

Por exemplo, supondo que o *gazetteer* GeoNames (GeoNames, 2012) seja adotado, a definição da *view* deve recuperar todos os objetos geográficos do *gazetteer* GeoNames, cujos centróides estejam dentro de uma determinada caixa delimitadora, definida sobre uma imagem de satélite. A definição da *view* também

poderia restringir os tipos dos objetos selecionados, através das classificações geográficas dos objetos definidas pelo *GeoNames Feature Codes*.

Para ilustrar a abordagem W-Ray para dados geográficos no formato raster, foram selecionados:

– Um fragmento da imagem da cidade do Rio de Janeiro a partir do Web site do "INPE - Instituto Nacional de Pesquisas Espaciais" (Figura 10), cujas coordenadas acima e abaixo da figura se referem à caixa delimitadora.
– O *gazetteer* GeoNames, para fornecer as características geográficas e suas classificações.

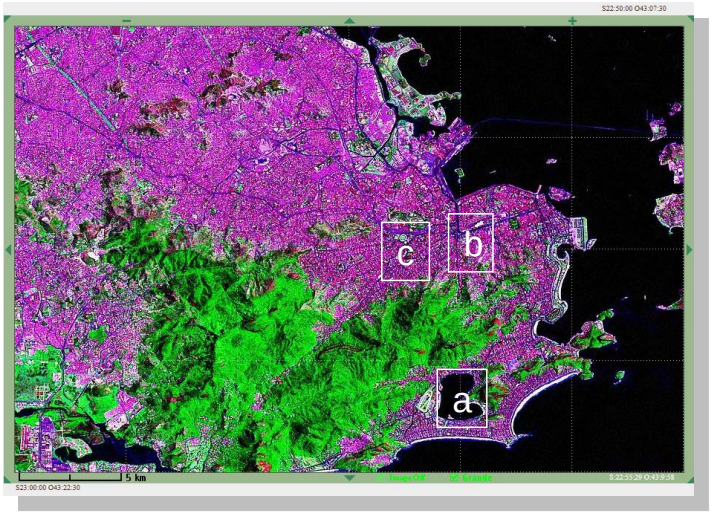


Figura 10 - Fragmento de uma imagem de satélite do Rio de Janeiro. Fonte: INPE

Pode-se definir uma *view* sobre o *gazetteer Geonames*, selecionando características geográficas, cujos centróides estejam contidos na caixa delimitadora da imagem, e que estejam classificadas pelo *GeoNames Feature Codes* como *"stream, lake,..."*. Os passos para a definição das *views* seriam :

1. Os parâmetros georeferenciados são extraídos a partir da imagem. Neste caso, o fragmento da imagem possui caixa delimitadora definida por ((22° 50" 00"S, 43° 07" 30"O), (23° 00" 00"S, 43° 22" 30"O)).
2. Ao consultar o dicionário geográfico *Geonames* com os parâmetros de georreferenciamento extraídos no Passo 1 e o *GeoNames Feature Codes* com a classificação *stream*, *lake*, ..., muitos objetos são localizados. Três exemplos foram selecionados (Figura 10):
    a. Recurso ("Rodrigo de Freitas", *lake*)
    b. Recurso ("Comprido", *stream*)

c. Recurso ("Maracanã", *stream*)

3. Os resultados, quando convertidos para sentenças, podem descrever a imagem, conforme sentença a seguir.

> **L7ETM21707620030220** is a(n) *satellite image* that has **Rodrigo de Freitas** *lake*, **Comprido** *stream* and **Maracanã** *stream* .

Exemplos concretos de *views* sobre dados geográficos serão apresentados no capítulo 5.

Dados Estatísticos:

A definição de *views* sobre um *data warehouse* que contém cubos de dados[25] requer uma estratégia diferente para evitar a geração de um número muito grande de sentenças. W-Ray define as seguintes diretrizes para cubos de dados:

- Definir *views* que capturem sub-cubos dos cubos armazenados no *data warehouse* de forma a reduzir o número total de sentenças;
- Definir *views* que correspondam às dimensões dos cubos de dados, ou para as combinações das dimensões mais importantes;
- Se o número de sentenças ainda continuar muito alto, considerar a definição de *views* que capturam apenas os metadados que descrevem os cubos armazenados no *data warehouse*.

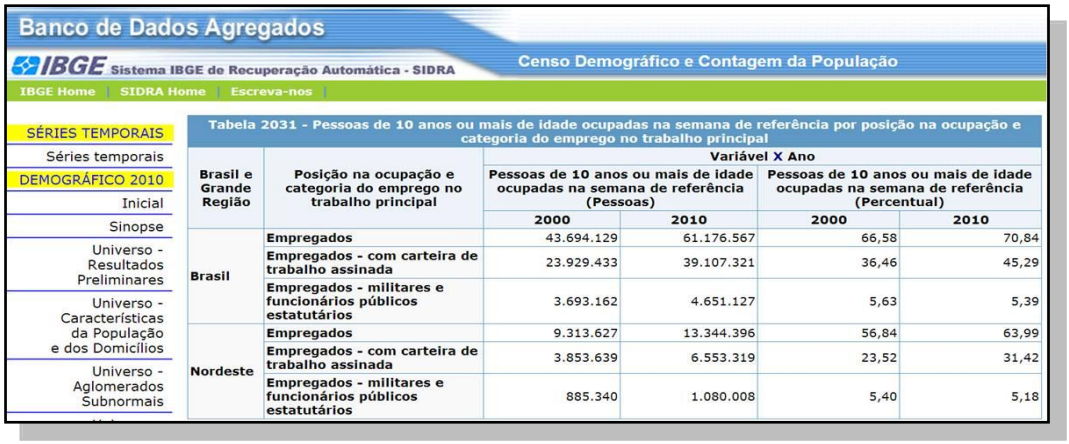
A estratégia adotada para definir *views* afeta diretamente a triplificação dos cubos de dados. Por exemplo, considere que a *view* é definida como uma tabela plana, onde que cada dimensão é mapeada para uma coluna diferente e as variáveis estatísticas[26] correspondem aos dados estatísticos de uma coluna separada. Neste caso, o processo de triplificação irá gerar um indivíduo RDF para cada linha da *view* materializada. Por outro lado, se as *views* representam uma tabela cruzada[27], o processo de triplificação irá gerar um indivíduo RDF para cada célula. Neste último caso, a *view* que possui os valores da variável estatística terá uma relação n-ária com as *views* que representam o domínio de cada dimensão. A

[25] Ver definição no Apêndice A

[26] Variável estatística - é uma característica qualquer de interesse que é associada à população ou à amostra para ser estudada estatisticamente. Ex.: Sexo, idade, ocupação.

[27] Uma tabela cruzada de duas dimensões possui variáveis como cabeçalhos de linhas e de colunas e suas células representam um dado agregado.

Figura 11 contém um exemplo de dados estatísticos, provenientes do Banco de dados Agregados do IBGE, para os quais se pretende criar *views*. A primeira alternativa seria modelar a *view* como uma tabela plana como na Figura 12. A segunda alternativa seria modelar como uma tabela cruzada como na Figura 13.



Banco de Dados Agregados

IBGE Sistema IBGE de Recuperação Automática - SIDRA — Censo Demográfico e Contagem da População

IBGE Home | SIDRA Home | Escreva-nos

SÉRIES TEMPORAIS
Séries temporais
DEMOGRÁFICO 2010
Inicial
Sinopse
Universo - Resultados Preliminares
Universo - Características da População e dos Domicílios
Universo - Aglomerados Subnormais

Tabela 2031 - Pessoas de 10 anos ou mais de idade ocupadas na semana de referência por posição na ocupação e categoria do emprego no trabalho principal

Brasil e Grande Região	Posição na ocupação e categoria do emprego no trabalho principal	Variável X Ano			
		Pessoas de 10 anos ou mais de idade ocupadas na semana de referência (Pessoas)		Pessoas de 10 anos ou mais de idade ocupadas na semana de referência (Percentual)	
		2000	2010	2000	2010
Brasil	Empregados	43.694.129	61.176.567	66,58	70,84
	Empregados - com carteira de trabalho assinada	23.929.433	39.107.321	36,46	45,29
	Empregados - militares e funcionários públicos estatutários	3.693.162	4.651.127	5,63	5,39
Nordeste	Empregados	9.313.627	13.344.396	56,84	63,99
	Empregados - com carteira de trabalho assinada	3.853.639	6.553.319	23,52	31,42
	Empregados - militares e funcionários públicos estatutários	885.340	1.080.008	5,40	5,18

Figura 11 - Tabela com dados agregados do Sistema SIDRA-IBGE

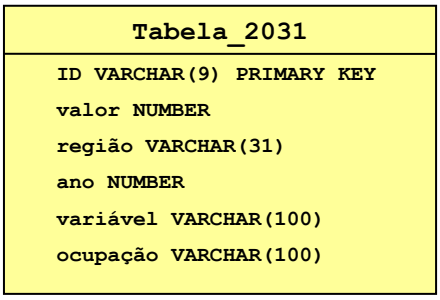


Figura 12 - Modelo de tabela plana gerado para os dados estatísticos da Figura 11.

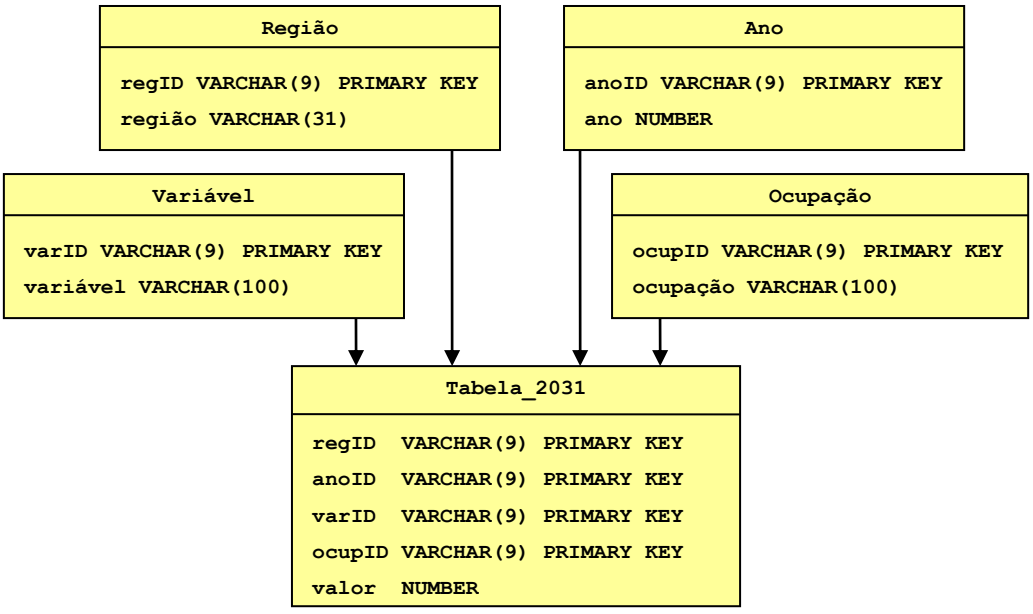


Figura 13 - Modelo de tabela cruzada gerado para os dados estatísticos da Figura 11.

No capítulo 5 a questão da publicação de cubos de dados será aprofundada utilizando um exemplo específico.

3.2.2. Projeto da Ontologia

Na abordagem W-Ray o esquema RDF é usado para triplificar os dados das *views*, gerar o RDFa embutido nas páginas da Web e orientar a construção dos *templates* que auxiliam a geração das sentenças em linguagem natural. Neste contexto, o mapeamento do RDB-to-RDF é particularmente importante por dois motivos. Primeiro, os termos usados em banco de dados são tipicamente inadequados para serem exteriorizados para os usuários e indexados por mecanismos de busca. Segundo, para publicar os dados também na Web de Dados, é necessário seguir as normas para publicação de dados ligados, que exigem o alinhamento entre vocabulários. Por isso, antes da geração do esquema RDF, proveniente do esquema das *views*, o projetista deve:

- Fazer o alinhamento com vocabulários já existentes. Para isso, ele deve selecionar os vocabulários com que irá trabalhar, tais como, ontologias, tesauros ou glossários, que cubram o domínio da aplicação em questão, e ontologias de alto nível, tais como, a versão do WordNet em OWL (WordNet, 2009) ou SUMO - Suggested Upper Merged Ontology (SUMO, 2009).
- Fazer o alinhamento dos atributos que possuem domínio enumerado.
- Mapear os tipos de dados do banco de dados relacional para os tipos de dados utilizados em XML.

Como a execução do alinhamento é uma tarefa pesada que requer envolvimento humano, é conveniente o desenvolvolvimento de uma ferramenta que auxilie o projetista W-Ray. Esta ferramenta é apresentada no capítulo 5.

3.2.3. Projeto dos Templates

Este trabalho utiliza uma técnica própria de mapeamento RDF-to-LN que foi inspirada nas técnicas apresentadas no capítulo 2. Apenas foi adotada a mesma solução para simplificação das sentenças utilizada em Swoop e já discutida no capítulo 2. A parte fundamental da abordagem W-Ray está na tradução de dados

da *view* materializada para sentenças em linguagem natural. Os *templates* irão orientar a síntese do texto a ser incluído nas páginas Web.

A geração dos *templates* é baseada no vocabulário e na *ontologia da aplicação* definidos/criados na etapa anterior. Conforme descrito no capítulo 2, são fontes de texto:

- nomes de classe - que podem ser interpretados como substantivos e que funcionam como sujeitos ou objetos na frase.
- nomes de *object properties* - que funcionam como verbos transitivos;
- nomes de *datatype properties* - que funcionam como adjetivos, advérbios ou substantivos.
- pode-se formar frases combinando os textos embutidos no esquema RDF com os indivíduos.

Avaliando o mapeamento RDF-to-LN, pode-se concluir que os tipos de *template* dependem dos tipos de propriedade RDF. Com base nesta observação, os *templates* W-Ray são classificados como:

- *simples*, quando a propriedade é uma *datatype property* com valor único;
- *multivalorados*, quando a propriedade é uma *object property* proveniente de um relacionamento 1-n;
- *relacionamento binário*, quando a propriedade é uma *object property* proveniente de uma relacionamento n-m binário, ou seja, entre duas tabelas;
- *relacionamento n-ário*, quando a propriedade é uma *object property* proveniente da reificação (Noy & Rector, 2006) de um relacionamento n-ário, ou seja, entre mais de duas tabelas;

Os exemplos a seguir, ilustram como os *templates* são gerados, usando a seguinte convenção:

- **negrito**: utilizado para a apresentação dos dados nas sentenças prontas;
- **<negrito entre ângulos>**: espaço reservado para exibição de dados em um *template;*
- *Itálico*: classes e propriedades do esquema RDF;
- textos planos: palavras adicionais inseridas automaticamente.

A Figura 14 descreve três *views* definidas sobre o Web site do IBGE (http://mapas.ibge.gov.br). As *views Rio* e *Estado* contém somente dados alfanuméricos e foram criadas, respectivamente, a partir das camadas dos mapas *Hidrografia* e *Divisão Política*. A *view cruza* é definida pela consulta espacial sobre as camadas dos mapas *Hidrografia* e *Divisão Política*, capturando um relacionamento topológico n-m.

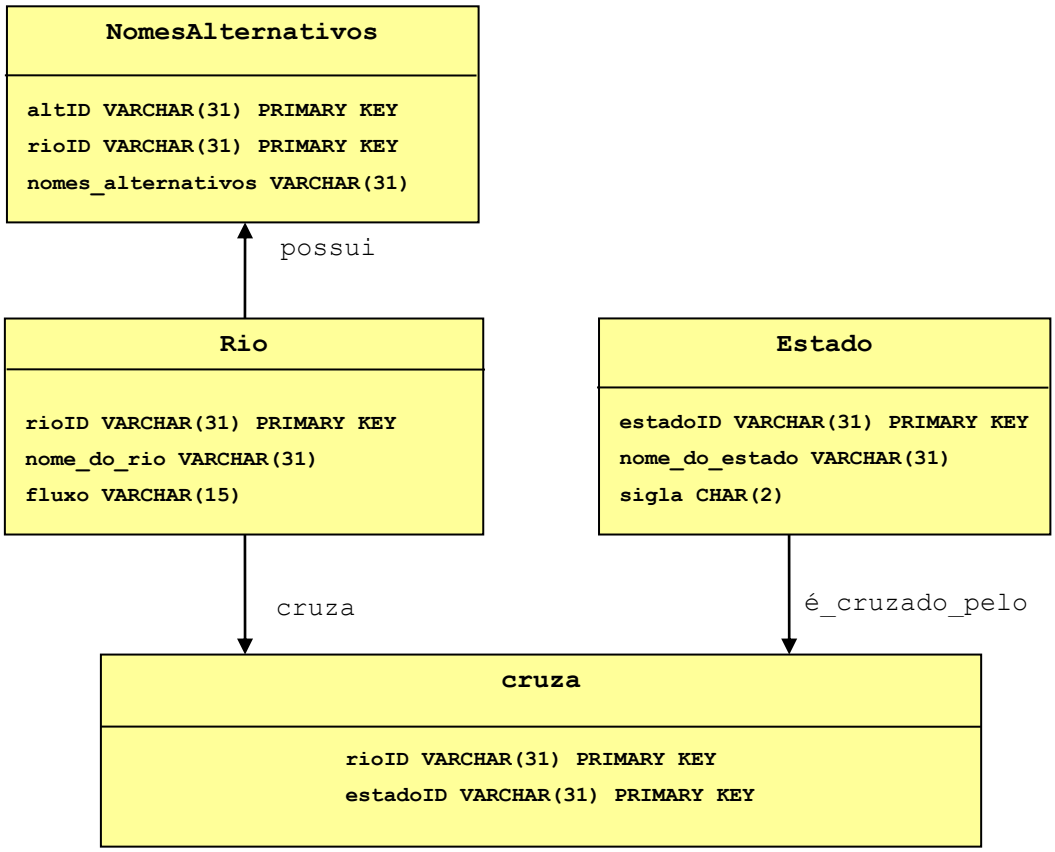


Figura 14 - Exemplo de três *views* sobre mapas do IBGE

Um mapeamento direto do esquema das *views* da Figura 14 gera um esquema RDF com as seguintes classes e propriedades:

classes: *Rio, Estado, NomesAlternativos*

datatype properties: *nome_do_rio,nome_do_estado,fluxo, sigla,nomes_alternativos*

object property: *cruza, possui*

object property inversa: *é_cruzado_pelo*

O exemplo abaixo se refere ao *template simples* associado à *datatype property* ***fluxo***:

<**nome_do_rio**> é um(a) *rio* que possui *fluxo* <**fluxo**>.

Um exemplo de sentença gerada a partir do *template* acima é:

Amazonas é um(a) *rio* que possui *fluxo* **permanente**.

Os *templates multivalorados* são mais elaborados, porque todas as triplas referentes à *object property* proveniente de um relacionamento 1-n devem ser representadas na mesma frase. O segundo exemplo mostra um *template multivalorado* associado à classe *NomesAlternativos*:

<nome_do_rio> é um(a) *rio* que possui *nomes alternativos* **<nomes_alternativos>**,…, e **<nomes_alternativos>**.

Um exemplo de sentença gerada a partir do *template* acima é:

Amazonas é um(a) *rio* que possui *nomes alternativos* **Carhuasanta**, **Lloqueta**, **Apurímac**, **Ene**, **Tambo**, **Ucayali** e **Solimões**.

O próximo exemplo ilustra o *template relacionamento binário* associado à *object property cruza*:

<nome_do_rio> é um(a) *rio* que *cruza* o(a) *estado* **<nome_do_estado>**.

Amazonas é um(a) *rio* que *cruza* o(a) *estado* **Pará**.

Como um exemplo final, é mostrado um *template* gerado a partir de uma *reificação*, ou seja, quando, no mapeamento de uma *view* para RDF, é necessário fazer a *reificação*. A Figura 15 ilustra esta questão.

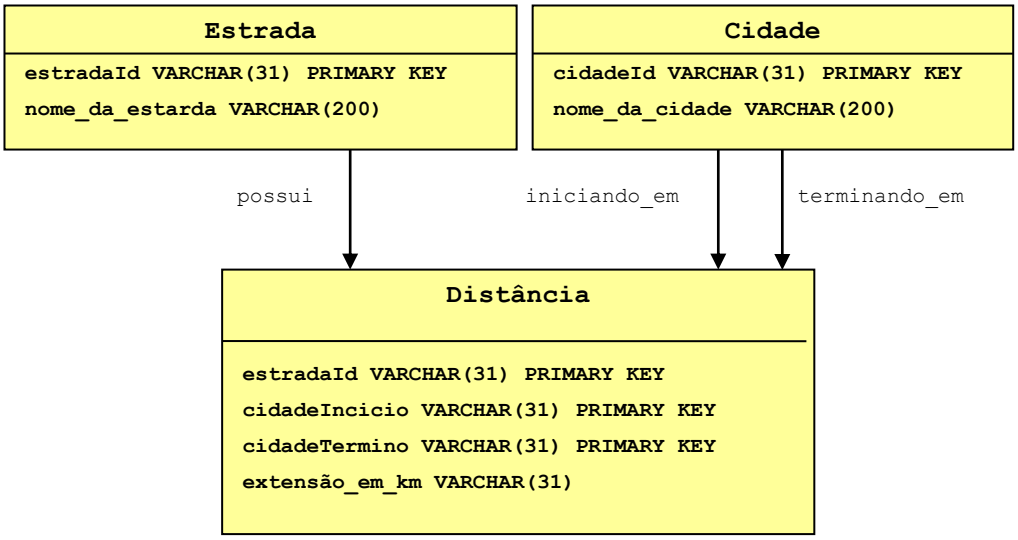


Figura 15 - Exemplo de um esquema das *views* que requer reificação para o mapeamento para RDF

Como a *view Distância* representa um relacionamento n-m com a *view Cidade*, e possui o atributo *extensão_em_km*, ela deve ser mapeada como uma classe (para representar os valores do atributo *extensão_em_km*). Os resultados da *reificação* são as seguintes classes e propriedades:

Classes: *Estrada, Cidade, Distância*

datatype proprerty: *extensão_em_km*

object properties: *iniciando_em, terminando_em, possui*

object properties inversas: *é_inicio_da, é_termino_da, referente_a*

Uma vez que a *view Distância* é representada por uma classe artificialmente introduzida pela *reificação*, é razoável criar um *template* único para a sentença que representará este cenário. Com base em informações sobre esta *reificação*, capturadas na etapa de projeto da ontologia de aplicação, o *template* a seguir é gerado automaticamente:

<**nome_da_estrada**> é uma(o) *estrada* que possui *extensão* <**extensão**> *inciando em* <**cidadeIncicio**>, e *terminando em* <**cidadeTermino**>.

Um exemplo para o *template relacionamento n-ário* acima seria:

BR-040 é uma(o) *estrada* que possui *extensão em km* **1178.7**, *iniciando em* **Brasilia** e *terminando em* **Rio de Janeiro**.

Por último, os *templates* da abordagem W-Ray espelham a ideia publicada em (Hewlett et al., 2005), ou seja, é capaz de concatenar todas as *datatype properties* e *object properties* na mesma sentença, para um mesmo sujeito. Esta ideia é particularmente útil quando a sentença é gerada a partir do *template relacionamento* (*n-ário* ou *binário*). Neste caso, se o número de indivíduos da classe, que funciona como o objeto da sentença, for elevado, isto pode implicar na geração de uma grande quantidade de sentenças todas com o mesmo sujeito. A Figura 16 mostra um exemplo desta simplificação.

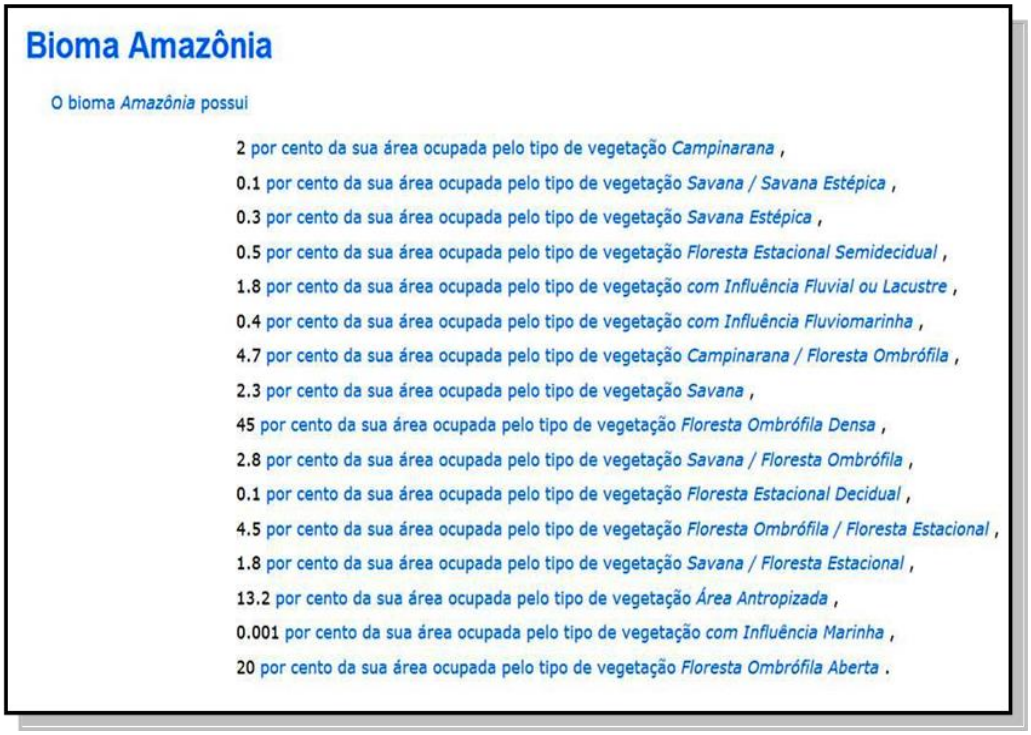


Figura 16 - Sentença simplificada - gerada a partir de uma reificação.

3.2.4. Projeto do Web site

As páginas Web são geradas pela abordagem W-Ray seguindo três diretrizes:

1. Diretrizes de acessibilidade do W3C (Caldwell et al., 2008);
2. Diretrizes divulgadas pela Google que visam otimizar a indexação de sites (Google-OptimizationGuide, 2012);
3. Diretrizes para publicação de dados ligados na Web (Bizer et al., 2009).

Decidiu-se utilizar, na abordagem W-Ray, as recomendações divulgadas pela Google, porque elas são válidas para a maioria dos motores de busca existentes e também porque a Google possui o mecanismo de busca mais utilizado atualmente.

Analisando as recomendações do W3C e as da Google, observa-se que algumas delas são bem similares, apesar dos seus objetivos serem distintos. O terceiro princípio de acessibilidade recomendado pelo W3C diz: *"A informação e a operação da interface de usuário devem ser compreensíveis."* Isto significa que os usuários (independente de suas limitações) devem ser capazes de compreender a informação, assim como a operação da interface com o usuário. O item 3 das diretrizes Google afirma: *"o conteúdo de uma página Web deve ser: fácil de ler; organizado em torno do tema; com uma linguagem relevante; atualizado e, original; e principalmente, ser criado para os usuários, e não para os motores de busca"*. As duas recomendações reforçam que a informação deve ser direcionada a humanos. Mas o objetivo da diretriz da W3C é a acessibilidade tanto para máquinas quanto para humanos e o da Google reflete uma preocupação com a manipulação do *pageRank*. De fato, os rastreadores Web da Google podem interpretar as palavras distribuídas aleatoriamente ou repetidamente em uma página da Web, como uma tentativa de manipulação do *pageRank*, e, assim, penalizar a indexação da página.

Ainda comparando as recomendações, pode-se observar que algumas do W3C, que são específicas para usuários com deficiência visual, de fato coincidem com as orientações da Google. Analisando-se este fato, pode-se concluir que as dificuldades encontradas pelo usuário com deficiência visual são semelhantes às encontradas por um motor de busca, durante a etapa de coleta de dados. A Google oferece recomendações para descrever o conteúdo de imagens através do atributo "alt" (Google-OptimizationGuide, 2012), enquanto que o princípio 1 do W3C recomenda que sejam feitas descrições para tornar as imagens acessíveis a todos os humanos (Caldwell et al., 2008). O conteúdo de uma imagem é invisível, tanto para um deficiente visual como para um motor de busca, mas um texto alternativo, que descreve a imagem, pode ser indexado por um motor de busca e sua leitura pode ser feita por um leitor de tela, para o usuário com deficiência visual.

Na verdade, muitas das estratégias adotadas, pela abordagem W-Ray, para resolver as limitações de motores de busca também se aplicam para a concepção

de uma interface de acesso a banco de dados, voltada para usuários com deficiências visuais.

As diretrizes do W3C e da Google, adotadas pela abordagem W-Ray com o objetivo de publicar páginas Web são:

- Criar hiperlinks entre os dados publicados, para melhorar a exploração de dados através de navegação (recomendação W3C 2.4 e Google 2).
- Criar conteúdo com frases bem estruturadas (recomendação W3C 3 e Google 3.1).
- Usar textos para descrever as imagens, quando o atributo "alt" não for suficiente (recomendação W3C 1.1 e Google 3.3).
- Páginas devem conter títulos e subtítulos bem definidos (recomendação Google 1).

Além dessas recomendações, a abordagem W-Ray adota as seguintes diretrizes, ao criar suas páginas Web:

- A estrutura do site, que irá conter os dados da *view* materializada, deve ser decidida antecipadamente e deve refletir o conjunto de *views*.
- Uma página HTML pode descrever uma ou mais *views* do banco de dados a ser publicado.
- As páginas HTML devem conter hiperlinks, partindo dos termos presentes nas sentenças, que são provenientes dos nomes de classes e propriedades da ontologia, para os conceitos correspondentes.
- As sentenças, geradas a partir da mesma *view* materializada, podem ser agrupadas em uma ou mais páginas da Web. Estas páginas devem estar ligadas a uma página inicial (*home page do site*) através de um hiperlink, e entre si.
- As sentenças geradas a partir de uma *reificação* são agrupadas em uma página, que aponta para outras páginas, que, por sua vez, contêm as descrições das classes envolvidas nos respectivos relacionamentos.
- As sentenças podem ser organizadas através de uma hierarquia para serem agrupas em títulos e subtítulos na página;
- As sentenças devem ser ligadas diretamente ao resultado da consulta do formulário, ou seja, diretamente ao dado. Se por motivos de segurança ou porque o banco de dados é muito volumoso e apenas um resumo dos dados é publicado através da abordagem W-Ray, pode não existir então a

correspondência direta entre o sujeito das sentenças e o dado. Neste caso, recomenda-se que a ligação seja feita com os formulários HTML de consulta ao banco de dados relacional, da Deep Web, que está sendo descrito no site W-Ray. É desta forma que a abordagem W-Ray faz a conexão com os dados armazenados nos bancos de dados da Deep Web criando um caminho para que os *search engines* possam localizar estas informações.

As páginas HTML geradas pela abordagem W-Ray podem ser publicadas com RDFa embutido, se esta for a escolha do usuário. Neste caso, seguimos as recomendações para publicação de dados ligados na Web, conforme descritas no capítulo 2. O RDFa é gerado conforme Adida et al., (2012) e os atributos são embutidos conforme resumo apresentado no capítulo 2.

3.2.5. Geração do Web Site

A etapa de Geração do Web Site oferece duas alternativas:

(1) Geração de páginas HTML com RDFa embutido. A geração das sentenças é executada com base nos *templates* e o RDFa é embutido no HTML, com base na ontologia de aplicação. As páginas HTML com RDFa são geradas em conformidade com a etapa de projeto do Web site e publicadas na Web convencional e na Web de dados.

(2) Geração de páginas HTML sem RDFa embutido. Com exceção da geração do RDFa, as páginas HTML são geradas da mesma forma que no item anterior e publicadas apenas na Web convencional.

Esta etapa é pesada e não necessita de decisões do usuário, o que possibilita a sua automatização. A ferramenta que implementa esta etapa é apresentada no capítulo 5.

3.2.6. Triplificação

A etapa de triplificação gera triplas RDF para os dados das *views* materializadas com base na *ontologia da aplicação* resultante do mapeamento RDB-to-RDF. Da mesma forma que a etapa anterior, esta também pode ser automatizada.

3.3. Comparação com trabalhos relacionados

Ao comparar a abordagem W-Ray com as abordagens existentes para a localização de informações na *Deep Web*, se conclui que a abordagem W-Ray pode ser entendida como uma alternativa para a abordagem *Surfacing*, que retira a responsabilidade dos mecanismos de busca, no que se refere à sondagem automática da *Deep Web*, e levando-a para os bancos de dados. Esta mudança requer a interferência da figura do administrador de banco de dados (que chamamos aqui de projetista W-Ray), assim como a autorização dos donos dos dados.

Este envolvimento do administrador do dado tem um aspecto positivo porque garente que só serão publicados os dados autorizados por seus donos, mas cria um custo para a sua execução. Embora este custo possa ser visto como um aspecto negativo, o controle sobre a publicação com envolvimento humano já aparece em outras abordagens. Esse é o caso da abordagem *Busca por Produto* ou mesmo das ferramentas de mapeamento RDB-to-RDF, discutidas no capítulo 2.

Outras diferenças em relação à *Surfacing* são a capacidade da abordagem W-Ray de manter a semântica dos dados estruturados e de fornecer visibilidade a diferentes tipos de dados.

Ao se comparar a abordagem W-Ray com as de mapeamento RDB-to-RDF, é possível observar que, assim como no Triplify, a ideia de trabalhar com *views* acrescenta as seguintes vantagens:

- SQL é uma linguagem amplamente conhecida;
- SQL é uma linguagem desenvolvida para estruturas relacionais e, portanto, oferece recursos não disponíveis nas outras abordagens de mapeamento RDB-TO-RDF, como a agregação e funções de agrupamento.
- *Views* permitem que o usuário publique apenas o conjunto de dados que achar conveniente.

No entanto, a abordagem W-Ray se difere da Triplify na solução oferecida para o alinhamento entre vocabulários. Em Triplify o alinhamento é feito através da introdução de “dicas” na linguagem SQL e na abordagem W-Ray pode ser implementada uma ferramenta de auxílio para esta tarefa.

A abordagem W-Ray não utiliza nenhuma das técnicas de mapeamento RDB-TO-LN descritas no capítulo 2. Apenas foi adotada a mesma solução para simplificação das sentenças utilizada em Swoop e já discutida no capítulo 2. Um dos objetivos da abordagem W-Ray é elevar a legibilidade das sentenças e ao mesmo tempo manter a sentença estruturada. Por isso, sentenças complexas podem ser geradas a partir do *template relacionamento n-ário* com boa legibilidade podendo ser simplificadas através da concatenação das *object properties* de um mesmo sujeito. A concatenação das propriedades de um mesmo indivíduo em uma mesma sentença também melhora a legibilidade porque não sobrecarrega a página com um número elevado de sentenças com o mesmo sujeito. Solução semelhante é apresentada em Swoop (capítulo 2).

De acordo com Hollink et al. (2003), são geradas sentenças em LN estruturadas. No entanto, as sentenças W-Ray obtêm esta estrutura através da inclusão de RDFa embutido no HTML. As sentenças também podem funcionar como anotações semiautomáticas para os tipos de dados geográficos em formato vetorial e *raster*.

Apesar da preocupação com a legibilidade das sentenças, detalhes como a geração automática de sentenças com concordância nominal não são importantes porque o principal objetivo das páginas W-Ray é atrair os crawlers.

3.4. Resumo

Este capítulo, apresentou a nova abordagem que torna visíveis dados da *Deep Web* denominada W-Ray e suas etapas. Foi descrito como as *views* materializadas devem ser geradas sobre o banco de dados relacional e quais diretrizes são consideradas essenciais nesta etapa. Foi apresentado como um conjunto de *views* é mapeado para um esquema RDF na abordagem W-Ray, ressaltando a importância de se seguir as recomendações de dados ligados. Foi apresentado como as *views* devem ser geradas para os tipos de dados estatísticos e geográficos. Destacou-se como as descrições semiautomáticas podem ser geradas para imagens de satélite na abordagem W-Ray. Os tipos de *templates* essenciais para a geração das sentenças em LN foram descritos incluindo a complexidade de se gerar automaticamente uma sentença a partir de uma reificação e como é feita a

simplificação da sentença através da concatenação das *object properties* para um mesmo sujeito. Se apresentou como as sentenças são organizadas na página HTML e no como os Web sites são publicados através de páginas HTML com ou sem RDFa embutido. Por fim, algumas comparações entre W-Ray e outras abordagens foram apresentadas.

4
Implementação

Com o objetivo de auxiliar o uso da abordagem W-Ray, foi desenvolvida uma ferramenta, denominada W-RayS. A ferramenta apoia cada uma das etapas da abordagem W-Ray. Inicialmente são apresentados seus módulos e a seguir cada módulo é detalhado. Ao final são apresentadas algumas contribuições finais destacando as vantagens do uso da ferramenta.

4.1. Visão geral da Ferramenta W-RayS

A abordagem W-Ray é semiautomática, pois embora possua um conjunto de etapas automatizáveis, necessita de uma série de decisões do projetista W-Ray para que os dados sejam publicados de forma eficiente; a ferramenta W-RayS foi desenvolvida para dar suporte à abordagem.

A Figura 17 mostra na coluna da esquerda os módulos principais que suportam a abordagem, na coluna do centro as etapas da abordagem W-Ray, e na coluna da direita o resultado de cada etapa.

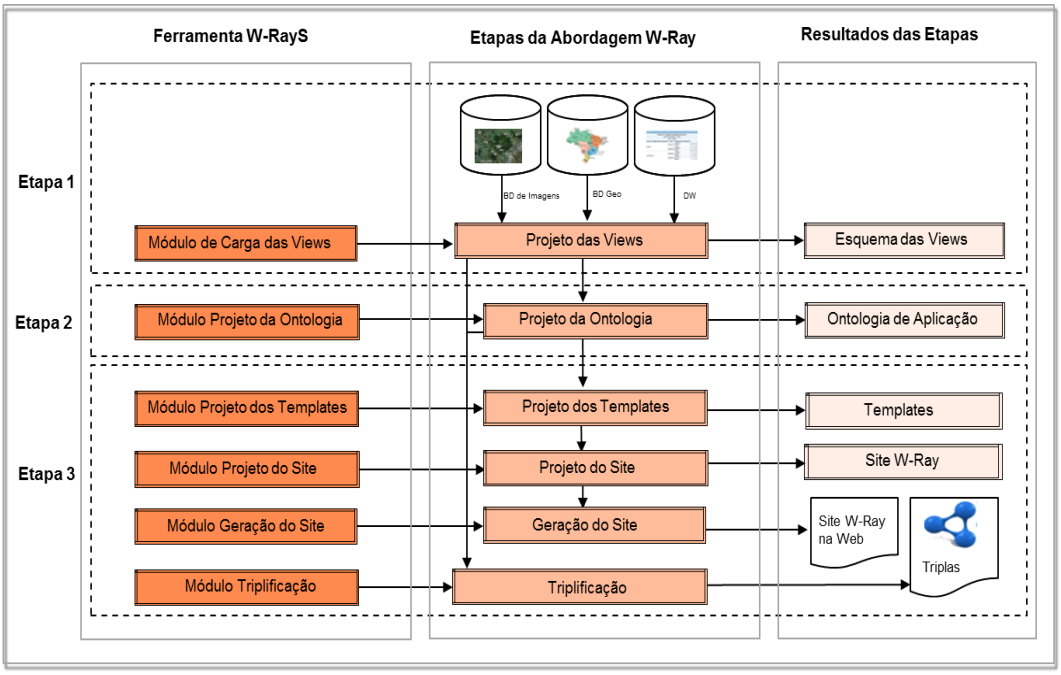


Figura 17 - Etapas da abordagem e respectivos módulos da ferramenta

A ferramenta W-RayS foi implementada em Microsoft Visual Basic 2008[28] (VB8). Alguns módulos, como os de carga de vocabulários, foram implementados em Python 2.5[29].

A ferramenta W-RayS inclui um **Catálogo** que armazena de forma persistente todos os detalhes do projeto. O objetivo do Catálogo é permitir que futuras atualizações de um projeto W-Ray sejam efetuadas sem que o mesmo necessite ser refeito. O Catálogo foi implementado em Oracle e seu esquema pode ser encontrado no Apêndice B.

Na Figura 17 não aparece um módulo para o projeto das *views* e sim um módulo de carga das *views*. Isto ocorre porque o projeto das *views* é realizado usando as ferramentas do SGBD. No entanto, as *views* materializadas devem ser carregadas no catálogo W-RayS. Para esta tarefa, a ferramenta dispõe de um **Módulo de Carga das *Views***, que é capaz de espelhar automaticamente as *views* com seus atributos, chaves primárias e secundárias, no banco de dados da ferramenta. O **Módulo de Carga das *Views*** está customizado para o SGBD Oracle. No entanto, módulos semelhantes podem ser desenvolvidos para outros SGBDs e acoplados à ferramenta W-RayS.

Os demais módulos da ferramenta W-RayS são:

1. **Módulo Projeto da Ontologia**: ajuda o projetista W-Ray a selecionar nas ontologias apropriadas (tesauros ou glossários, desde que estejam expressos em RDF), os termos necessários para o alinhamento com a ontologia de aplicação.
2. **Módulo Projeto de Templates**: ajuda o projetista a alterar os *templates padrão* para melhorar a legibilidade das sentenças em LN.
3. **Módulo Projeto do Web Site**: ajuda o projetista a customizar a estrutura do site que abrigará as sentenças que descrevem os dados.

[28] O Microsoft Visual Basic 2008 (VB8) é uma linguagem de programação criada pela Microsoft. É totalmente orientada a objetos e possui suporte total à UML. É distribuída com o Visual Studio.Net Framework 3.5. (manual disponível em http://msdn.microsoft.com/en-us/library/sh9ywfdk.aspx)

[29] Python é uma linguagem de programação de alto nível com semântica dinâmica, interpretada, orientada a objetos e sintaxe muito simples. O interpretador Python e suas bibliotecas padrão estão disponíveis sem custo para as principais plataformas. (manual disponível em http://docs.python.org/2.5/ref/ref.html)

4. **Módulo Publicação do Web Site**: cria as páginas Web de acordo com a estrutura do site, os *templates*, os mapeamentos de ontologias e os dados provenientes das *views* materializadas.
5. **Módulo Triplificação**: apenas converte os dados das *views* materializadas em triplas, de acordo com os mapeamentos da ontologia. O mapeamento é implementado como um processo batch que materializa as triplas e as armazena em um repositório RDF, criado pela própria ferramenta.

Cada um destes módulos é detalhado a seguir. Um passo a passo da ferramenta pode ser encontrado no Apêndice C.

4.2. Módulo Projeto da Ontologia

O **Módulo Projeto da Ontologia** possui três sub-módulos para suportar a etapa de Projeto da Ontologia. São eles: Sub-módulo Carga da Ontologia, Sub-módulo Alinhamento da Ontologia, Sub-módulo Ontologia da Aplicação. Cada um deles será detalhado a seguir.

4.2.1. Sub-módulo Carga da Ontologia

O sub-módulo Carga da Ontologia permite que o projetista carregue, no catálogo, ontologias, tesauros ou glossários já conhecidos a fim de serem usados posteriormente. O objetivo desta tarefa é facilitar o trabalho de alinhamento, a priori, da *ontologia da aplicação* com ontologias de alto nível ou de domínio, conhecidas e consagradas.

O catálogo possui a ontologia WordNet pré-carregada. Essa versão está em OWL e se encontra disponível no portal da SUMO (WordNet, 2009). Nela, todos os termos da WordNet são instâncias das classes NounSynset, VerbSynset, AdjectiveSynset e AdverbSynset. No entanto, antes do processo de carga da ontologia WordNet no catálogo da ferramenta W-RayS, a sua modelagem foi alterada e todos os verbos foram definidos como *object properties*, e todos os relacionamentos que envolvem a classe VerbSynset foram excluídos. Com estas alterações foram aumentadas as oportunidades de alinhamento com os verbos da WordNet, para os projetistas W-Ray.

4.2.2. Sub-módulo Alinhamento da Ontologia

O sub-módulo Alinhamento da Ontologia é implementado em quatro etapas: *alinhamento de classes; alinhamento de datatype properties; alinhamento de object properties* e *alinhamento de indivíduos*.

Em todas as etapas, o sub-módulo oferece facilidades de navegação e pesquisa para ajudar o projetista a alinhar os termos da *ontologia da aplicação* com os termos de outras ontologias. O módulo também permite que o projetista procure sinônimos com a ajuda do WordNet. Além disso, ele ajuda o projetista a alinhar os indivíduos resultantes das *views* materializadas com indivíduos de outras ontologias. O resultado do alinhamento de um indivíduo é expresso com a ajuda da primitiva *owl:sameAs*.

Os sub-módulos de alinhamento da ontologia suportam os idiomas português e inglês. Um termo em português pode ser alinhado com o termo equivalente em inglês. Podem ser incluídos *labels* nos dois idiomas no esquema RDF final. A Figura 18 mostra um exemplo de um alinhamento de uma *object property* e a Figura 19 mostra a facilidade de navegação e busca para alinhamento.

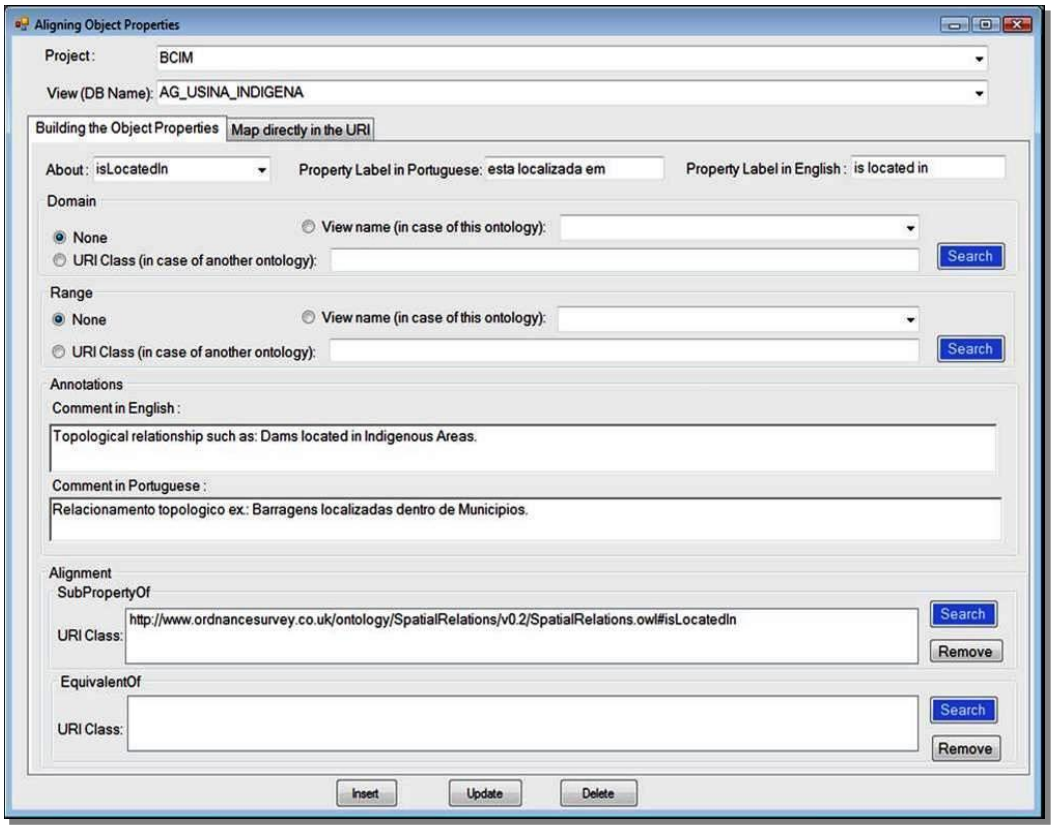


Figura 18 - Formulário para alinhamento de *object properties*

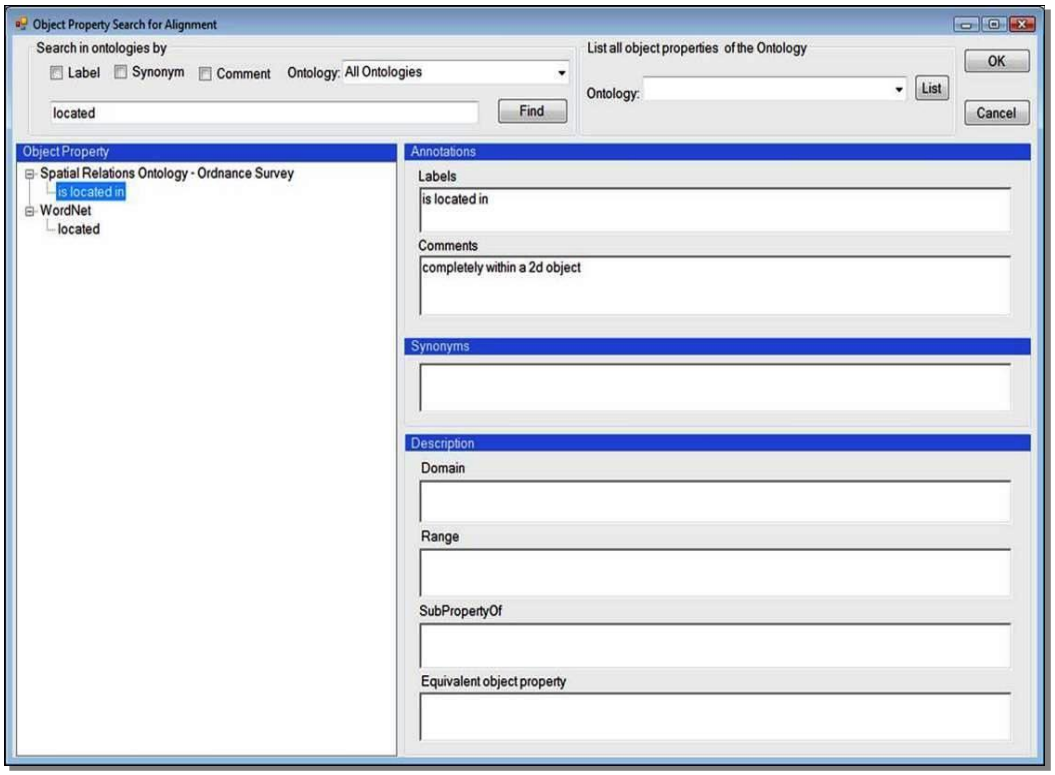


Figura 19 - Formulário de busca de *object properties* em outras ontologias

4.2.3. Sub-módulo Ontologia da Aplicação

O Sub-módulo Ontologia da Aplicação implementa automaticamente uma estratégia similar às recomendações descritas pelo *W3C Direct Mapping of Relational Data to RDF* (Arenas et al., 2011), onde:

- tabelas são mapeadas para classes;
- atributos são mapeados para *datatype properties*;
- relacionamentos binários (definidos através de chaves estrangeiras) são mapeadas para *object properties*;
- relacionamentos n-ários são mapeados em classes e propriedades, através de reificação, de acordo com a recomendação do W3C (Noy & Rector, 2006). Noy & Rector (2006) recomendam que, no caso de relacionamento n-ário, cada tabela deve ser mapeada como uma classe. Até mesmo a tabela que representa o relacionamento n-ário deve ser mapeada como uma classe artificial. Além disso, todos os seus relacionamentos com as outras tabelas devem ser mapeados como *object properties*, onde o domínio de cada uma delas é definido pela classe artificial.

Todos os objetos da ontologia são gerados automaticamente por este sub-módulo e os alinhamentos ficam armazenados de forma persistente no banco de dados da ferramenta, podendo ser alterado posteriormente pelo projetista W-Ray.

4.3. Módulo Projeto de Templates

O Módulo Projeto de Templates define *templates padrão*, para facilitar o mapeamento das classes e propriedades da *ontologia da aplicação* para substantivos, verbos, adjetivos ou advérbios. Conforme descrito no capítulo 3, os *templates padrão* adotados na abordagem W-Ray são:

– *simples*, quando a propriedade é uma *datatype property* com valor único;
– *multivalorados*, quando a propriedade é uma *object property* proveniente de um relacionamento 1-n;
– *relacionamento binário*, quando a propriedade é uma *object property* proveniente de uma relacionamento n-m binário, ou seja, entre duas tabelas;
– *relacionamento n-ário*, quando a propriedade é uma *object property* proveniente da reificação (Noy & Rector, 2006) de um relacionamento n-ário, ou seja, entre mais de duas tabelas.

O módulo segue três conjuntos de regras para a implementação destes *templates padrão*:

- as regras para a reificação dos relacionamentos, propostas por Noy & Rector (2006);
- as regras para a construção de frases com base em diretrizes, como descrito por Fliedl et al. (2010);
- as regras para a construção de frases longas, como em Hewlett et al. (2005);

O *Módulo de Projeto de Templates* é composto por três sub-módulos para suportar a etapa de Projeto de Templates: *Parâmetros de Views; Parâmetros de Atributos; Parâmetros de Relacionamentos*.

Os sub-módulos têm como objetivo fornecer:

- a possibilidade de ajustes nos *templates padrão*;
- a possibilidade de alteração dos nomes das *views*, seus atributos e seus relacionamentos para a publicação nas páginas Web. Isto é importante quando

o projetista, durante o processo de criação das *views*, não se preocupa em usar nomes inteligíveis e representativos;

- facilidade para alinhamento com ontologias carregadas na ferramenta W-RayS sem exigir conhecimento de OWL ou RDF.

Foi observado nos primeiros testes da ferramenta W-RayS, que os *templates padrão*, embora regulares, podem conter um nível elevado de redundância, podendo comprometer a legibilidade das sentenças. Para contornar esse problema, os sub-módulos oferecem a oportunidade de ajustar os *templates padrão*. Todos os ajustes tornam-se persistentes para facilitar novas gerações de sentenças. Os ajustes permitidos são:

– se o nome do atributo deve constar da frase ou apenas o seu valor;
– se o nome do atributo deve aparecer antes ou depois do valor do atributo;
– qual a organização dos atributos no *template*, ou seja, em que ordem devem aparecer na frase;
– inclusão de negrito, itálico e aspas na apresentação dos valores dos atributos na frase;
– inclusão de artigos, pronomes e pontuação;
– concordância verbal, no caso dos relacionamentos;
– concordância nominal, no caso dos nomes dos atributos;
– definição do sujeito da sentença.

Se o projetista W-Ray não fizer nenhum ajuste, as sentenças serão automaticamente geradas de acordo com os *templates padrão*. O único parâmetro fortemente recomendado é o que define o sujeito da sentença, quando esta é proveniente de uma reificação. Neste caso, se o sujeito de uma sentença não for definido, serão criadas automaticamente tantas sentenças quanto for o número de *object properties*, cada uma com um sujeito diferente. A consequência disso pode ser a geração de sentenças mal formadas ou sem significado, que fogem das diretrizes do W3C e Google.

Um ponto importante e que deve ser ressaltado é que cada sub-módulo também fornece a possibilidade de alinhamento com outros vocabulários, sem exigir que o projetista W-Ray possua conhecimento em RDF. Para este fim, foi projetado um módulo com facilidades de navegação e pesquisa, nos termos de

ontologias previamente carregadas no catálogo W-RayS, e que são visualizadas como termos de um glossário, ou seja, sem exigir os detalhes de um alinhamento em OWL. A facilidade implementada também permite que o projetista procure sinônimos com a ajuda do WordNet. A seguir é apresentado um exemplo.

Se o projetista desejar tornar legível o nome da *view* "AG_USINA" e, ao mesmo tempo, definir um conceito para esta *view*, ele necessita procurar conceitos de glossários já existentes (ontologias ou tesauros em RDF). Neste caso, ele pode clicar no botão *search* do sub-módulo ***Parâmetros da View,*** mostrado na Figura 20, que abrirá um formulário de busca por palavra-chave (Figura 21). Esta busca permite verificar os conceitos relacionados aos termos das ontologias disponíveis no catálogo W-RayS. O formulário de busca só listará nomes de classes, porque ele é capaz de reconhecer que o usuário está querendo alinhar uma *view*, isto é, não perde o contexto. Supondo que o usuário selecione o termo *Power Station*, que é uma classe da ontologia *Buildings and Places* (Ordnance, 2008), conforme Figura 21, a ferramenta fará o mapeamento da seguinte maneira:

- o nome do *label* da classe será *power station* (este nome pode ser alterado pelo usuário), conforme (1) na Figura 20.
- a *view*, cujo nome é "AG_USINA", é mapeada para uma classe denominada "PowerStation" que é parte da ontologia que está sendo gerada a partir do esquema das *views*, conforme (2) na Figura 20.
- a classe "PowerStation" mapeada será subclasse da classe "PowerStation" da ontologia *Buildings and Places*.

A mesma estratégia é usada para *object properties*. No caso de *datatype properties*, é feito o reuso da propriedade da ontologia escolhida.

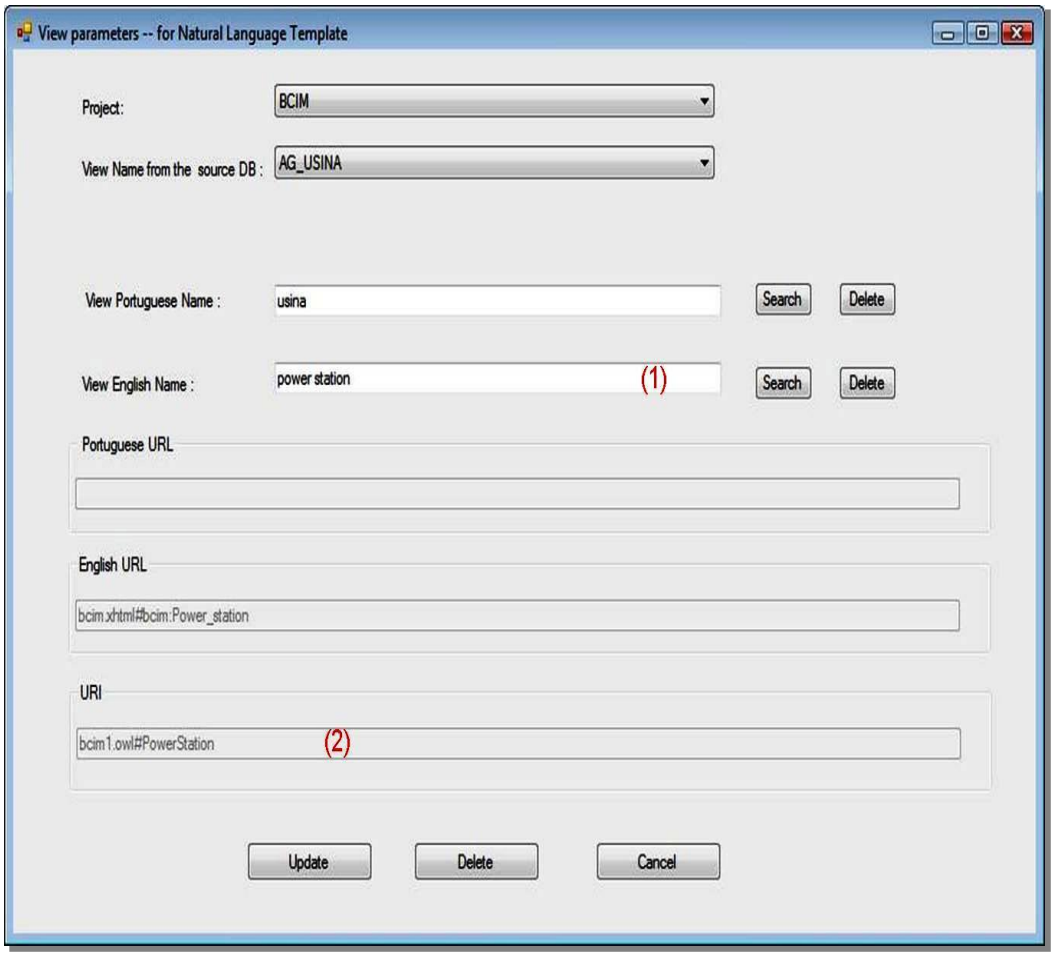


Figura 20 - Módulo de ajuste dos templates de LN

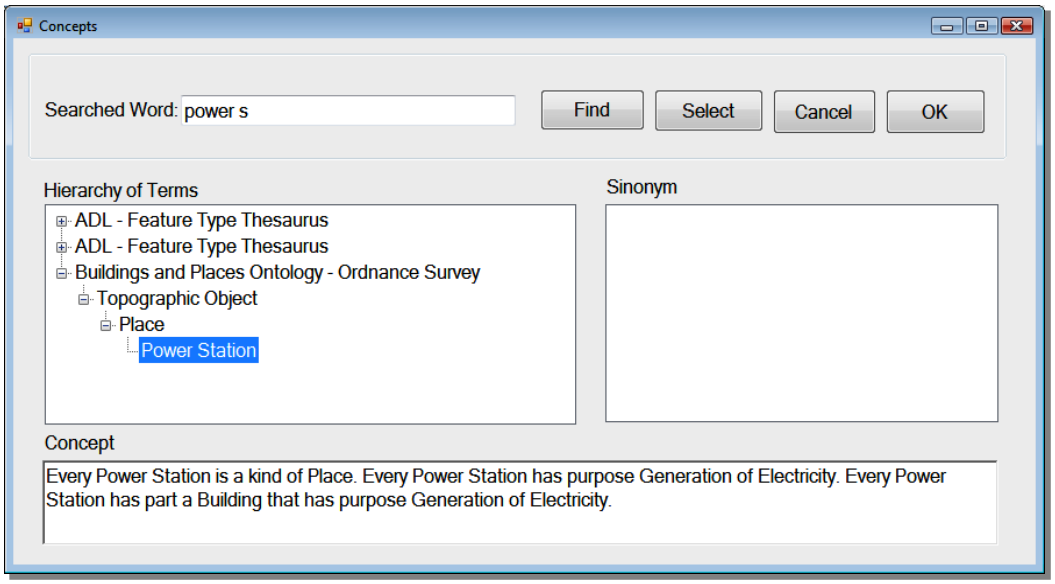


Figura 21 - Módulo que auxilia a busca de conceitos para o alinhamento entre vocabulários

4.4. Módulo Publicação do Web Site

O módulo Publicação do Web Site implementa a estrutura do site com base nos *templates*, nos mapeamentos RDF e nos dados provenientes das *views* materializadas. Três sub-módulos suportam estas tarefas: *Parâmetros do Web Site, Geração de RDFa e Geração de HTML*.

Um pseudocódigo do módulo *Geração de RDFa* encontra-se disponível no Apêndice E.

4.4.1. Sub-módulo Parâmetros do Web Site

O sub-módulo Parâmetros do Web Site é responsável pela definição de parâmetros para melhorar a geração de um site W-Ray.

Uma das tarefas da etapa de publicação dos sites é garantir a implementação de um conjunto mínimo das recomendações do W3C (Caldwell et al. 2008) e da Google (Google-OptimizationGuide, 2012), descritas no capítulo 3. O módulo de geração de páginas HTML é capaz de seguir estas recomendações automaticamente como, por exemplo, utilizando o nome do projeto como título da página e os nomes das classes como subtítulos. No entanto, o usuário pode melhorar o projeto do site, definindo alguns parâmetros tais como:

- Hierarquias de *templates*. Pode ser utilizada para definir títulos e subtítulos ou para compor sentenças.
- Endereços dos formulários Web. Devem ser informados para permitir o acesso direto ao banco de dados subjacente, estabelecendo assim uma ligação entre as páginas Web da superfície e os dados *Deep Web*.
- A forma como as sentenças serão agrupadas nas páginas da Web, o que pode ser implementado de dois modos:
    - default - publica todas as sentenças em apenas uma página;
    - página - publica uma sentença em cada página.
- Ordem em que um tipo de sentença deve aparecer na página. Este parâmetro é válido se mais de um tipo de *template* é utilizado na mesma página.

4.4.2. Sub-módulo Geração de HTML

O sub-módulo Geração de HTML é responsável pela geração de todas as páginas que irão compor o site de um projeto W-Ray. Este sub-módulo gera o HTML sem o RDFa.

A geração de uma página HTML é feita com base nos templates, nos dados provenientes das *views* e nos parâmetros definidos nas etapas do projeto. Este módulo gera um conjunto de páginas da seguinte maneira:

- As sentenças geradas a partir do tipo de *template relacionamento* (*binário ou n-ário*) são agrupadas em uma página, que aponta para outras páginas, que contêm as descrições das classes, ou seja, as sentenças geradas a partir dos *templates simples* e *multivalorados*.
- As sentenças geradas a partir dos tipos de *templates simples* e *multivalorados* são agrupadas em uma ou mais páginas. O sujeito de cada sentença deve apontar para os formulários que permitem o acesso ao banco de dados, para que seja estabelecida a conexão com a *Deep Web*. A conexão é feita através da concatenação da URL, que identifica o formulário de entrada para a *Deep Web,* com o valor da chave primária, que identifica cada sujeito da sentença (este valor funciona como um parâmetro para acessar o dado específico na Deep Web). Esta conexão ocorre em diferentes níveis, dependendo do tipo de domínio do dado ou da segurança requerida pelo usuário. Por exemplo, uma descrição de um objeto geográfico pode apontar diretamente para a sua respectiva localização no mapa. Por outro lado, dados estatísticos desagregados ou microdados são dados confidenciais que só podem ser divulgados após sua agregação. Neste caso, o link definido sobre o sujeito apontará para o formulário de entrada de dados.
- Os nomes das classes e propriedades devem apontar para os respectivos conceitos. Isto só é possível se o projetista incluiu os conceitos ou executou o alinhamento com outros vocabulários. Os termos podem estar ligados aos conceitos de duas maneiras: através de sumários incluídos no final de cada página ou diretamente à um resumo da *ontologia da aplicação*. Na primeira opção são inseridas automaticamente as descrições para os termos existentes nas sentenças no final da página. Também são automaticamente gerados *links* entre os termos das sentenças e essas decrições inseridas. Isto só é possível

quando o projetista W-Ray executa o alinhamento com outras ontologias ou liga o termo a um conceito previmente carregado no banco de dados W-Ray.

Ex.: http://tomcat.inf.puc-rio.br:8080/muralmaps/biomarelevo.html#unidade_geomorfologica

Na segunda opção, se o alinhamento não é executado, o projetista deve inserir um parâmetro via programa informando a localização da página que possui um resumo da ontologia.

Ex.: http://www.inf.puc-rio.br/~hpiccinini/wray/bcim.xhtml#bcim:type_of_energy

A Figura 22 apresenta uma página da Web gerada após um projeto completo para mapas do IBGE.

A Figura 23 ilustra como as páginas podem ser interligadas: (1) página principal do site que aponta para as outras páginas HTML; (2) página com sentenças geradas a partir do *template* relacionamento n-ário; (3) páginas com sentenças geradas a partir do *template* simples e multivalorado; (4) páginas da *Deep web*.

Figura 22 - Página gerada para o mapa de Biomas do IBGE

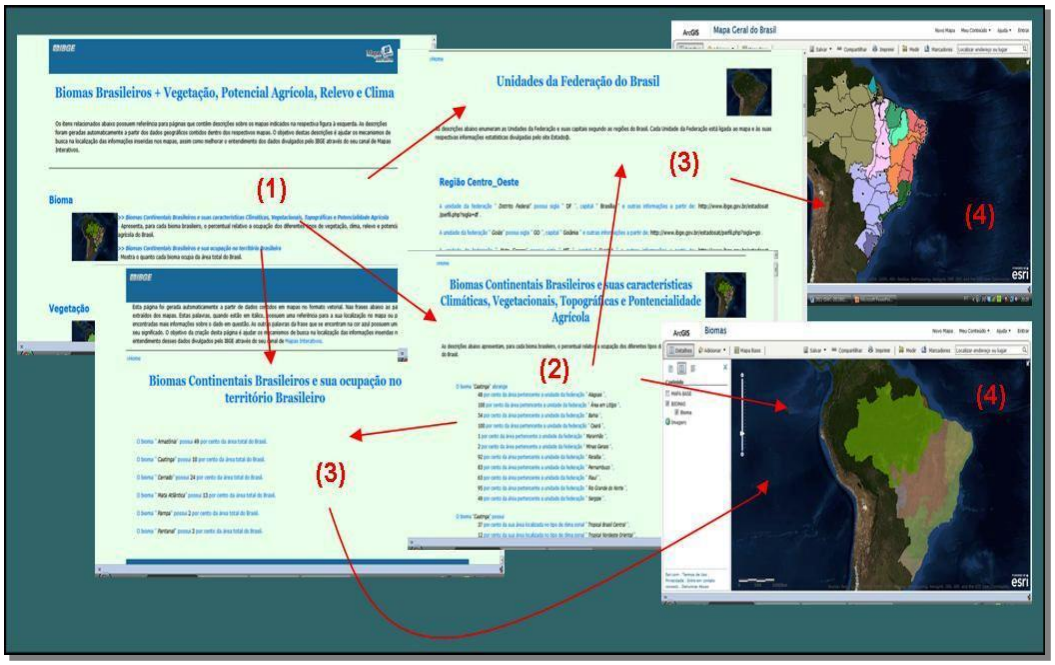


Figura 23 - Exemplo de hiperlinks entre as páginas HTML geradas e dados da *Deep Web*

4.4.3. Sub-módulo Geração de RDFa

O sub-módulo Geração de RDFa, além de executar as mesmas tarefas descritas no *Módulo Geração de HTML,* deve embutir o RDFa no HTML gerado da seguinte maneira:

- os dados provenientes de domínios definidos por enumeração, se corretamente alinhados pelo projetista, serão ligados às instâncias correspondentes dos vocabulários externos;
- a página HTML será descrita de acordo com a *ontologia da aplicação,* gerada para publicação dos dados, onde os termos dessa ontologia serão ligados aos termos dos vocabulários externos, caso o alinhamento tenha sido feito;
- as URIs utilizadas são *"hash URIs",* tanto para os indivíduos como para os termos da ontologia. Os fragmentos que identificam um indivíduo são compostos pela chave primária. Neste caso, é importante ressaltar que, quando o projetista W-Ray é oficialmente o responsável por sua produção e manutenção, então também é dele a responsabilidade pela padronização dos identificadores únicos. Caso contrário, devem ser criados indivíduos, que funcionam como *aliases,* que devem ser alinhados com indivíduos, que possuem notória responsabilidade sobre o dado. Por exemplo, o IBGE é oficialmente o órgão do governo brasileiro responsável pelo mapeamento de

todo o território nacional, o que inclui todos os dados referentes aos municípios brasileiros. Se o INPE decidir publicar indivíduos pertencentes a uma classe *MunicípiosBrasileiros*, ele deve alinhar seus indivíduos com os do IBGE, ao mesmo tempo, que cabe ao IBGE a padronização e definição das URIs que identificam os municípios brasileiros.

O RDFa contido na Figura 25, foi gerado para o trecho de um documento HTML da Figura 24. Este exemplo é referente à publicação de dados ligados, provenientes de dados geográficos em formato vetorial. A sentença da Figura 24 foi gerada a partir de um *template de relacionamento n-ário*.

....
O bioma "*Caatinga*" abrange *48* por cento da área localizada na unidade da federação "*Alagoas*".

Figura 24 - Trecho de um documento HTML.

```
....
<div about="http://www.inf.puc-rio.br/~hpiccinini/wray/biome.owl#-39.84049008,-9.44877857" typeof="biome:Biome">  (1)
    <a href="#bioma">O bioma</a>  (2)
    <span property="geonames:name" content="Caatinga" >  (3)
        <i><a href="bioma.html#a-39.84049008-9.44877857">Caatinga</a></i>  (4)
    </span>  (5)
    <a href="#abrange">abrange</a>  (6)
    <dl rel="biome:encloses">  (7)
        <dd about="http://www.inf.puc-rio.br/~hpiccinini/wray/biome.owl#-39.84049008,-36.71609664,-9.44877857,-9.64509767"
        typeof="biome:Coverage">  (8)
            <span property="biome:percentOf" datatype="xsd:decimal">48</span>  (9)
            <a href="#por_cento_de">por cento da área</a>  (10)
            <a href="#pertencente_a">pertencente a</a>  (11)
            <a href="#unidade_da_federacao">unidade da federação</a> <i>  (12)
            <span rel="biome:locatedIn">  (13)
                <span about="http://www.inf.puc-rio.br/~hpiccinini/wray/biome.owl#-36.71609664,-9.64509767"
                typeof="bcim:State">  (14)
                    <span property="geonames:officialName" content="Alagoas">  (15)
                        <a href="uf.html#a-36.71609664-9.64509767">Alagoas</a>  (16)
                    </span>  (17)
                </span>  (18)
            </span>  (19)
            </i>.  (20)
        </dd>  (21)
    </dl>  (22)
</div>
```

Figura 25 - Trecho de um HTML com RDFa embutido

O RDFa apresentado na Figura 25 é composto pelas seguintes linhas:

Linha 1 - A tag abre o parágrafo da sentença que contém o atributo *about*, com a URI do indivíduo da classe *Biome*. A tag só é fechada no final do exemplo (linha 23).

Linha 3 - marca o nome de uma *feature* geográfica, através da *datatype property geonames:name* da ontologia Geonames, para o indivíduo da classe *Bioma*. Note que o atributo *content* é usado para definir que o dado é do tipo literal.

Linha 7 - utiliza o atributo *rel* para marcar a *object property biome:encloses* onde o sujeito RDF da tripla é da classe *Biome* e o objeto RDF é da classe *Coverage*. A tag <dl>, contendo a *object property*, só é fechada no final do exemplo e está contida na tag .

Linha 8 - A tag <dd>, com o atributo *about*, marca a URI do indivíduo da classe *Coverage*. Este indivíduo é o objeto, na tripla formada pela *object property biome:encloses* e, ao mesmo tempo, é o sujeito, na tripla formada pela *object property biome:locatedIn.* A classe *Coverage* representa o mapeamento do relacionamento n-m do esquema das views.

Linha 9 - marca a propriedade *biome:percentOf* - valor 48 - do segundo sujeito, que é um atributo do relacionamento n-m.

Linha 13 - utiliza o atributo *rel* para marcar a *object property biome:locatedIn* da tripla RDF, onde o sujeito RDF é da classe Coverage e o objeto RDF é da classe State.

Linha 15 - marca a propriedade *geonames:officialName* do indivíduo da classe *State*.

4.5.Módulo Triplificação

A triplificação é feita a partir dos dados das *views* materializadas carregadas no catálogo W-Ray e com base na *ontologia da aplicação* resultante do mapeamento RDB-to-RDF.

As URIs são geradas, concatenando-se o *namespace* (definido pelo nome do projeto W-Ray) com a chave primária dos dados.

O procedimento de triplificação é o mesmo utilizado para a geração das triplas RDF embutidas no HTML.

O arquivo gerado é um *dump* do esquema das views armazenado em um repositório próprio.

4.6. Considerações Finais

A persistência de um projeto W-Ray permite que, quando as *views* materializadas forem atualizadas, todas as páginas HTML podem ser regeradas sem a necessidade de se refazer o projeto W-Ray, exigindo apenas o trabalho de recarga das *views*.

A persistência dos alinhamentos fornece a possibilidade de se refazer, no caso de atualizações do dado, a triplificação sem a necessidade de que todo o alinhamento seja refeito.

Devido ao fato de que as *views* materializadas são importadas para o catálogo da ferramenta W-RayS, podemos combinar dados pertencentes a mais de um banco de dados relacional em um mesmo projeto W-Ray e, consequentemente, em uma mesma ontologia de aplicação, como proposto em (Konstantinos et al., 2010).

A fase do Projeto das *Views* é totalmente manual. Esta fase é realizada pelo projetista W-Ray (administrador de dados), assim como o é nas ferramentas de triplificação que seguem os métodos de mapeamento de ontologias de domínio.

Diferentemente de algumas ferramentas de triplificação, como D2R Server, a ferramenta W-RayS não requer conhecimento de linguagens de mapeamento para alinhamento entre ontologias e seleção de dados, bastando interagir com interfaces amigáveis.

O projetista W-Ray pode ou não optar pela publicação de dados ligados ou por melhorar a legibilidade das sentenças.

A ferramenta W-RayS dá suporte à geração de sentenças estruturadas, apoiada por vocabulários controlados, da mesma forma que a ferramenta descrita em (Hollink et al. 2003), onde o WordNet é o vocabulário nativo na ferramenta. A ferramenta W-RayS também permite a inclusão de outros vocabulários. Durante esta etapa o projetista W-Ray pode fazer o alinhamento a priori com outras ontologias sem que seja exigido um conhecimento de OWL. Se o projetista possuir conhecimento de OWL e fizer a opção pela publicação de dados ligados, a ferramenta oferece facilidades para alinhamentos mais ricos.

Qualquer que seja o nível de conhecimento do projetista sobre OWL, ou seja, se a etapa de alinhamento da ontologia foi executada com ou sem conhecimento de OWL (conforme descrito no item 4.3), a ferramenta mantem a semântica dos dados estruturados ao publicá-los como texto e agrega mais semântica a estes dados. Desta forma, os mecanismos especializados em busca na Web de dados podem localizá-los e fazer inferências sobre os mesmos.

A ferramenta W-RayS consegue melhorar a legibilidade e ao mesmo tempo manter a sentença estruturada. Sentenças complexas são geradas a partir do

template relacionamento n-ário com boa legibilidade e podem ser simplificadas através da concatenação das *object properties* de um mesmo sujeito.

Toda esta estrutura do site pode ser gerada automaticamente, porque a ferramenta W-RayS adota algumas definições default. Entretanto, para que as páginas possam ser organizadas com um formato voltado para humanos, alguns parâmetros podem ser definidos pelo usuário a fim de melhorar a apresentação da página.

Dentro da ferramenta W-RayS qualquer tipo de dado possui o mesmo tratamento. A ferramenta é totalmente independente do tipo de dado que está sendo publicado. Ela não oferece nenhuma facilidade para dados geográficos ou estatísticos. Na verdade, apenas na etapa de criação das *views* é oferecido um tratamento diferenciado para os dados geográficos e estatísticos, conforme diretrizes definidas no capítulo 3.

Por fim, os Web sites gerados pela ferramenta W-RayS podem ser publicados em qualquer idioma. Bastando para isso que as *views* sejam geradas com seus metadados e dados no idioma desejado e que os programas sejam customizados para organizar as sentenças automaticamente no idioma em questão. Atualmente a ferramenta que implementa a abordagem se encontra preparada para geração de páginas HTML em Inglês e Português.

4.7. Resumo

Neste capítulo foi apresentada a ferramenta W-RayS que implementa e apoia as etapas da abordagem W-Ray. Uma visão geral da ferramenta foi apresentada, a fim de localizar o problema, e em seguida cada módulo foi detalhado. Foi descrito o procedimento para a publicação de dados ligados, através da ferramenta W-RayS, enfatizando a importância desta tarefa. Foi descrito como customizar um site W-RayS e como a geração de páginas HTML, com e sem RDFa, é executada. Foi apresentado um exemplo detalhado da geração de uma sentença em LN publicada em HTML com RDFa embutido. Por fim, foram relatados alguns detalhes da ferramenta e suas vantagens.

5
Aplicação da abordagem W-Ray a casos reais

Este capítulo descreve a aplicação da abordagem W-Ray, com suporte da ferramenta W-RayS, para gerar sites com informações reais provenientes de dados do IBGE e INPE.

Foram desenvolvidos quatro sites W-Ray, cada um com base em diferentes fontes de dados: dados estatísticos, dados de mapas em foramato vetorial e dados de imagens de satélite em formato *raster*.

Este capítulo descreve cada caso detalhadamente. No próximo capítulo, é analisada a experiência do uso da abordagem W-Ray e da ferramenta W-RayS.

5.1. Descrição dos projetos

Com o objetivo de avaliar a abordagem W-Ray, foram utilizados os dados da *Deep Web* de duas instituições do governo brasileiro: o Instituto Brasileiro de Geografia e Estatística (IBGE) e o Instituto Nacional de Pesquisas Espaciais (INPE).

O IBGE é oficialmente o órgão responsável pelas pesquisas estatísticas do Brasil e pelo mapeamento geográfico de todo o território brasileiro. Uma das missões do IBGE é a divulgação dos seus dados estatísticos e geográficos através da Web convencional. Neste contexto, utilizou-se a abordagem W-Ray para dar visibilidade aos dados estatísticos e dados em formato vetorial dos seguintes produtos, disponíveis no site do IBGE, e armazenados em bancos de dados da *Deep Web*:

- Sistema de Dados Agregados (SIDRA) (Figueredo & Masello, 2005) – é o sistema mais acessado, no site do IBGE, no que se refere aos dados estatísticos (http://www.sidra.ibge.gov.br/)
- Base Cartográfica Vetorial Contínua do Brasil ao Milionésimo (BCIM) - é um dos produtos mais procurados no IBGE (http://mapas.ibge.gov.br/).

- Mapas Escolares ou Murais - voltados para estudantes do ensino fundamental (http://mapas.ibge.gov.br/).

No quarto caso, a abordagem W-Ray foi utilizada para prover acesso a dados em formato *raster*, armazenados no catálogo de imagens de satélite do INPE. A missão do INPE é *"produzir ciência e tecnologia nas áreas espacial e do ambiente terrestre e oferecer produtos e serviços singulares em benefício do Brasil"*. Este catálogo de imagens, que se encontra na *Deep Web*, pode ser acessado através de formulários HTML a partir de http://www.dgi.inpe.br/CDSR/.

Com exceção do experimento SIDRA, que foi publicado apenas na Web Convencional, todos os outros foram disponibilizados tanto na Web convencional como na Web de dados.

Com o objetivo de validar o RDFa gerado pela ferramenta W-RayS, foi utilizado o RDFa Developer 1.1.1, um add-on que pode ser facilmente acoplado ao navegador Firefox. Ele permite a identificação de todas as triplas RDFa embutidas em uma página Web. Além disso, a validação das páginas foi feita através dos seguintes validadores disponibilizados pelo W3C: RDFa Validator (RDFa-Validator, 2012), RDFa Destiller and Parser (RDFa-Destiller, 2012) e HTML Validator (HTML-Validator, 2012).

Os sites desenvolvidos pela abordagem W-Ray estão disponíveis nos seguintes endereços:

SIDRA: http://www.sidra.ibge.gov.br/SIDRA_WRAY/default.htm

BCIM: http://tomcat.inf.puc-rio.br:8080/bcim/

Mapas Murais: http://mapas.ibge.gov.br/interativos/ferramentas

Imagens de Satélite: http://tomcat.inf.puc-rio.br:8080/image/

5.2. Projeto SIDRA

O Sistema de Banco de Dados Agregado (SIDRA) (Figueredo & Masello, 2005) é o principal sistema on-line para consulta de dados estatísticos agregados do IBGE. SIDRA é organizado como um grande *data warehouse*, contendo dados previamente agregados, provenientes de pesquisas e censos do IBGE, relativos a demografia, comércio, agricultura, indústria, emprego, índices de inflação, dentre outros.

As motivações para a escolha deste projeto foram:

- O desafio de trabalhar com tipo de dados estatísticos que requerem um tratamento diferenciado, devido ao grande volume de dados;
- Testar a escalabilidade da abordagem W-Ray em grandes volumes da dados;
- Aplicar a abordagem em dados que nunca tivessem sido indexados pelos motores de coleta;
- Testar a geração de sentenças na língua portuguesa.

Até a implementação deste caso, os motores de coleta não indexavam o SIDRA, uma vez que sua interface é baseada em formulários dinâmicos e seu arquivo robots.txt bloqueava a entrada dos módulos de coleta. Com a aplicação da abordagem W-Ray foi gerado um Web site em português (ainda acessível a partir de: http://www.sidra.ibge.gov.br/SIDRA_WRAY/default.htm), que expôs os dados do banco de dados SIDRA para os rastreadores durante o período de cinco meses. Durante este período o banco de dados esteve disponível sem bloqueio via arquivo robots.txt.

Conforme indica a abordagem W-Ray, foram realizadas as seguintes etapas: Projeto das Views, Projeto da Ontologia e Publicação do Site. Cada uma delas está detalhada a seguir.

5.2.1. Projeto de Views

O projeto de criação das *views* foi discutido com especialistas do IBGE com o objetivo de fornecer visibilidade aos dados mais importantes do banco de dados SIDRA, de forma resumida, sem perda de integridade e com segurança, ou seja, permitindo a localização apenas de dados que pudessem ser disponibilizados ao público. O SGBD utilizado pelo banco de dados SIDRA é o Oracle.

Os dados estatísticos, por serem dados não convencionais, requerem uma atenção maior na criação das *views* que, na abordagem W-Ray, funcionam como insumo para a geração das sentenças. Um cubo de dados SIDRA normalmente tem as seguintes dimensões:

- período de tempo que os dados se reportam;
- espaço territorial referenciado;

- variável quantitativa ou agregada que está sendo medida em cada célula (população, número de famílias, etc);
- outras variáveis, denominadas classificações estatísticas (sexo, estado civil, etc.) que, por sua vez, possuem seus valores definidos por categorias (masculino, feminino,...);
- cada categoria de uma classificação pode ser combinada, em um produto cartesiano, com as categorias das demais classificações.

Os fatos são definidos como assuntos (Nascimento, Nupcialidade, Atividade, etc), que estão relacionados aos vários cubos de dados existentes para uma determinada pesquisa ou estudo (Censo Demográfico, Pesquisa Mensal de Emprego, Índice de Preços ao Consumidor). As células contêm os valores das variáveis de agregação.

Segundo a abordagem W-Ray pode-se usar duas estratégias para publicar um cubo estatístico: a primeira é modelando um cubo como uma tabela plana e a segunda como uma tabela cruzada.

A primeira estratégia (e mais ingênua) se caracteriza por criar uma frase para descrever cada célula do cubo, usando o modelo de tabela plana. A seguir é apresentado um cubo de dados (tabela 2647) e uma célula.

Pesquisa: Registro Civil
Assunto: Nascimento
Tabela 2647:

Dimensões	Categorias	#Categorias
Variável*: Pessoas Nascidas vivas*		
Mês do registro	Janeiro, Fevereiro, ….	12
Ano	2004 to 2009	6
Sexo	Masculino, Feminino, Ignorado	3
Local do nascimento	Hospital, casa, ...	4
Idade da mãe no nascimento	< 15 anos; 15 to 19 anos, …	45
Número de filhos por nascimento	1, 2, 3,...	4
Região do registro da criança	País, Estado, Município, ...	6393

Célula 63	Janeiro	2009	Feminino	Hospital	< 15	1	Rio de Janeiro

Para a célula acima, a seguinte sentença deveria ser gerada com um modelo de tabela plana:

"*Rio de Janeiro* é um **estado** que possui *63* **pessoas nascidas vivas**, no(a) **mês do registro** *janeiro*, no(a) **ano** *2009*, com **sexo** *Feminino*, **local de nascimento** *hospital*, **idade da mãe na ocasião do parto** *menor que 15 anos* e **numero de crianças por nascimento** igual a *1*."

Modelando o cubo (tabela 2647) como uma tabela plana, seria gerado um cubo com 994.239.360 células (1 x 12 x 6 x 3 x 4 x 45 x 4 x 6393), ou com 4.587.616.800 células (2 x 13 x 6 x 4 x 5 x 46 x 5 x 6393), se forem consideradas as variáveis derivadas e os totais de cada classificação. Uma vez que o BD SIDRA possuía 2.166 cubos na data do desenvolvimento deste projeto, o número de células seria da ordem de 10^{13} e consequentemente o mesmo número para as sentenças, o que é inviável.

Diante deste cenário, a solução adotada foi direcionar os usuários através das sentenças para cada formulário SIDRA referente à tabela que está sendo descrita (página HTML (1) da Figura 26) e não para o resultado da consulta que mostra o valor do dado agregado (página HTML (2) Figura 26). Desta forma, as sentenças são formadas apenas pelos metadados dos cubos estatísticos com hiperlinks para os respectivos formulários (conforme descrito no capítulo 2).

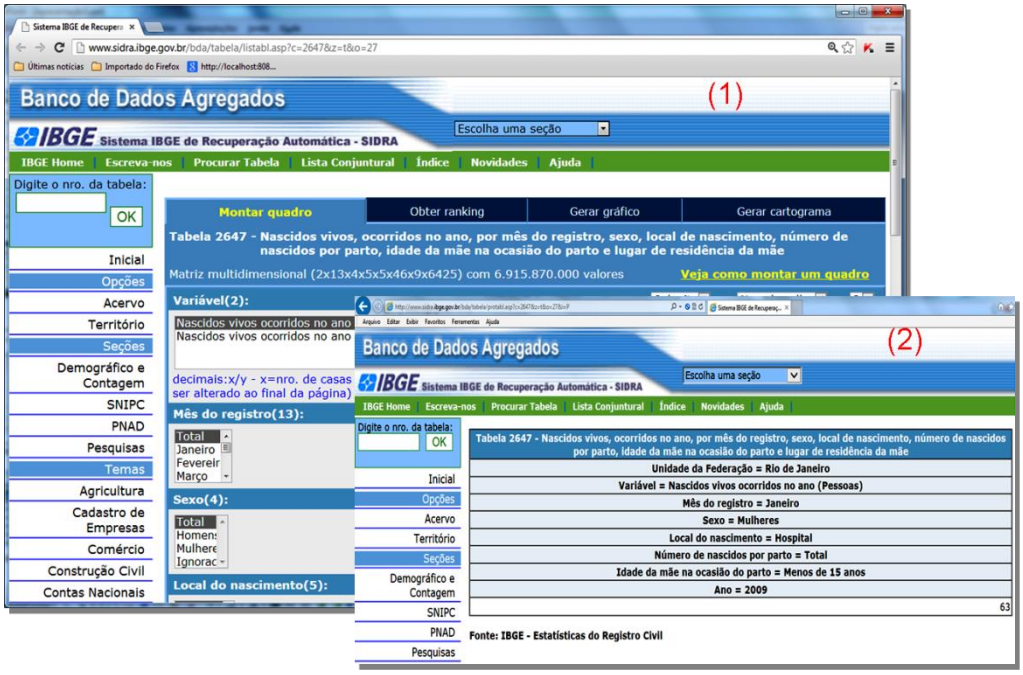


Figura 26 - SIDRA: Formulário HTML e página HTML dinâmicos

Os formulários HTML do sistema SIDRA possuem uma particularidade importante: todos os formulários HTML de entrada de dados são páginas dinâmicas, criadas através de um método *get* que, após o seu preenchimento, geram novas páginas dinâmicas através do método *post*, cujo objetivo é recuperar e exibir os valores para os cruzamentos montados sobre uma tabela pelo usuário. Isto é possível porque o SIDRA trabalha com um banco de metadados, que

subsidia a geração dos formulários dinâmicos para cada cubo de dados, e um banco de dados, que fornece os valores dos cubos. No exemplo da Figura 26, (1) é o formulário dinâmico gerado para a ***tabela 2647*** através do método *get* e (2) é a página dinâmica, que exibe o resultado de uma agregação feita a partir de (1), através do método *post*. Com isso observa-se que as sentenças geradas segundo a abordagem W-Ray, mesmo tendo sido formadas apenas com os metadados do cubo estatístico, continuam sendo capazes de dar visibilidade a dados da *Deep Web,* uma vez que estes metadados também estão armazenados em bancos de dados da *Deep Web*.

No projeto SIDRA foi decidido publicar apenas os metadados dos cubos estatísticos. Esta decisão ainda não foi suficiente para reduzir o número de sentenças publicadas. Entretanto, com esta solução passou-se a ter a possibilidade de cortar uma ou mais dimensões do cubo de dados desde que a enumeração dos metadados referente às dimensões cortadas fosse mantida dentro de cada página HTML. A seguir estão descritas as tentativas de redução do volume de dados no projeto das *views*:

Primeira tentativa:

Foram mantidas nas *views* todas as dimensões de cada cubo de dados. Como conseqüência, cada página da Web tinha um título, que correspondia ao nome da pesquisa estatística, e um sub-título, referente ao espaço territorial. As páginas Web foram quebradas por localização geográfica da pesquisa. Este modelo de *views* acarretaria a geração de aproximadamente 287.000 páginas Web. Um número ainda inviável.

Segunda tentativa:

A dimensão referente ao período de tempo seria agregada ao nome do cubo de dados e, assim, cada página da Web teria um título correspondente à pesquisa que incluía o período da pesquisa (tal como "Censo Demográfico 2010"). Também foi eliminada a dimensão *localização geográfica*, que foi transferida para o fim de cada página como um sentença única, contendo todos as localidades abrangidas pela pesquisa. Uma *view* criada para o principal foco da agregação ou assunto, funcionaria como sub-título. Com esta alteração, o número total de páginas Web foi reduzido para 743. No entanto, o número total de sentenças ainda continuou elevado, devido ao produto cartesiano das categorias das classificações como, por exemplo: cruzamento das categorias da classificação *sexo* (categorias:

feminino, masculino, ignorado), com as da classificação *faixa de idade* (categorias: <15 anos, de 15 a 18 anos,...).

Terceira tentativa:

Foi mantido o modelo anterior e foi eliminada a dimensão referente às categorias das classificações, para reduzir o efeito do produto cartesiano. As categorias eliminadas foram transferidas para o final de cada página. Foram criadas *views* para as categorias de classificação e para a localização geográfica, que já havia sido transferida, a fim de gerar as sentenças no final das páginas. As classificações, contidas nas sentenças que descrevem o cubo, foram ligadas às suas respectivas categorias, através de hiperlinks dentro de cada página. Este foi o modelo final utilizado neste projeto.

A Figura 27 mostra um fragmento de um exemplo de uma página da Web gerada para o SIDRA. A Figura 28 mostra a estratégia da tabela cruzada para o esquema das *views* aplicado para a geração do tipo de sentença (2) mostrada na Figura 27. Note que para cada tipo de sentença é necessário a criação de uma ou mais views, que incluem o conjunto de dados que o usuário deseja mapear para uma única sentença.

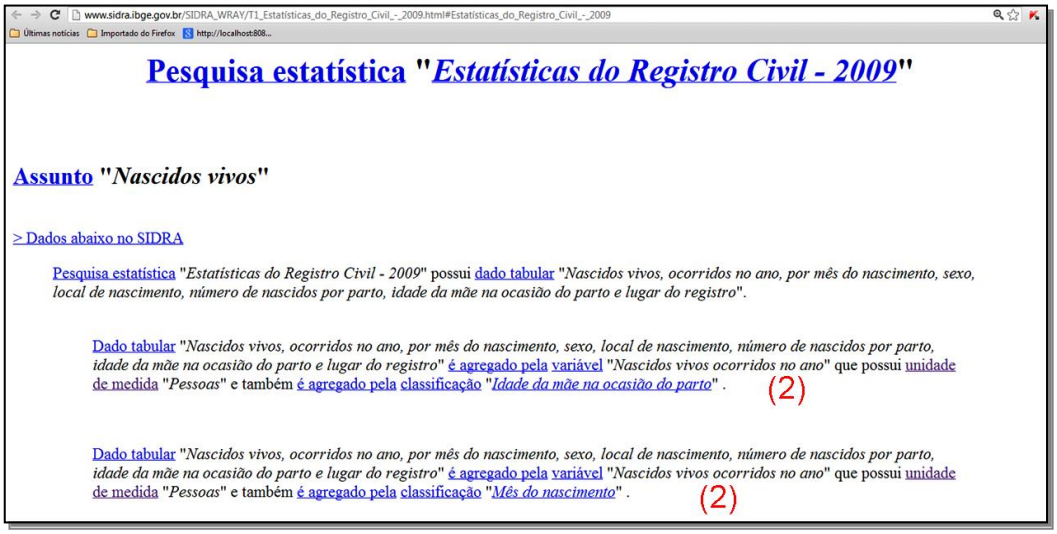


Figura 27 - Fragmento de uma página W-Ray gerada para o BD SIDRA

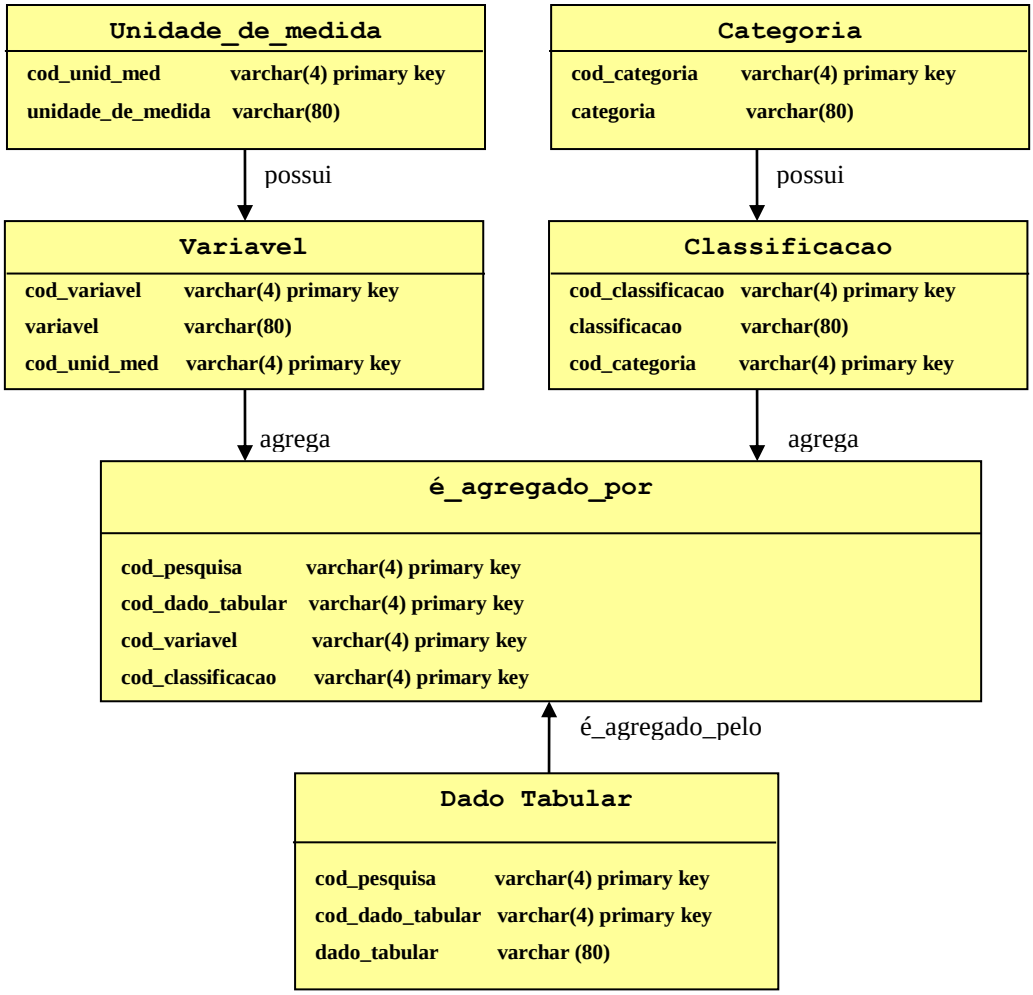


Figura 28 - Modelo das *views* para frase (2) indicada na Figura 27

A Tabela 4 relaciona as *views* geradas para o projeto SIDRA e suas descrições.

Tabela 4 - *Views* materializadas sobre o BD SIDRA

Nome da View	Descrição da View
OCORRENCIA_DE_PESQUISA	*Pesquisas do IBGE em um período de tempo.(Título)*
ASSUNTO	*Assunto ou tema que os dados agregados se referem.(subtítulo)*
DADOS_TABULARES	*Dados tabulares por pesquisa.*
E_AGREDAGA_POR	*Cruzamento (relacionamento ternário entre dado tabular classificação e variável).*
VARIAVEL	*Variáveis agregadas.*
UNIDADE_DE_MEDIDA	*Unidades de medida da variável agregada.*
CLASSIFICACAO	*Classificações de um dado tabular.*
CATEGORIA	*Categorias das classificações (Final da página)*
UNIDADE_TERRITORIAL	*Unidade territorial por pesquisa e no ano da pesquisa.(Final da página)*

5.2.2. Projeto da Ontologia

Como não há ontologia estabelecida para o domínio estatístico, foram adotados os seguintes glossários de termos estatísticos (traduzidos para o

Português): SDMX (SDMX, 2005), implementado pela OCDE (Organization for Economic Cooperation and Development); Glossary of Statistical Terms (OECD, 2007); e EUROSTAT Glossary of Statistical Terms (EUROSTAT, 2012). A Figura 29 mostra o glossário de termos customizado para o experimento SIDRA.

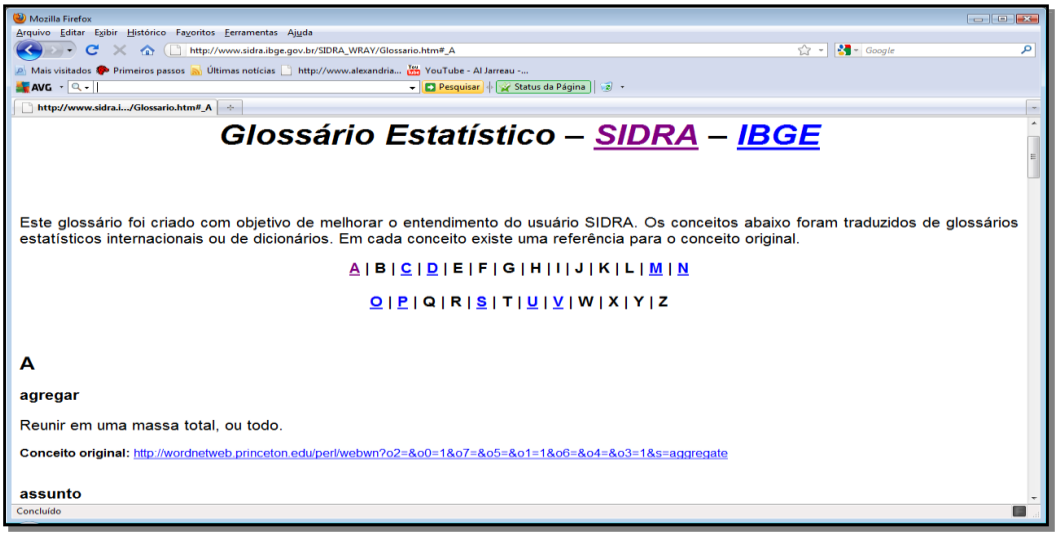


Figura 29 - Página W-Ray-SIDRA contendo glossário de termos estatísticos

5.2.3. Publicação do Web Site

Como o SIDRA divulga seus dados por pesquisas, decidiu-se que esta seria a maneira mais natural de estruturar as páginas Web. Em conformidade com o projeto de *views*, descrito anteriormente, o site que descreve o SIDRA foi gerado automaticamente. Quanto ao site do projeto é importante destacar:

– foi criada uma página inicial que contém uma lista de pesquisas e censos por período de tempo (Figura 30 – Page 1);
– cada entrada desta lista possui um hiperlink para uma página Web, com frases que descrevem o respectivo censo ou pesquisa (Figura 30 - Page 2);
– antes de cada sentença que, descreve os cubos de dados, foi gerado um hiperlink para o respectivo formulário de consulta a esses cubos de dados. As mesmas sentenças possuem hiperlinks para as sentenças que descrevem as classificações, com suas categorias utilizadas nos cubos de dados. Todos os termos usados nas sentenças possuem hiperlinks para a página que contém o glossário estatístico (Figura 30 - Page 2, 3 e 4).

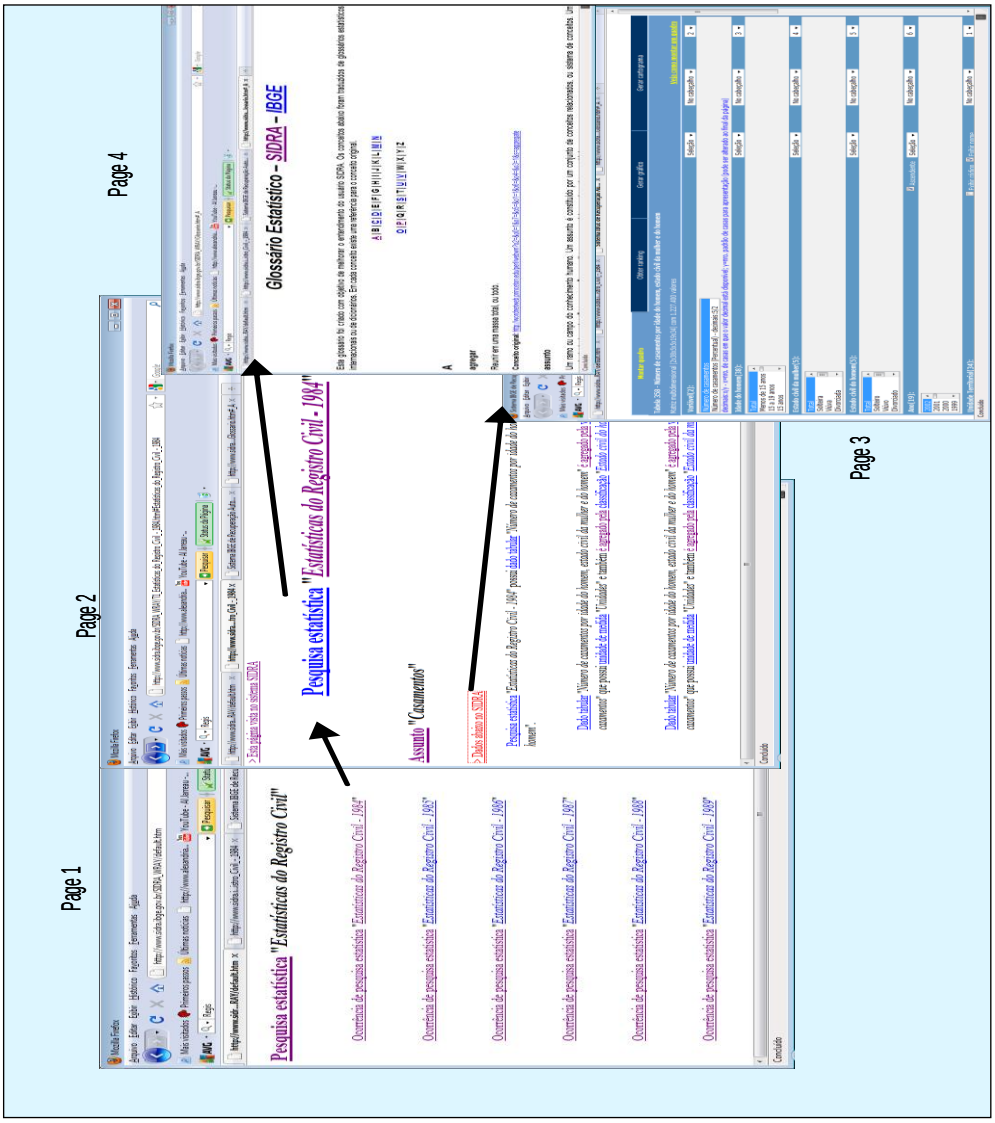


Figura 30 - Hiperlinks entre as páginas HTML e dados da *Deep Web*

5.3. Projeto BCIM

A Base Cartográfica Vetorial Contínua do Brasil ao Milionésimo (BCIM) contém dados cartográficos e metadados sobre hidrografia, hipsografia, localidades, limites políticos, edificações, sistemas de transporte e vegetação. Ela pode ser acessada através do site do IBGE, via o Sistema de Informações Geográficas do IBGE (http://mapas.ibge.gov.br/interativos/sig-ibge-aplicativo), e desde janeiro de 2013, também na galeria ArcGis-online do IBGE (ArcGis, 2012) (http://mapas.ibge.gov.br/interativos/galeria-arcgis-online).

O IBGE não publica resumos de dados de seus mapas e suas cartas, com exceção de alguns metadados definidos manualmente para estes produtos. No

entanto, este procedimento manual exige um tempo considerável e não abrange a maioria dos produtos disponíveis.

As motivações para a escolha deste projeto foram:

- Testar a abordagem W-Ray com dados em formato vetorial;
- Aplicar a abordagem em dados que nunca tivessem sido indexados pelos motores de coleta;
- Testar a geração automática de sentenças na língua inglesa;
- Testar todo o processo de publicação das sentenças na Web de dados.

Este projeto foi publicado na Web de dados, com sentenças geradas em Inglês e Português, em junho de 2011 no site da PUC-Rio. Em março de 2013 o projeto foi simplificado e apenas as sentenças em Inglês relacionadas aos dados da camada de USINAS se encontram disponíveis em: http://tomcat.inf.puc-rio.br:8080/bcim/

Não foi possível comparar o número de acessos através das nossas páginas HTML com o total de acessos à BCIM porque o log do servidor de mapas que o IBGE utilizava na ocasião do experimento não registrava o número de acessos por camada.

Conforme indica a abordagem W-Ray, foram realizadas as seguintes etapas: Projeto das *Views*, Projeto da Ontologia e Publicação do Site. Cada uma delas está detalhada a seguir.

5.3.1. Projeto das Views

O projeto de criação das *views* contou com a participação de especialistas do IBGE. O principal objetivo do projeto era avaliar o comportamento da abordagem para os dados vetoriais. Como a BCIM possui um número grande de camadas e os especialistas do IBGE aconselharam a trabalhar apenas com as camadas de áreas indígenas, usinas e barragens, oleodutos, e municípios. Esta orientação refletia a preocupação existente no âmbito do governo federal, em 2011, com os investimentos em infraestrutura, que poderiam afetar as populações indígenas do país. Nessa ocasião, o IBGE estava recebendo muitos pedidos de produtos geográficos envolvendo este tema e que não eram encontrados pelos usuários, via mecanismos de busca. Considerando a consulta por palavra-chave apresentada na Figura 31(a), ainda hoje, maio de 2013, se um usuário enviar esta pesquisa sobre o

site do IBGE a um motor de busca na Web convencional, o motor de busca não encontrará qualquer mapa, ou mesmo uma página com hiperlink para algum mapa, que contenha informações desta natureza. Isto acontece porque os rastreadores da Web convencional não indexam dados em formato vetorial. Entretanto, se o usuário executar a mesma consulta em inglês, sem especificar o site do IBGE, conforme Figura 31(b). O motor de busca será capaz de indicar o site W-Ray criado para a BCIM na PUC-Rio, onde existe um hiperlink para seus dados na Deep Web. A consulta da Figura 31(b) está em inglês porque no experimento BCIM as páginas HTML estáticas foram geradas em inglês. Se estas páginas estivessem em português, a mesma consulta da Figura 31(a), sem especificar o Web site, poderia ser executada que as paginas W-Ray seriam retornadas pelo mecanismo de busca.

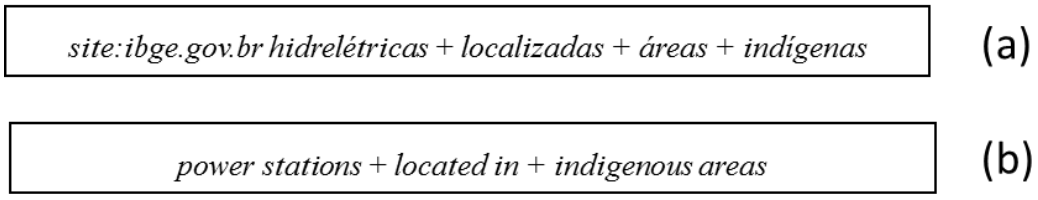


Figura 31 - Consulta por palavra-chave sobre o site do IBGE

Conforme resumo apresentado na Tabela 5, a definição das *views* foi feita seguindo as recomendações para dados vetoriais da abordagem W-Ray e se destacam os seguintes aspectos:

– As definições das *views* devem garantir que os atributos que representam uma chave primária devem ser substituídos por nomes externos significativos, uma vez que estes formarão as sentenças. No caso da BCIM, não foram incluídos os dados da camada de oleodutos, uma vez que tais objetos geográficos não possuíam um nome externo na base de dados BCIM, apesar destes objetos cruzarem áreas indígenas.
– As *views* mais importantes foram definidas com a ajuda de consultas espaciais, que capturaram relações topológicas entre os objetos geográficos. Por exemplo, a camada de áreas indígenas foi associada a uma *view*, que incluía os nomes das usinas *localizadas nas* (relação topológica) reservas, etc.

Tabela 5 - *Views* criadas a partir da BCIM envolvendo usinas, barragens, áreas indígenas e municípios.

Nome da View	Descrição da View
ELEMENTO_GEOGRAFICO	*Elementos Geográficos (título.)*
MUNICIPIO	*Informações não espaciais sobre os municípios.*
AREA_INDIGENA	*Informações não espaciais sobre áreas indígenas.*
USINA	*Informações não espaciais sobre as usinas.*
USINA_INDIGENA	*Informações espaciais envolvendo usinas e áreas indígenas.*
USINA MUNICIPIO	*Informações espaciais envolvendo usinas e municípios.*
BARRAGEM	*Informações não espaciais sobre as barragens.*
BARRAGEM_INDIGENA	*Informações espaciais envolvendo barragens e áreas indígenas.*
BARRAGEM MUNICIPIO	*Informações espaciais envolvendo barragens e municípios.*

Os fragmentos dos URIs aplicados aos indivíduos da classe "Usina" foram criados a partir das chaves primárias do banco de dados BCIM. Essas identificações não são divulgadas pelo IBGE como uma identificação padrão. No entanto, os fragmentos de URIs, que identificam os indivíduos da classe "Município", são provenientes de códigos padronizados pelo IBGE que, neste caso, é oficialmente o dono desse dado.

5.3.2. Projeto da Ontologia

O projeto da ontologia é muito importante, porque a definição dos nomes antes do mapeamento RDB-to-RDF afeta diretamente a capacidade de leitura das sentenças finais. Por exemplo, as *views* contidas nas Tabelas 6 e 7 são correlatas.

Tabela 6 - Exemplo de uma *view* AG_USINA com apenas uma linha de dados e com nomes dos atributos abreviados

CD_USI	NOM_USI	TP_ENERG	LAT	LONG	MET_MOD	FONT_INFO
1	*Passo Real*	*Hidrelétrica*	*-29,0442*	*-53,18587*	compilação	IBGE/DGC/CCAR

Tabela 7 - Exemplo de uma *view* AG_USINA com apenas uma linha de dados e com nomes dos atributos sem abreviação

USINA	NOME_USINA	TIPO_DE_ENERGIA	LATITUDE	LOGITUDE	METODO_DE_MODIFICACAO	FONTE_DE_INFORMACAO
1	*Passo Real*	*Hidrelétrica*	*-29,0442*	*-53,18587*	compilação	IBGE/DGC/CCAR

Considerando a primeira linha da Tabela 6 e que o projetista W-Ray optou pela geração totalmente automática das sentenças, sem nenhum ajuste após a criação das *views*, ou seja, deixando os nomes originais da *view* e dos atributos inalterados. Esta *view* é mapeada diretamente para uma classe com sete *datatype*

properties correspondentes aos seus atributos e, em seguida, esses nomes são aplicados no *template* de geração das sentenças:

> **Passo Real** é uma *ag usina* que possui *tp energ* **hidrelétrica** , *lat* ***-29,0442***, *long* **-53,18587**, *met mod* **compilação** e *font info* **IBGE/DGC/CCAR**.

Se o projetista W-Ray criar as *views* conforme o que é mostrado na primeira linha da Tabela 7, a frase automaticamente gerada será:

> **Passo Real** é uma *usina* que possui *tipo de energia* **hidrelétrica**, *latitude* **-29,0442**, *longitude* **-53,18587**, *metodo de modicacao* **compilação** e *fonte de informacao* **IBGE/DGC/CCAR**.

A última sentença ainda não pode ser considerada satisfatória para a língua portuguesa, porque, no processo de criação das *views*, os SGBDs não permitem notações léxicas como o cedilha, til, acentos, apóstrofo. Por isso, é aconselhável alterar os termos para melhorar a legibilidade. No caso da BCIM, como as sentenças foram geradas na língua inglesa, elas não requereram alteração dos nomes das *datatype properties*. No entanto, alguns ajustes foram feitos no *template* da sentença, tais como, a alteração da ordem das propriedades, e a apresentação de alguns valores em *itálico*, conforme pode ser observado nas sentenças apresentadas na Figura 32.

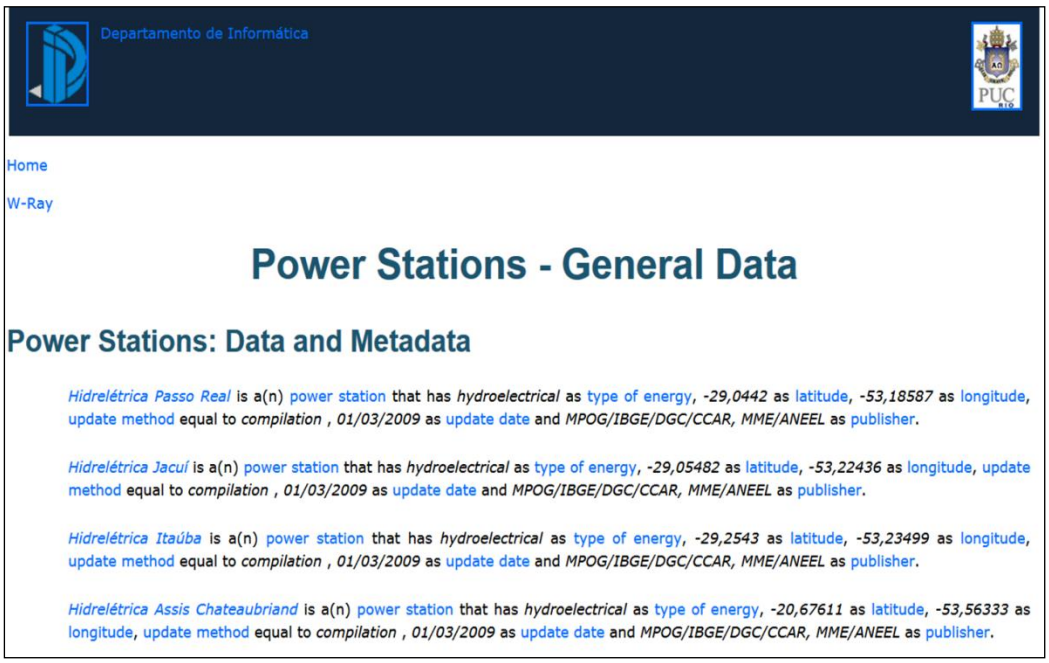


Figura 32 - Página Web correspondente a camada "Usinas"

A fim de garantir a publicação do site gerado pela abordagem W-Ray também na Web de dados, a *ontologia da aplicação* foi alinhada com vocabulários notoriamente conhecidos para o domínio geográfico.

Seguindo as orientações listadas no site W3C Geospatial Ontologies (W3CGeo, 2005) foram utilizados os seguintes vocabulários para melhorar a descrição semântica dos dados publicados: Dublin Core Metadata Element Set com as extensões definidas na versão de janeiro 2008; versão RDF de ADL Feature Type Thesaurus[30] (ADL-RDF, 2004); WordNet em OWL (WordNet, 2009), com as modificações implementadas pela ferramenta W-RayS; ontologia Geonames (GeoNames, 2012); ontologias BuildingsAndPlaces, SpatialRelations and Topography[31] (Ordnance, 2008); WGS84 Geo Positioning[32] (WGS84, 2009).

Várias *datatype properties* destas ontologias foram reutilizadas e algumas *object properties* foram alinhadas, como *subproperty* de propriedades destas ontologias. As classes criadas foram alinhadas como *subclass* de classes de outras ontologias. A *ontologia da aplicação* do projeto *BCIM*, também denominada *bcim*, está descrita em http://www.inf.puc-rio.br/~hpiccinini/wray/bcim.xhtml)

5.3.3. Publicação do Web Site

Como os dados da BCIM são organizados em camadas temáticas, decidiu-se organizar o site de forma semelhante (conforme http://tomcat.inf.puc-rio.br:8080/hpiccinini/bcim). Quanto ao site do projeto BCIM, é importante destacar:

- A página inicial do site tem um índice, listando as camadas temáticas, onde cada entrada possui um hiperlink para a página Web correspondente.
- Cada página web, referente a uma camada temática, agrupa os objetos geográficos da camada. Por sua vez, cada objeto geográfico possui um

[30] ADL Feature Type Thesaurus - conjunto de termos para categorias geográficas projetado para ser usado no Alexandria Digital Library Gazetteer (ADL) - gazetteer criado pela Universidade da California.

[31] BuildingsAndPlaces, SpatialRelations and Topography - conjunto de ontologias do domínio geográfico disponibilizado pela Ordnance Survey - agência do governo britânico responsável pelo mapeamento geográfico da Inglaterra.

[32] WGS84 Geo Positioning - vocabulário em RDF para representar informações de latitude, longitude e altitude, no datum geodésico de referência WGS84.

hiperlink para a BCIM disponível no site do IBGE, fazendo assim a ligação com a *Deep Web*.

– Cada página Web, que inclui os relacionamentos topológicos entre as camadas da BCIM, aponta para as páginas que possuem a descrição dos objetos geográficos.

5.4. Projeto Mapas Murais

O terceiro projeto usando a abordagem W-Ray envolve os Mapas Murais do Brasil, disponíveis no site do IBGE. Os mapas murais são mapas produzidos pelo IBGE para fins escolares que ganharam este nome porque são expostos em murais na sala de aula. Podem envolver vários temas como vegetação, relevo, biomas, potencial agrícola do solo, climas, etc. Abrangem sempre todo o Brasil e são construídos na escala 1:5000000.

As motivações para a escolha deste projeto com dados vetoriais foram:

– Avaliar os benefícios da visibilidade dos dados da *Deep Web*, através da abordagem W-Ray. No projeto BCIM, isso não foi possível, porque o servidor de mapas do IBGE não registrava o número de acessos por camada, o que inviabilizou a comparação do total de acessos às camadas da BCIM com os acessos feitos provenientes do site W-Ray.
– Testar a geração de sentenças provenientes de *templates* do tipo *relacionamento n-ário.,* No projeto *BCIM* foram utilizados apenas *templates* do tipo *relacionamento n-m.*
– Testar a legibilidade de frases longas com mais de um objeto para um mesmo sujeito.

O site gerado, através da abordagem W-Ray, para os mapas murais foi gerado em Português e incluído no site do IBGE que divulga os mapas interativos, por um período de seis meses no ano de 2012. Ao fazer isso, adiciou-se credibilidade institucional ao Web site desenvolvido neste projeto. Atualmente este está disponível em http://www.inf.puc-rio.br/~hpiccinini/Wray/muralmaps.

Conforme indica a abordagem W-Ray, foram realizadas as seguintes etapas: Projeto das *Views*, Projeto da Ontologia e Publicação do Site. Cada uma delas está detalhada a seguir.

5.4.1. Projeto das Views

O projeto de *views* do Projeto Mapas Murais, contou com a participação de especialistas do domínio geográfico do IBGE. Em entrevista com estes especialistas, foi identificado o seguinte cenário:

– historicamente, os mapas murais sempre foram muito procurados em meio impresso no IBGE. Quando publicados em meio digital, através do ArcGis-Online, alguns deles não foram muito acessados.
– dentre os mapas murais, o mapa de Biomas é o produto com maior número de acessos. O mesmo não acontece com outros temas.

Com o objetivo de reverter esta situação, decidiu-se pela criação de um site, seguindo a abordagem W-Ray, que relacionasse o mapa de biomas aos mapas que retratam os seguintes temas: vegetação, relevo, clima, potencial agrícola e unidade da federação. A expectativa era que, após a indexação do novo site pelos motores de coleta, os usuários se sentissem motivados ou conseguissem encontrar com mais facilidade os outros mapas temáticos relacionados com o mapa de Biomas.

As *views* criadas estão descritas na Tabela 8, onde algumas delas representam o relacionamento topológico entre o mapa de biomas e os outros mapas (*localizada_em; composta_por; formada_por; ocupada_pelo; pertencente_a*). Estas *views* só puderam ser criadas porque todos os mapas murais possuem a mesma projeção e utilizam a mesma base cartográfica. A Figura 33 ilustra como os mapas foram interpretados como camadas, para a geração dos relacionamentos topológicos.

Tabela 8 - *Views* geradas sobre os mapas murais

Nome da View	Descrição da View
Bioma	*Informações não espaciais sobre os biomas brasileiros.*
Clima	*Informações não espaciais sobre os tipos clima no Brasil.*
Potencial_Agricola	*Informações não espaciais sobre o potencial agrícola do solo.*
Relevo	*Informações não espaciais sobre os tipos de relevo do Brasil.*
Vegetacao	*Informações não espaciais sobre os tipos de vegetação do Brasil.*
Unidade_da_federacao	*Informações não espaciais sobre as unidades da federação brasileira.*
localizada_em	*Informações espaciais envolvendo biomas e climas.*
composta_por	*Informações espaciais envolvendo biomas e potencial agrícola.*
formada_por	*Informações espaciais envolvendo biomas e relevo.*
ocupada_pelo	*Informações espaciais envolvendo biomas e vegetação.*
pertencente_a	*Informações espaciais envolvendo biomas e unidades da federação.*

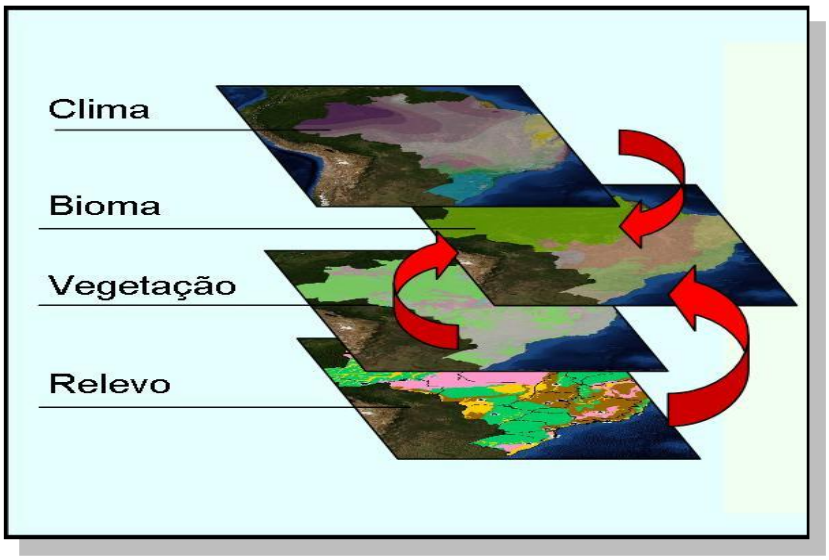


Figura 33 - Mapas temáticos vistos como camadas.

Através da geração de coordenadas geográficas dos centróides dos polígonos foi possível gerar os relacionamentos topológicos entre os mapas murais. Uma vez que o IBGE ainda não possui um identificador único para estes objetos, as mesmas coordenadas foram utilizadas para identificar os objetos geográficos. Para evitar a repetição de sentenças, foi selecionado apenas um tipo de objeto geográfico por bioma.

5.4.2. Projeto da Ontologia

Foram utilizados os mesmos vocabulários do projeto BCIM, com exceção da versão RDF de *ADL Feature Type Thesaurus* (ADL-RDF, 2004) e BuildingsAndPlaces e Topography (Ordnance, 2008) que não foram utilizados. A *ontologia da aplicação* criada no experimento com a BCIM também foi usada como um vocabulário para alinhamento. Várias *datatype properties* destas ontologias foram reutilizadas e algumas *object properties* foram alinhadas como *subproperty* de propriedades destas ontologias. As classes criadas foram alinhadas como *subclass* de classes destas ontologias.

A *ontologia da aplicação* do projeto Mapas Murais, denominada *Bioma*, está descrita em: http://www.inf.puc-rio.br/~hpiccinini/wray/biomapt.html

A Figura 34 mostra uma página da Web com frases referentes aos relacionamentos topológicos criados. Esta página inclui sentenças geradas a partir do *template relacionamento n-ário,* que segue as regras de reificação e construção de frases longas.

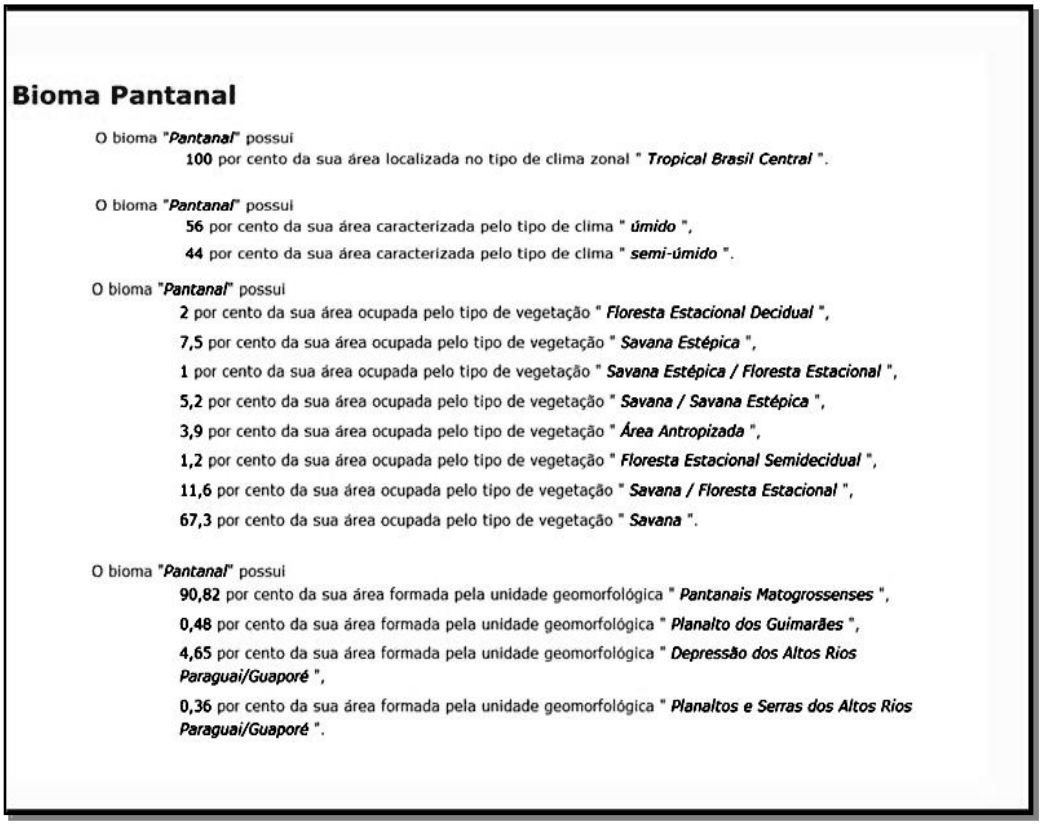


Figura 34 - Sentenças que capturam relacionamentos topológicos conforme publicação feita no site do IBGE.

5.4.3. Publicação do Web Site

No projeto Mapas Murais, a apresentação das páginas Web foi melhorada de forma a torná-las mais amigáveis aos usuários, e seguindo, dentro do possível, os padrões mínimos exigidos para publicações no IBGE.

Foram incluídos manualmente metadados ao final de cada página Web, descrevendo os mapas envolvidos na criação das sentenças. Além disso, passou-se a disponibilizar os conceitos dos termos envolvidos na sentença na mesma página das sentenças e, a partir dos conceitos, foram inseridos hiperlinks para um resumo

da ontologia, conforme pode ser observado na Figura 35. A Figura 36 apresenta os hiperlinks entre as páginas do projeto Mapas Murais.

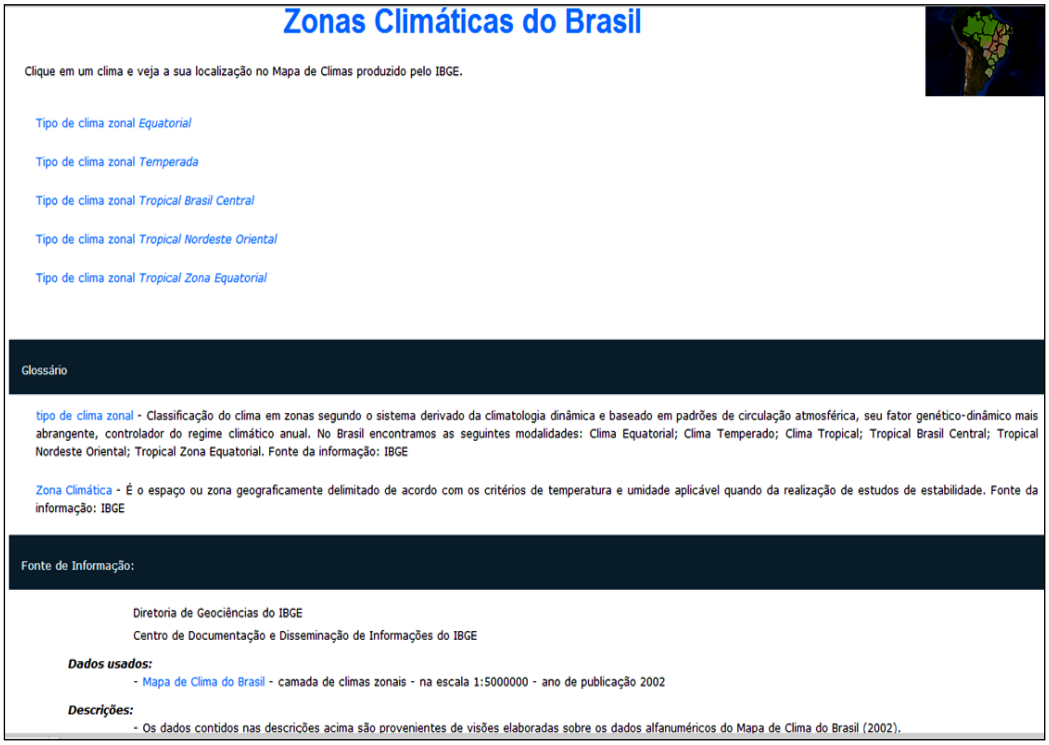


Figura 35 - Conceitos e metadados inseridos ao final da página.

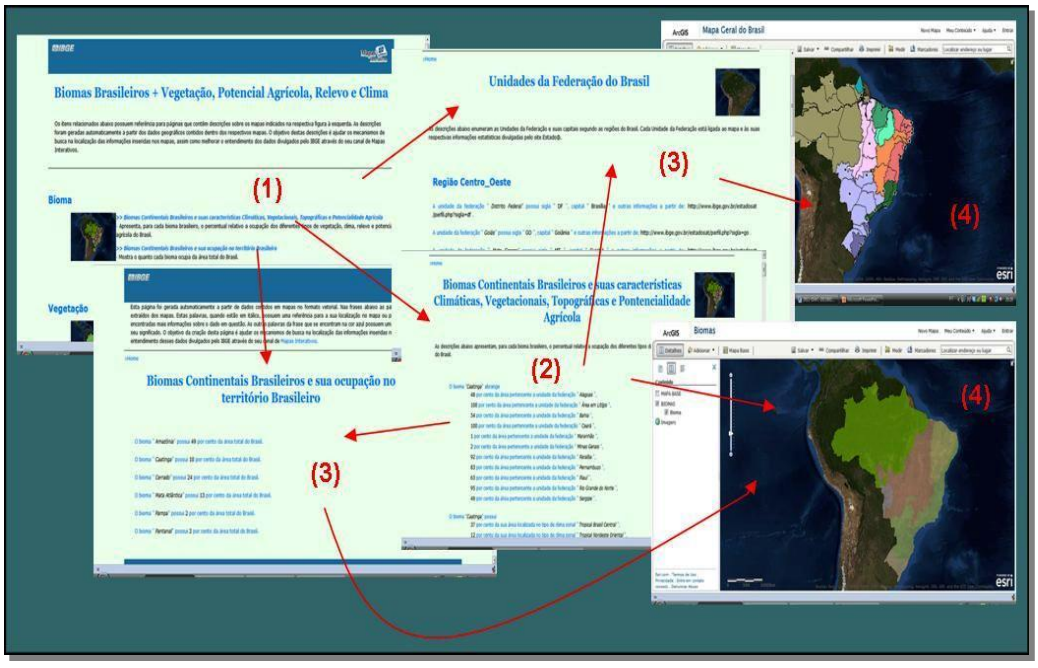


Figura 36 - Organização do site W-Ray para o projeto Mapas Murais

5.5. Projeto Imagens de Satélite

Este projeto envolve imagens de satélite, em formato *raster*, pertencentes ao acervo do Instituto Nacional de Pesquisas Espaciais (INPE), que são divulgadas na Web, através de formulários dinâmicos a partir de http://www.dgi.inpe.br/CDSR/. As imagens de satélite, que anteriormente eram invisíveis para os mecanismos de busca, passaram a ser visíveis, graças às descrições geradas automaticamente, usando a abordagem W-Ray, com o auxílio de um gazetteer.

A motivação para a escolha deste projeto foi a possibilidade de usar a abordagem W-Ray com dados em formato raster.

Neste projeto as sentenças foram geradas em inglês, a partir de março de 2013 e estão disponíveis em: http://tomcat.inf.puc-rio.br:8080/image/

5.5.1. Projeto das Views

Para fornecer visibilidade aos dados em formato *raster*, foi escolhido o *Geonames Gazetteer*, disponível em http://www.geonames.org/. A Tabela 9 mostra as *views* geradas a partir do *gazetteer* e do banco de dados de Imagens de Satélite.

Tabela 9 - *Views* geradas sobre o Geonames Gazetteer e BD de imagens de satélite

Nome da View	Descrição da View
FeatureCategory	Código e o nome das categorias geográficas definidas pelo projeto Geonames.
FeatureCode	Código e nome das classificações dentro de uma categoria geográfica. (Título)
Feature	Características geográficas existentes dentro de uma caixa delimitadora previamente definida pelas latitudes e longitudes que delimitam o *grid* das cenas de satélite que cobrem inteiramente o estado do Rio de Janeiro.
SatelliteImage	Informações não espaciais das imagens de satélite.
LocatedIn	Informações espaciais sobre as características geográficas e as imagens de satélite.

Foram selecionadas apenas oito imagens provenientes de satélites distintos, que cobrem o estado do Rio de Janeiro, ou seja, que se encontram dentro dos limites da caixa delimitadora escolhida. Para cada imagem de satélite escolhida, foram selecionadas todas as características geográficas que estavam dentro do seu quadrante, ou seja, dentro das latitudes e longitudes que delimitam a imagem. Isto

foi possível, visto que cada característica geográfica disponível no *gazetteer* vem acompanhada de sua localização geográfica e cada cena de um satélite é cortada por uma caixa delimitadora que compõe o *grid* do satélite.

Optou-se pela criação de *views* para as categorias e códigos geográficos do *Geonames Gazetteer* e pela definição dessas *views* como títulos e subtítulos que são hierarquizados para que a abordagem W-Ray agrupe automaticamente as sentenças nos respectivos subtítulos, conforme pode ser observado na Figura 37. Esta estratégia evita a repetição de palavras em cada sentença.

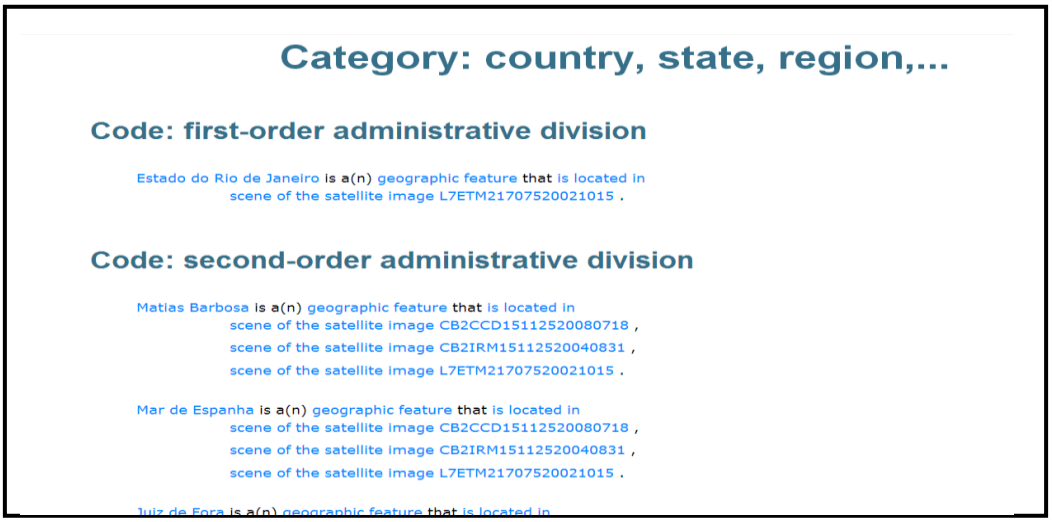


Figura 37 - Sentenças que relacionam as características geográficas com a imagem de satélite para a categoria *country, state, region*.

5.5.2. Projeto da Ontologia

Os mesmos vocabulários do projeto Mapas Murais foram utilizados, com exceção da ontologia BCIM.

Na ontologia de aplicação, foi reutilizada a classe *Feature* da ontologia Geonames.

5.5.3. Publicação do Web Site

O site para as imagens de satélite foi gerado automaticamente através da ferramenta W-RayS, sendo relevante os seguintes aspectos:

– foi gerada uma página inicial que funciona como índice para as outras páginas.

– foi gerada uma página que contém as sentenças, que descrevem as imagens de satélite onde cada sujeito possui um hiperlink para as imagens do catálogo do INPE, a fim de fazer a ligação com a Deep Web.
– foi gerada uma página que contém as sentenças, que descrevem as características geográficas importadas do *Geonames Gazetteer*, onde cada sujeito possui um hiperlink para o serviço de pesquisa e visualização de imagens de satélite, fornecido pela *Geonames* cujos dados também se encontram na *Deep Web.*
– foram geradas oito páginas contendo as sentenças que relacionam as imagens de satélite do INPE com as características geográficas do *Geonames Gazetteer*, onde cada página agrega as sentenças de uma determinada categoria geográfica definida pelo projeto Geonames.

Os metadados e conceitos foram projetados exatamente como no projeto Mapas Murais.

5.6. Comentários finais

Durante a aplicação da abordagem W-Ray a casos reais, observou-se que a maior dificuldade ocorreu na etapa **projeto das *views***. O grau de dificuldade varia de um tipo de dado para outro. No caso dos dados estatísticos, esta etapa exigiu um esforço maior e, mesmo assim, devido à natureza dos dados, as sentenças resultantes apresentaram baixa legibilidade. Por outro lado, no caso dos dados vetoriais, a tarefa de geração dos relacionamentos topológicos, mesmo requerendo um tempo razoável, não apresentou dificuldade. Para os dados em formato raster, a etapa de criação das views também foi bastante simples.

No que se refere à etapa **projeto da ontologia**, a procura e seleção de vocabulários relevantes para os domínios de cada aplicação se mostrou uma tarefa demorada. Vale observar que, se vários projetos são feitos para o mesmo domínio, esta tarefa é dispendiosa apenas no primeiro projeto. Após a carga dos vocabulários na ferramenta W-RayS, a tarefa de alinhamento apoiada pela ferramenta não exige muito tempo do projetista W-Ray.

A etapa de **publicação do web site** é rápida porque é totalmente automatizada requerendo apenas a definição de um pequeno conjunto de parâmetros para a melhoria do site.

Uma inovação implementada a partir de março de 2013 é o desvio automático para o respectivo formulário dinâmico da *Deep Web* sem exibir as páginas geradas pela abordagem W-Ray. Neste procedimento, em tempo de carga das páginas W-Ray, é identificada a URL de onde partiu o acesso e, caso esta corresponda a uma página do Google, faz-se um desvio baseado nas palavras-chave identificadas na pesquisa via Google. Identifica-se a sentença que possui o maior número de palavras-chave pesquisadas e desvia-se automaticamente para o objeto correspondente. Caso haja empate, desvia-se para o objeto da primeira sentença localizada. Deste modo, as páginas W-Ray são visualizadas apenas pelos mecanismos de busca ou por usuários que conhecem o endereço do site do projeto W-Ray. Como esta estratégia não permite a visualização das páginas HTML geradas pela abordagem W-Ray, a preocupação com a legibilidade das sentenças pelos humanos diminui. Entretanto, as páginas continuam visíveis aos mecanismos de busca e, por isso, deve ser mantida a preocupação de se gerar as páginas e sentenças de acordo com as diretrizes definidas pelos mecanismos de busca. Maiores detalhes desta implementação pode ser encontrada no Apêndice D.

O desvio automático foi implementado somente para o mecanismo da Google, embora nada impeça que se implemente também para outros mecanismos de busca. Este procedimento pode ser executado em qualquer browser, mas funciona conforme o esperado apenas para os browsers que usam o protocolo HTTP. No caso de protocolo HTTPS não se tem acesso às palavras-chave via URL gerada pela Google. Neste caso, o desvio é feito para o formulário de acesso ao banco de dados.

5.7. Resumo

Foram descritos alguns dos pontos relevantes relacionados à aplicação da abordagem W-Ray para a geração de um site que descreve dados estatísticos do Banco de Dados Agregados (SIDRA) do IBGE. Foram apresentados dois projetos W-Ray, desenvolvidos utilizando dados do IBGE em formato vetorial. Um deles publica os dados da Base Cartográfica Vetorial Contínua do Brasil ao Milionésimo (BCIM) e o outro publica os dados de Mapas Murais. O quarto projeto W-Ray apresentado, foi desenvolvido para os dados em formato *raster* disponíveis na Web através do catálogo de imagens de satélite do INPE.

6
Análise dos Resultados

Este capítulo resume e analisa os resultados obtidos através dos projetos desenvolvidos utilizando-se a abordagem W-Ray e sua ferramenta W-RayS, que foram apresentados no capítulo 5. São descritos os critérios para avaliação dos projetos e os resultados são discutidos.

6.1. Critérios de avaliação

A abordagem W-Ray foi avaliada segundo três critérios:

1. A abrangência em diferentes tipos de dados;
2. Escalabilidade da ferramenta.
3. O aumento do número de acessos aos dados da Deep Web através dos sites gerados pela ferramenta W-RayS;

Abrangência em diferentes tipos de dados:

Este critério consiste em avaliar se a ferramenta W-RayS foi capaz de aplicar a abordagem W-Ray em diferentes tipos de dados e em diferentes contextos. Os projetos envolveram: dados estatísticos, no projeto *SIDRA* (IBGE); dados em formato vetorial, nos projetos *BCIM e Mapas Murais* (IBGE); dados em formato raster, no projeto *Imagens de Satélite* (INPE).

A mesma ferramenta W-RayS foi utilizada nesses projetos com diferentes tipos de dados sem demandar qualquer tipo de customização. Portanto, considera-se que este critério foi atendido.

Escalabilidade:

Escalabilidade pode ser definida como a capacidade que um programa de computador possui, de continuar a funcionar bem, quando o seu contexto é alterado no tamanho ou no volume, com o objetivo de satisfazer a necessidade do usuário. Tipicamente, a mudança de escala é para um tamanho ou volume maior.

No contexto deste trabalho de tese, a porção de trabalho da ferramenta W-RayS cresce quando o volume de dados do esquema das *views* é alto. Sob este aspecto, a ferramenta W-RayS pode ser considerada escalável se suportar o aumento do volume de dados de entrada, conseguindo manipular estes dados para gerar os resultados esperados (páginas com suas sentenças) de forma uniforme.

Este critério foi avaliado apenas através do projeto SIDRA.

Aumento do acesso à Deep Web:

Para a avaliação deste critério, foi preciso computar não só o número de acessos às páginas geradas pela ferramenta W-RayS, mas também o número de cliques efetuados nos links existentes nas páginas W-Ray que fazem a conexão com a *Deep Web*. Com este objetivo, cada evento *clique* foi capturado através das páginas HTML (evento ONCLICK) e registrado através de um código Javascript, que grava os dados referentes ao evento num arquivo de *log*. A ferramenta *Google Analytics* não foi utilizada para esta avaliação porque este tipo de ferramenta fornece estatística sobre o número de acessos ao site, mas não fornece estatística sobre o número de cliques em cada link dentro do site.

Uma vez obtido o total de cliques nos links para a *Deep Web*, é preciso compará-lo com o total de acessos feitos ao site destino na *Deep Web* para então contabilizar o ganho efetivo. Por este motivo, são necessárias também medidas no site destino. Estas medidas foram obtidas através de arquivos de *log*, existentes para os bancos de dados da *Deep Web*, ou através do número de acessos disponibilizado pelo próprio servidor, como no caso do servidor de mapas ArcGis-Online.

Dos quatro projetos desenvolvidos, apenas dois deles foram avaliados de acordo com este critério: *Mapas Murais* e *SIDRA*. Não foi possível computar o experimento *BCIM* porque o log existente não fornecia o número de acessos à cada camada da *BCIM*. No projeto *Imagens de Satélite* não tivemos acesso ao log do banco de dados. No entanto, como já mencionado anteriormente, estes projetos serviram para avaliar a capacidade de se aplicar a abordagem W-Ray em diferentes tipos de dados.

6.2. Resumo dos experimentos

A seguir apresentamos na Tabela 10 um resumo dos quatro projetos e os resultados do projeto SIDRA e Mapas Murais.

Tabela 10 - Resumo dos experimentos

	SIDRA	BCIM	Mapas Murais	Imagens de Satélite
Critérios avaliados	escalabilidade, volume de acessos e tipo de dado	tipo de dado	volume de acessos e tipo de dado	tipo de dado
Instituição	IBGE	IBGE	IBGE	INPE
Porta principal de entrada para Deep Web	http://www.sidra.ibge.gov.br/	http://mapasinterativos.ibge.gov.br/sigibge/	http://www.arcgis.com/home/search.html?q=ibge&t=content&focus=maps	http://www.dgi.inpe.br/CDSR/
Local inicial de publicação	http://www.sidra.ibge.gov.br/	http://www.inf.puc-rio.br/~hpiccinini/	http://mapasinterativos.ibge.gov.br/	http://tomcat.inf.puc-rio.br:8080/image/
Loca atual de publicação	http://www.sidra.ibge.gov.br/SIDRA_WRAY/default.htm	http://tomcat.inf.puc-rio.br:8080/bcim/	http://tomcat.inf.puc-rio.br:8080/muralmaps/	http://tomcat.inf.puc-rio.br:8080/image/
Período de avaliação	de abril a setembro de 2011	de agosto a janeiro de 2012 e desde março de 2013	de maio a outubro de 2012 e desde março 2013	a partir de março de 2013
Total de *views*	11	9	11	5
Total de sentenças	43.325	7.192	337	21.235
Total de páginas HTML	743	8	12	10
Idioma de publicação	Português	Inglês	Português	Inglês
Tipo de dado Publicado	Estatístico	Geográfico em formato vetorial	Geográfico em formato vetorial	Geográfico em formato raster
Ambiente de publicação	Web Convencional	Web Convencional e Web de dados	Web Convencional e Web de dados	Web Convencional e Web de dados
Vocabulários reutilizados	Glossários: SDMX, OECD, EUROSTAT	Ontologias: Dublin Core, ADL, Wordnet, Geonames, WGS84, BuildingsAndPlaces, SpatialRelations, Topography	Ontologias: Dublin Core, Wordnet, Geonames, WGS84, SpatialRelations, BCIM	Ontologias: Dublin Core, Wordnet, Geonames, WGS84, SpatialRelations,

6.3.Resultados - SIDRA

Neste projeto foi testada a abordagem com dados estatísticos. As páginas geradas ficaram disponíveis, no site oficial do SIDRA/IBGE, para os motores de coleta durante um período de cinco meses (de abril de 2011 a março de 2011).

O site W-Ray para o SIDRA está disponível a partir de: http://www.sidra.ibge.gov.br/SIDRA_WRAY/default.htm

6.3.1. Critério escalabilidade

Através do projeto SIDRA foi possível avaliar a escalibilidade da abordagem W-Ray e da ferramenta W-RayS para um grande volume de dados.

A ferramenta funcionou corretamente, conseguindo sintetizar 43.325 sentenças em 743 páginas geradas a partir de onze *views* construídas sobre o banco de dados SIDRA. Ao todo foram publicadas 45 pesquisas estatísticas, divulgadas pelo IBGE, até março de 2011.

O tempo de processamento foi elevado, mas o desempenho aqui não é a medida principal e sim, o atendimento das tarefas para um grande volume de dados.

6.3.2. Critério aumento do acesso à Deep Web:

A primeira avaliação realizada sobre o aumento no acesso aos dados da *Deep Web,* através do site W-Ray, enfocava o número total de visitantes nas páginas da *Deep Web*. A Figura 38 mostra o total de visitantes provenientes das páginas W-Ray e o total de visitantes provenientes de outras páginas do site SIDRA, durante os cinco meses. Diante destes números, foi feita uma nova avaliação sobre o *log* do BD SIDRA e descobriu-se que o total de visitantes não seria a medida mais importante neste projeto porque, como o sistema SIDRA é muito conhecido, na grande maioria das vezes, seus usuários se conectam diretamente no site ou fazem a busca pelo nome do sistema através de um mecanismo de busca.

Como o objetivo do projeto W-Ray SIDRA era atrair novos usuários, foi feita uma nova avaliação do arquivo *log*, onde o filtro passou a ser o número de

acessos provenientes de mecanismos de busca. As Figuras 39 e 40 mostram os resultados obtidos com este novo enfoque.

A Figura 39 contabiliza o número de acessos às páginas da *Deep Web* provenientes de buscas que não continham a palavra-chave "sidra". O primeiro mês apresenta um elevado número de acessos provenientes dos mecanismos de busca, devido aos testes para verificação do experimento. A partir do segundo mês, pode ser constatado um crescimento real nos acessos realizados através das páginas geradas pela ferramenta W-RayS. Os resultados podem ser considerados satisfatórios uma vez que se conseguiu atrair novos usuários a partir do site W-Ray, mesmo que esse total de acessos à *Deep Web* não tenha alcançado valores elevados.

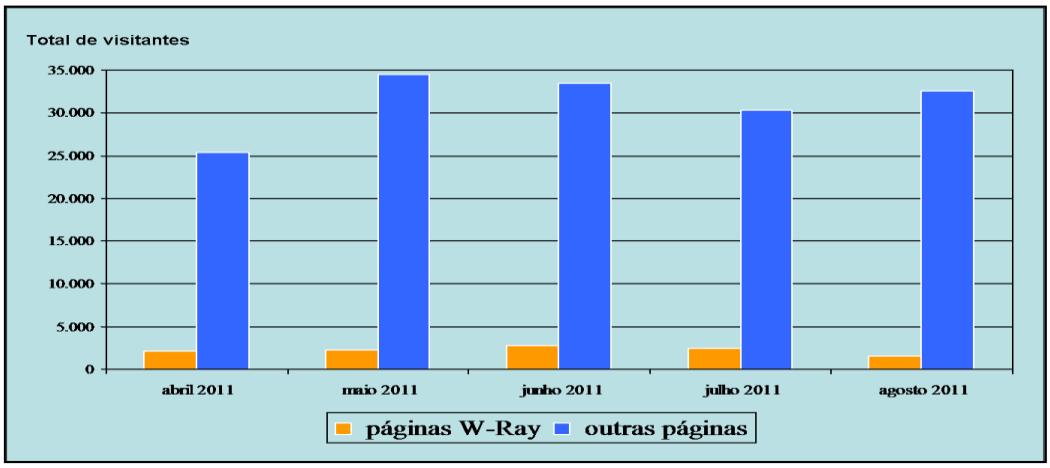


Figura 38 - Total de visitantes provenientes do site W-Ray e total de visitantes provenientes de outras páginas do site SIDRA

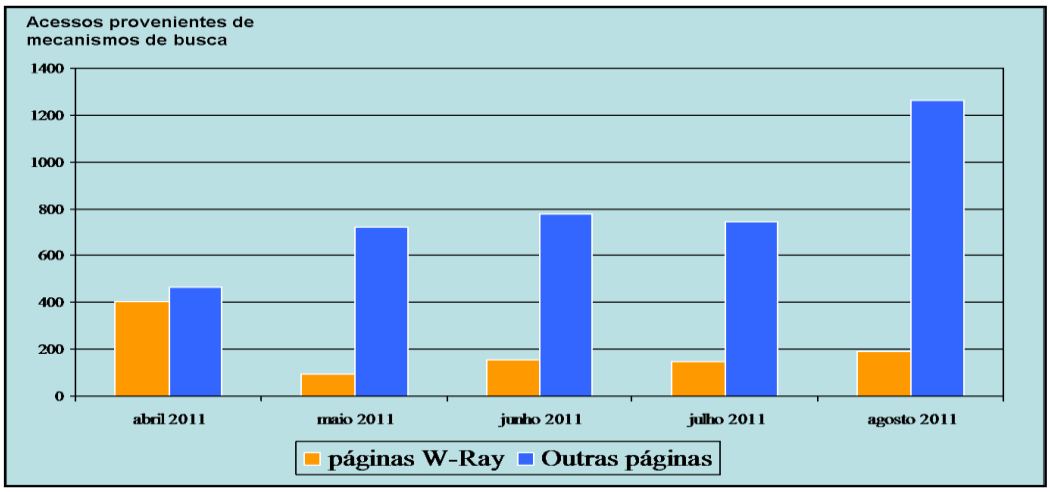


Figura 39 - Número de acessos ao SIDRA provenientes do mecanismos de busca

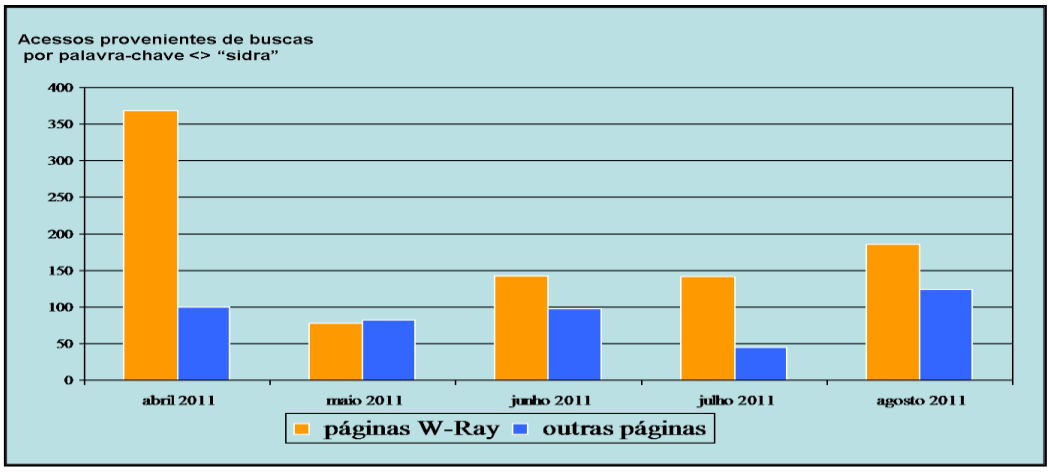


Figura 40 - Número de acessos ao SIDRA provenientes de buscas por palavra-chave diferente de "SIDRA" nos mecanismos de busca

Conforme discutido no capítulo 5, antes da publicação das páginas, o SIDRA não era indexado por qualquer mecanismo de busca porque seu arquivo robots.txt bloqueava a entrada dos rastreadores. Durante os cinco meses que o projeto SIDRA ficou no ar, o arquivo robots.txt foi eliminado. Isso possibilitou que iniciativas de indexação da *Deep Web* (como a da Google) entrassem sistematicamente nos formulários dinâmicos simulando consultas indefinidamente, o que acarretava constantes quedas no servidor do IBGE. No final desse período, o administrador do banco de dados SIDRA voltou a colocar o robots.txt, desta vez incluindo no bloqueio as páginas do W-Ray.

6.4.Resultados - Mapas Murais

No projeto Mapas Murais foi possível avaliar o uso da abordagem W-Ray com dados em formato vetorial. As páginas geradas pela ferramenta W-RayS ficaram disponíveis no site oficial de mapas interativos do IBGE, para os motores de coleta, durante um período de seis meses (de maio de 2012 a novembro de 2012). A partir de março de 2013 as mesmas páginas passaram a ser hospedadas no site da PUC.

Todos os dados da *Deep Web*, ou seja, os Mapas Murais, estão disponíveis no servidor de mapas ArcGis-Online.

O site W-Ray para os Mapas Murais está disponível em: http://tomcat.inf.puc-rio.br:8080/muralmaps/.

6.4.1. Critério aumento do acesso à Deep Web

O número de acessos foi contabilizado através do evento ONCLICK, capturado nas páginas HTML e registrado em um arquivo de *log* por um código JavaScript. Após a contabilização do evento *clique*, os totais de acessos foram comparados com o número total de visualizações divulgados pelo ArcGis-Online para cada mapa.

A seguir a análise dos acessos está divida conforme local de hospedagem das páginas W-Ray.

6.4.1.1. Site W-Ray hospedado no IBGE

As Figuras 41, 42, 43, 44, 45 e 46 apresentam um gráfico de comparação entre o número de acessos provenientes das páginas W-Ray e o de outras páginas para os respectivos mapas: Biomas, Mapa Geral do Brasil, Vegetação, Relevo, Clima e Potencial Agrícola.

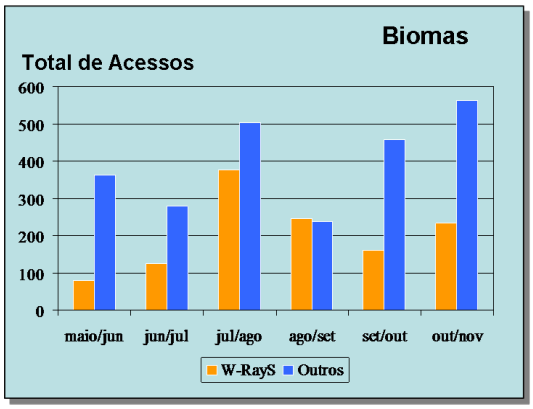


Figura 41 - Resultados Mapa Bioma (2012)

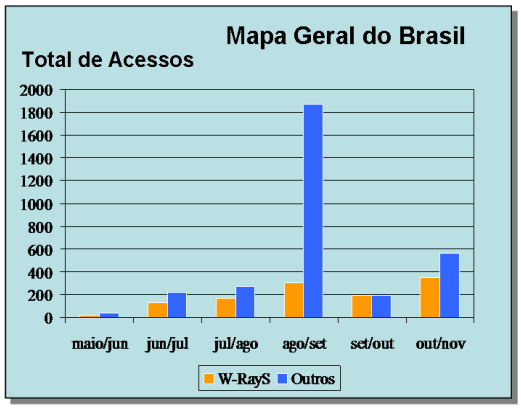
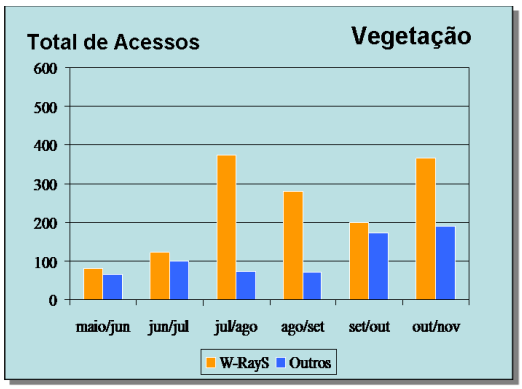


Figura 42 - Resultados Mapa Geral do Brasil (2012)



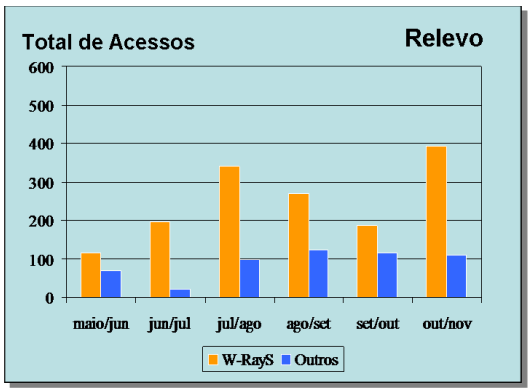


Figura 43 - Resultados Mapa Vegetação (2012)

Figura 44 - Resultados Mapa relevo (2012)

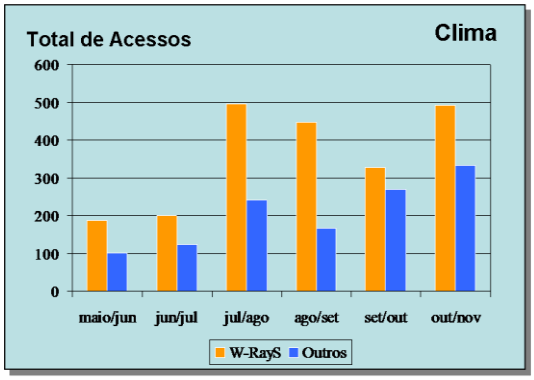


Figura 45 - Resultados Mapa Clima (2012)

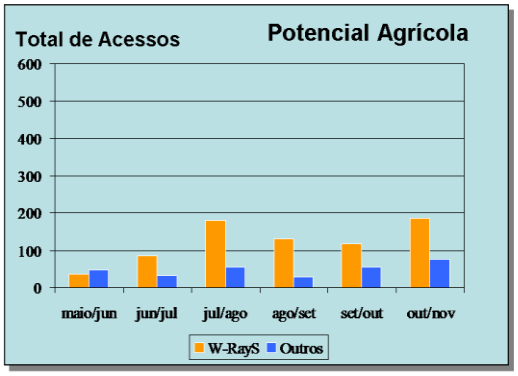


Figura 46 - Resultados Mapa Potencial Agrícola (2012)

A intenção do projeto Mapas Murais era despertar o interesse dos usuários para outros mapas murais, usando como atrativo o mapa de Biomas, que na ocasião, era o mapa mais acessado. O mapa de Biomas e o Mapa Geral do Brasil continuaram tendo a maioria dos acessos provenientes de outras páginas que não as geradas pela ferramenta W-RayS. No entanto, obteve-se sucesso com os mapas de Vegetação, Relevo, Clima e Potencial Agrícola. Em alguns meses, como julho/agosto de 2012, o número de acessos provenientes do site W-Ray contabiliza, para alguns mapas, o dobro ou mais, dos acessos provenientes de outras páginas.

O Mapa Geral do Brasil apresentou um comportamento diferente entre os meses de agosto e setembro de 2012. Neste período, o número de acessos através de outras páginas cresceu de forma vertiginosa. Talvez a procura tenha aumentado devido ao início do segundo período letivo do ano de 2012, principalmente se o endereço do mapa tiver sido divulgado pelos professores para os seus alunos.

Embora os resultados tenham sido bons, é preciso observar que o número de acessos contabilizados para o projeto Mapas Murais W-Ray envolveu não só aqueles originados de um mecanismo de busca, mas também de acessos feitos por usuários diretamente às páginas W-Ray via site do IBGE. O *log* das páginas W-Ray, na época do experimento, não discriminava a origem do acesso.

6.4.1.2. Site W-Ray hospedado na PUC

No início de novembro as páginas W-Ray foram retiradas do site do IBGE e, a partir de março de 2013, passaram a ser hospedadas na PUC. Durante o experimento na PUC, passou-se a capturar as pesquisas por palavra-chave feitas via Google para o redirecionamento automático ao mapa, sem que o usuário visualize as páginas do site W-Ray. Os totais de acesso deste período podem ser conferidos nas Figuras 47, 48, 49, 50, 51 e 52 respectivamente para os mapas Biomas, Mapa Geral do Brasil, Vegetação, Relevo, Clima e Potencial Agrícola.

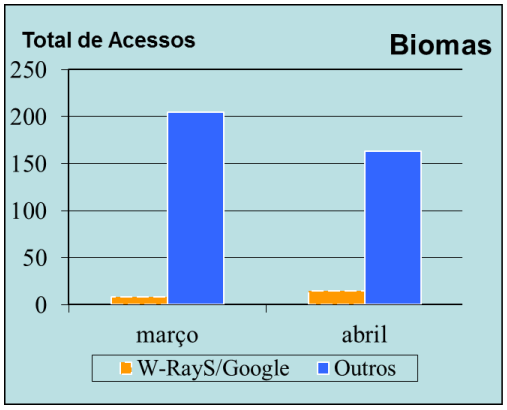


Figura 47 – Resultado Mapa Bioma (2013)

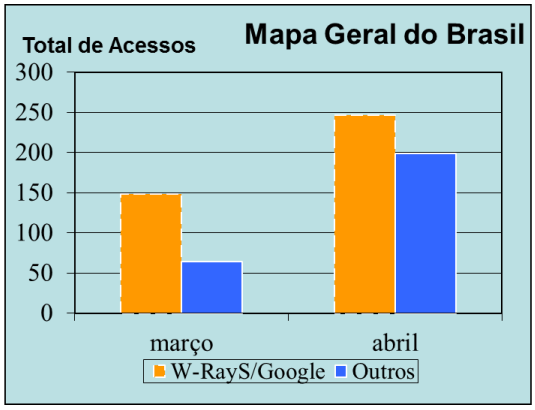


Figura 48 - Resultado Mapa Geral do Brasil (2013)

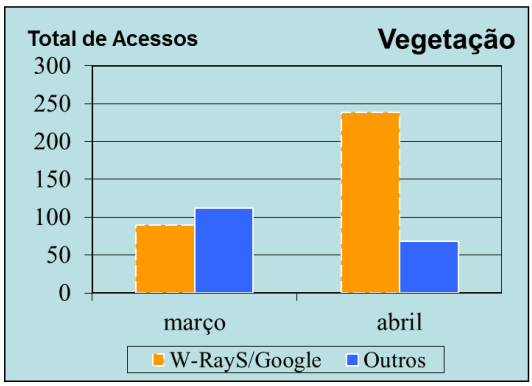


Figura 49 – Resultado Mapa de Vegetação (2013)

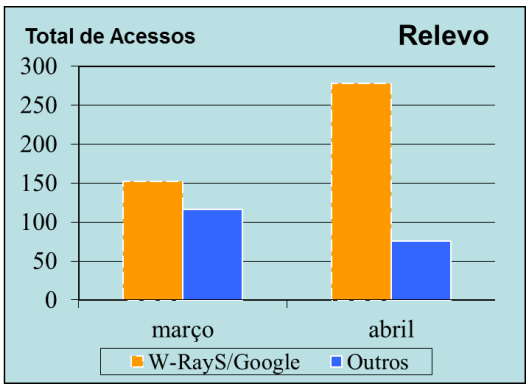


Figura 50- Resultado Mapa de Relevo (2013)

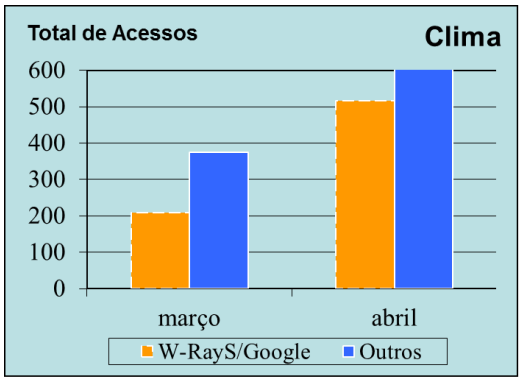


Figura 51– Resultado Mapa de Clima (2013)

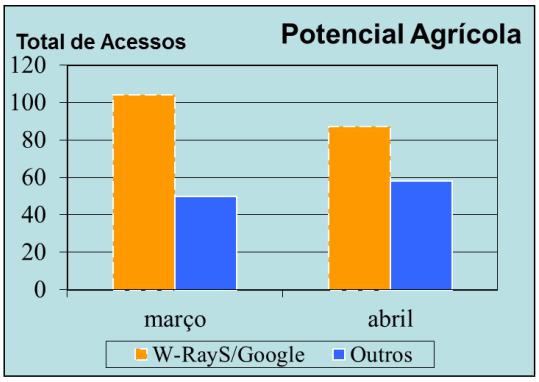


Figura 52- Resultado Mapa Potencial Agrícola (2013)

Nas figuras acima, as barras amarelas correspondem exclusivamente a acessos via Google, enquanto que as barras azuis correspondem a acessos via páginas do IBGE sem envolver mecanismos de busca. Apesar do pouco tempo de avaliação, os resultados apontam para um incremento no número de acessos.

6.5.Resumo

Foram descritos os critérios para a avalição dos projetos W-Ray SIDRA e Mapas Murais apresentados no capítulo 5. No caso SIDRA, mesmo que o total de acessos à *Deep Web* não tenha alcançado valores elevados, os resultados com o experimento SIDRA/IBGE se mostraram satisfatórios porque novos usuários foram atraídos a partir do site W-Ray. Também foi avaliada a escalabilidade da abordagem W-Ray e da ferramenta W-RayS para o elevado volume de dados do SIDRA.

O estudo de caso com Mapas Murais apresentou bons resultados no que se refere ao quesito *aumento de acesso à Deep Web*, onde se conseguiu atrair usuários através das páginas do site W-Ray.

7
Conclusões

"As coisas tangíveis
tornam-se insensíveis
à palma da mão.

Mas as coisas findas
muito mais que lindas,
essas ficarão."

trecho do poema *Memória* de Carlos Drummond de Andrade (Claro Enigma 1951)

Este trabalho tratou do problema de tornar visíveis dados da Deep Web, sob uma nova perspectiva, que retira dos mecanismos de busca a responsabilidade de prover visibilidade aos dados, levando-a para os administradores de bancos de dados. A abordagem proposta, denominada W-Ray, juntamente com sua ferramenta W-RayS se mostraram promissoras diante dos resultados obtidos. Inicialmente neste capítulo são descritas as contribuições alcançadas por esta pesquisa, a seguir discute-se as limitações da proposta e por fim, novas direções para trabalhos futuros são apresentadas.

7.1. Contribuições

Neste trabalho, foram apresentadas a abordagem W-Ray e uma ferramenta denominada W-RayS que suporta esta abordagem. A principal contribuição deste trabalho é a proposta de uma abordagem sistemática capaz de tornar visíveis na superfície da Web convencional, e na Web de dados diferentes tipos de dados ocultos na *Deep Web*.

A abordagem proposta consiste em publicar uma parte significativa dos dados armazenados em um banco através de sentenças em LN, organizadas em páginas estáticas com RDFa embutido a fim de atraírem os mecanismos de busca,

sem perder a semântica dos dados estruturados. Desta forma, um banco de dados passa a ter um Web site que o reflete e que serve de ponto de entrada para a riqueza dos dados armazenados. A abordagem também suporta a publicação de diferentes tipos de dados. Além dos dados convencionais, são contemplados os provenientes de um *data Warehouse* e os Geográficos em formato raster ou vetorial. Por último, W-Ray é capaz de proporcionar visibilidade dos dados tanto na superfície da Web convencional quanto na Web de dados.

Quatro casos reais foram desenvolvidos utilizando a abordagem W-Ray e sua ferramenta W-RayS, para gerar páginas HTML estáticas na Web. Os casos envolvem três diferentes tipos de dados: dados em formato vetorial, armazenados em servidores de mapas de grande porte; dados estatísticos, armazenados em *data warehouse* de grande porte; e imagens de satélite, armazenadas em bancos de dados.

A abordagem W-Ray atribui a responsabilidade de decidir quais dados devem ser expostos na Web e como eles devem ser publicados, para a figura do administrador do banco de dados, aliviando assim, as inúmeras tentativas de preenchimento de formulários HTML pelos motores de coleta de dados. Tal sondagem gera uma enorme quantidade de acessos que podem levar, por razões de segurança e desempenho, a um completo bloqueio do acesso dos motores de coleta aos formulários, que são a porta de entrada para a *Deep Web*. Essa mudança de paradigma representa a maior diferença em relação a abordagem *Surfacing*, recentemente proposta para o problema da *Deep Web*.

Mais detalhadamente as contribuições desta pesquisa são:

1. **Controle sobre a exposição dos dados**: a etapa de *Projeto das Views* encapsula o núcleo do problema de decidir quais dados devem ser publicados e como descrever dados armazenados em bancos de dados da *Deep Web*. No caso de dados geográficos este problema fica encapsulado através da definição de *views* com a ajuda de consultas espaciais que retornam dados alfanuméricos. Esta solução fornece a segurança de que apenas os dados definidos pelo dono do dado estarão visíveis;
2. **Reutilização e alinhamento de ontologias**: a etapa de concepção da Ontologia é apoiada por módulos da ferramenta W-RayS que facilitam a reutilização e alinhamento de ontologias, incorporando as recomendações de dados ligados. A ferramenta fornece também a opção de executar um

alinhamento mais simples entre ontologias sem exigir do projetista conhecimentos relativos à Web Semântica.

3. **Publicação e Triplificação baseados na mesma ontologia**: o estágio de Publicação do site W-Ray e Triplificação são ambos baseados no mesmo projeto da ontologia.
4. **Utilização proveitosa da tecnologia dos mecanismos de busca tradicionais**: o estágio de publicação do Web site resulta em páginas HTML estáticas, que são facilmente indexadas pelos motores de coleta de dados. Desta forma, os usuários passam a localizar dados da *Deep Web* através do mecanismo de busca de sua preferência.
5. **Geração automática de RDFa:** o RDFa embutido nas páginas Web permite que o mesmo projeto W-Ray seja publicado na Web convencional e na Web de dados. Com o RDFa embutido nas sentenças, a estrutura dos dados fica preservada o que permite consultas mais específicas por motores de busca ou agentes de software, capazes de analisar páginas Web com RDFa. Na abordagem *Surfacing* da Google, a estrutura dos dados é perdida.
6. **Suporte a dados em grande escala e a diferentes tipos de dados:** Os resultados apresentados mostram a capacidade de reprodução da abordagem para um grande volume de dados e em tipos de dados diferentes.
7. **Resultados positivos**: os resultados obtidos nos estudos de caso com dados reais permitem concluir que a abordagem é promissora. Foi obtido um grande aumento no número de acessos aos dados da *Deep Web* através dos sites W-Ray.

7.2. Limitações

As limitações do trabalho são inerentes à tecnologia de banco de dados. A maior carga de trabalho da abordagem W-Ray se encontra na definição das *views* materializadas dos dados que serão publicados. No entanto, o mesmo tipo de trabalho é requerido nas abordagens de *Busca por Produto* e nas ferramentas de mapeamento RDB-to-RDF. No caso de uma ferramenta RDB-to-RDF, para que apenas dados úteis e de domínio público sejam publicados na Web de dados, não basta triplificar o BD na sua íntegra, mas sim selecionar dados significativos, o

que requer um projeto de *views* sobre os dados ou alteração na linguagem de mapeamento.

Outra limitação está relacionada com as atualizações nos bancos de dados que podem requerer que as páginas W-Ray sejam regeradas. Se a atualização é apenas no dado, como o projeto W-Ray é persistente, basta que as *views* sejam regeradas e em seguida seja reexecutada a geração de paginas HTML. Este passo não requer envolvimento de qualquer especialista. Por outro lado, se o modelo conceitual do banco de dados for alterado, pode ser que o projeto W-Ray tenha que ser revisto.

No que se refere à geração de sentenças, uma limitação também inerente ao modelo do banco de dados é quando a sentença é gerada a partir de um *template de relacionamento n-ário* e o sujeito não é definido explicitamente pelo usuário. Neste caso, não é possível decidir automaticamente qual classe funcionará como sujeito da sentença e, se o sujeito não é definido explicitamente pelo usuário, serão geradas tantas sentenças quanto for o valor de *n* no relacionamento *n-ário*. Em cada sentença, uma classe diferente pertencente ao relacionamento *n-ário* funciona como sujeito. Se o usuário não definir o sujeito, além da geração de sentenças repetidas e sem valor para o entendimento humano, o número de sentenças pode aumentar muito.

Uma limitação da abordagem é a geração de uma grande quantidade de páginas HTML com descrições textuais extensas que podem tornar o site W-Ray confuso para o usuário.

Vale observar que, se a estratégia de desvio automático descrita nos comentários finais do capítulo 5 for implementada, os problemas descritos nos dois últimos parágrafos não são relevantes porque a visibilidade das páginas W-Ray será destinada apenas aos *crawlers* e não mais ao usuário.

Outra limitação da abordagem está relacionada aos nomes das tabelas, nomes de relacionamentos e nomes de atributos. Se o projetista W-Ray optar por não utilizar os módulos de projeto de templates oferecidos pela ferramenta W-RayS para a otimização destes nomes, então no momento da criação das visões ele deve se preocupar em definir nomes mais legíveis e significativos para estes elementos.

7.3. Trabalhos Futuros

Como trabalhos futuros é possível identificar:

- A possibilidade de implementação de um módulo de Projeto de Views que poderá sugerir definições de *views* para o administrador de dados, em uma análise com base na engenharia reversa do esquema do banco de dados.
- Publicar os dados da *Deep Web* em tabelas HTML e comparar os resultados com os dados publicados através de sentenças. Quando este trabalho foi iniciado nenhum mecanismo de busca indexava tabelas HTML. Atualmente, alguns mecanismos de busca ainda consideram tabelas como objetos visuais e os que conseguem indexá-las ainda não resolvem todos os problemas que envolvem este tipo de objeto HTML (Veneti et al., 2011; Madhavan et al., 2009; Cafarella et al., 2008).
- Automatizar ainda mais o processo de geração de linguagem natural, sem perder a legibilidade das sentenças.
- Avaliar qualitativamente o grau de compreensão humana e as dificuldades encontradas nos layouts das páginas Web geradas em nossos estudos de caso.
- Adequar as sentenças com vistas à verbalização de descrições de mapas voltados para deficientes visuais, tais como: mapas para a exploração e localização de pontos de interesse; mapas para orientação e movimento; mapas para fins educacionais.

Referências Bibliográficas

Adida, B., Herman, I., Sporny, M., Birbeck, M. (eds) (2012) **RDFa 1.1 Primer - Rich Structured Data Markup for Web Documents**. W3C Working Group Note, 07 June 2012. Available at: http://www.w3.org/TR/rdfa-primer/

ADL-RDF (2004**) RDF version of ADL Feature Type Thesaurus**. Available at: http://www.alexandria.ucsb.edu/gazetteer/FeatureTypes/ADL-all.rdf

ArcGis (2012) **ArcGis Online**. Available at: http://www.esri.com/software/arcgis/arcgisonline

Arenas, M., Prud'hommeaux, E., Sequeda, J. (2011) **A Direct Mapping of Relational Data to RDF**. W3C Working Draft.

Auer, S., Dietzold, S., Lehmann, J., Hellmann, S., Aumueller, D. (2009) **Triplify: light-weight linked data publication from relational databases**. Proc. WWW 2009, pp. 621–630.

Beckett, D. (2004) **RDF/XML Syntax Specification (Revised) - W3C Recommendation Retrieved 2013**. Available at: http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/

Beckett, D., Berners-Lee, T. (2011) **Turtle - Terse RDF Triple Language - W3C Team Submission. Retrieved 2013**. Available at: http://www.w3.org/TeamSubmission/turtle/

Bergman, M.K. (2001) **The Deep Web: Surfacing Hidden Value**. *J. Electr. Publ.*7.

Berners-Lee, T. (2006). **Linked Data - Design Issues**. Available at: http://www.w3.org/

Bizer, C., Heath, T., Berners-Lee, T. (2009**) Linked Data - The Story So Far**. *Int. Journal on Semantic Web and Information Systems* 5(3), pp. 1–22.

Bizer, C., Seaborne, A. (2004) **D2RQ – Treating Non-RDF Databases as Virtual RDF Graphs Endpoints**. Poster at the 3rd Int'l. Semantic Web Conf.

Breitman, K. (2005) - **Web Semântica - A Internet do Futuro**. LTC - Livros Técnicos e Científicos Editora S.A. - Rio de Janeiro - pp. 20 – ISBN 85-216-1466-7

Brickley, D., Guha, R.V. (2004) **RDF Vocabulary Description Language 1.0: RDF Schema**. Available at: http://www.w3.org/TR/rdf-schema/

Cafarella, M.J., Halevy, A.Y., Khoussainova, N. (2009) **Data integration for the relational web**. Proc. VLDB Endow. 2(1), pp. 1090–1101.

Cafarella, M.J., Halevy, A.Y., Madhavan, J. (2011) **Structured Data on the Web**. *Comm. of the ACM* 54(2), pp. 72-79.

Cafarella, M. J., Halevy, A., Zhang, Y., Wang, D. Z., and Wu, E.. (2008) **WebTables: Exploring the Power of Tables on the Web**. In VLDB, 2008

Caldwell, B., Cooper, M., Reid, L.G., Vanderheiden, G. (2008) **Web Content Accessibility Guidelines (WCAG) 2.0**. W3C Recommendation.

Callan, J. (2002) **Distributed information retrieval. In: Advances in Information Retrieval**. The Information Retrieval Series 7, Springer, USA, pp. 127–150.

Castillo, C., 2004. **Effective Web Crawling. PhD**. Thesis in Computer Science - University of Chile

Cerbah, F. (2008) **Learning highly structured semantic repositories from relational databases**. The Semantic Web: Research and Applications, pp. 777-781.

Das, S., Sundara, S., Cyganiak, R. (2012) **R2RML: RDB to RDF Mapping Language**. W3C Working Draft. Available at: http://www.w3.org/TR/r2rml/

Dragut, E. C., Meng, W., Yu, C. T.. (2012**) Deep Web Query Interface Understanding and Integration**. Editora Morgan & Claypool Publishers, pp. 1.1 3

Egenhofer, M.; Franzosa, R. (1991) **Point-Set Topological Spatial Relations**. International Journal of Geographical Information Systems, v. 5, n.2, p.161-174.

Eisenberg, V., Kanza, Y. (2012) **D2RQ/Update: Updating Relational Data via Virtual RDF**. Poster apresentado em World Wid Web 2012 – Lyon - França.

EUROSTAT (2012) **Glossary of Statistical Terms**. Available at: http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Glossary:Eurostat

Figueredo, L.A., Masello, J. (2005) **SIDRA - Aggregate Database – Definition and Loading**. Technical Note, Diretoria de Informática, IBGE, Rio de Janeiro, Brazil.

Fliedl, G., Kop, C., Vöhringer, J. (2010) **Guideline based evaluation and verbalization of OWL class and property labels**. Data & Knowledge Eng. 69(4), pp. 331-342.

Fuchs, N.E., Kaljurand, K., Kuhn, T. (2008) **Attempto Controlled English for Knowledge Representation**. Proc. Reasoning Web 2008, LNCS 5224, Springer, pp. 104–124.

Furtado A., Barbosa L. S. D. J., Casanova M. A., Piccinini H. (2010) **First version of a Prototype for Publishing Deep Web Data**. Tech Rep. 11/10. Dept. Informatics, PUC-Rio.

GeoNames (2012) **GeoNames Gazetteer**. Available at: http://www.geonames.org/

Google-OptimizationGuide (2012) **Google's Search Engine Optimization Starter Guide**.

Google-WebmasterBlog (2009) **Webmaster Blog**. Available at: http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html

Google-Shopping (2013) **Google Merchant Center - Especificação de Feed de produtos**. Available at: http://support.google.com/merchants/bin/answer.py?hl=pt-BR&answer=188494#BR

Ghawi, R., Cullot, N. (2007**) Database-to-ontology mapping generation for semantic interoperability**. Proc. Third International Workshop on Database Interoperability, InterDB.

He, H., Meng, W., Yu, C.T., Wu, Z. (2005) **Metaquerier: querying structured web sources on-the-fly**. Proc. of SIGMOD Conf., pp. 927–929.

He, H., Meng, W., Yu, C.T., Wu, Z. (2005) **Wise-integrator: A system for extracting and integrating complex web search interfaces of the Deep web**. Proc. 31st Int'l. Conf. on Very Large Data Bases, pp. 1314–1317.

Hewlett, D., Kalyanpur, A., Halaschek-Wiener, C., Kolovski, V. (2005) **Effective NL Paraphrasing of Ontologies on the Semantic Web**. Proc. Workshop on End-User Semantic Web Interaction I 4th Int'l. Semantic Web Conf.

Heath, T., Bizer, C. (2011) **Linked data: Evolving the Web into a Global Data Space**. Morgan & Claypool Publishers and available in http://linkeddatabook.com/editions/1.0/

Hollink, L., Schreiber, G., Wielemaker, J., Wieling, A. (2003) **Semantic Annotation of Image Collections**. Proc. Knowledge Markup and Semantic Annotation Workshop, Sanibel, Florida, USA.

HTML-Validator (2012) **W3C HTML Validator version 1.3**. Available at: http://validator.w3.org/

Kabisch, T., Dragut, E.C., Leser, U. (2010) **Deep Web Integration with VisQI**. Proc. VLDB Endow. 3(2).

Konstantinos, N., Vavliakis, Theofanis, K., Grollios, Pericles, A., Mitkas (2010) **RDOTE - Transforming Relational Databases into Semantic Web Data**. The International Semantic Web Conference (ISWC).

Lacy, L. W. (2005) **OWL: Representing Information Using The Web Ontology Language**. Trafford Publishing, Victoria, Canada. pp- 133 - ISBN 1-4120-3448-5

Leme, L.A., Brauner, D.F., Casanova, M.A., Breitman, K. (2007) **A Software Architecture for Automated Geographic Metadata Annotation Generation**. Proc. First Brazilian e-Science Workshop.

Machado, F. N. R. (2000) **Projeto de Data Warehouse – Uma visão Multidimensional**. Editora Érica.

Madhavan, J., Shawn, J. R., Cohen, S., Dong, X., Ko, D., Yu, C., Halevy, A. (2007) **Web-scale Data Integration: You can only afford to Pay As You Go**. Proceedings of the Conference on Innovative Data Systems Research.

Madhavan, J., Ko, D., Kot, L., Ganapathy, V., Rasmussen, A., Halevy, A.Y. (2008) **Google's Deep-Web Crawl**. Proc. VLDB Endow. 1(2), pp. 1241–1252.

Madhavan, J., Afanasiev, L., Antova, L., Halevy, A.Y. (2009) **Harnessing the Deep Web: Present and Future**. Proc. 4th Biennial Conference on Innovative Data Systems Research (CIDR), Asilomar, California, USA (Jan. 4-7).

Manola, F., Miller, E. (2004) - **RDF Primer - W3C Recommendation**. Available at: http://www.w3.org/TR/rdf-primer/

MapServer (2011**) MapServer open source web mapping**. Available at: http://mapserver.org/about.html#about

Maiti, A., Dasgupta, A., Zhang, N., Das, G. (2009) **HDSampler: Revealing Data Behind Web Form Interfaces**. Proc. 35th SIGMOD Conf., pp. 1131-1134.

Matthiessen, C.M.I; Bateman, J.A. (1991). **Text Generation and Systemic-Functional Linguistics**. Pinter Publishers, London.

McGuinness, D. L., Harmelen, F. (2004) **OWL - Web Ontology Language Overview**. Available at: http://www.w3.org/TR/owl-features/

Mobissimo (2004) Available at: http://www.mobissimo.com/search_airfare.php

Nguyen, T., Nguyen, H., Freire, J. (2010) **PruSM: A Prudent Schema Matching Approach for Web Forms**. Proc. Conf. on Information and Knowledge Management (CIKM), pp. 1385-1388.

Noy, N., Rector, A. (2006) **Defining N-ary Relations on the Semantic Web**. W3C Working Group Note.

Nunes, B. P., Mera, A., Casanova, M. A., & Kawase, R. (2012) **Automatically generating multilingual, semantically enhanced, descriptions of digital audio and video objects on the Web**. Advances in Knowledge-Based and Intelligent Information and Engineering Systems. M. Grana et al. (Eds.) - IOS Press, 2012, pp. 575 - 584

OECD (2007) **Glossary of Statistical Terms**. Available at: http://stats.oecd.org/glossary

Ordnance (2008) **Ordnance Survey Ontology**. Available at: http://www.ordnancesurvey.co.uk/oswebsite/ontology/

Pant, G., Srinivasan, P., e Menczer, F.. (2004) **Capítulo sob o título: Crawling the Web**. Web Dynamics: Adapting to Change in Content, Size, Topology and Use de Levene M., Poulovassilis A.. Editora Springer. pp 153.

Piccinini, H., Lemos, M., Casanova, M. A., Furtado, A.L. (2010a) **W-Ray: A Strategy to Publish Deep Web Geographic Data**. Proc. Int'l. Conf. on Adv.

in Concep. Modeling: Applic. and Challenges Workshops, LNCS 6413, Springer, Heidelberg, pp. 2-11.

Piccinini, H.; Lemos, M.; Casanova, M.A.; Furtado, A.L. (2010b) **W-Ray: A Strategy to Publish Deep Web Geographic Data**. Tech Rep. 10/10. Dept. Informatics, PUC-Rio

Prud'hommeaux, E., Hausenblas, M. (2010) **Use cases and requirements for mapping relational data bases to rdf**. Available at: http://www.w3.org/TR/rdb2rdf-ucr/

Queiroz, G. R.; Ferreira, K. R. (2006) **Tutorial sobre Banco de Dados Geográficos**. GeoBrasil 2006. Instituto Nacional de Pesquisas Espaciais.

Raghavan, S., Garcia-Molina, H. (2001) **Crawling the Hidden Web**. Proc. 27th Int'l. Conf. on Very Large Data Bases (VLDB '01), pp. 129–138

Rajaraman, A. (2009) **Kosmix: High Performance Topic Exploration using the Deep Web**. Proc. VLDB Endow. 2(2), pp. 1524-1529.

RDFa-Validator (2012) **W3C RDFa Validator**. Available at: http://www.w3.org/2012/pyRdfa/Validator.html

RDFa-Destiller (2012) **W3C RDFa Destiller and Parser**. Available at: http://www.w3.org/2007/08/pyRdfa/

Sahoo, S.S., Halb, W., Hellmann, S., Idehen, K., Thibodeau Jr, T., Auer, S., Sequeda, J., Ezzat, A. (2009) **A survey of current approaches for mapping of relational databases to rdf**. W3C RDB2RDF Incubator Group report.

SDMX (2005) **SDMX Standards Version 2.0**. Complete Package available at: http://sdmx.org/?page_id=16#package

Shestakov, Denis (2008). **Search Interfaces on the Web: Querying and Characterizing**. TUCS Doctoral Dissertations 104, University of Turku - Finland

Sauermann, L., Cyganiak, R. (2008) **Cool URIs for the Semantic Web**. Available at: http://www.w3.org/TR/cooluris/#r303gendocument

SUMO (2009) **SUMO - Suggested Upper Merged Ontology**. Available at: http://www.ontologyportal.org/

Veneti, P., Halevy, A., Madhavan, J., Pasca, M. (2011) **Recovering Semantics of Tables on the Web**. In VLDB, 2011.

Yahoo! (2008) **Yahoo! Developer Network**. Available at: http://developer.yahoo.com/blogs/ydn/posts/2008/09/search monkey_ support_for_rdfa_enabled/

WordNet (2009) - **WordNet Ontology**. Available at: http://www.ontologyportal.org/

W3CGeo (2005) **W3C Geo Ontologies**. Available at: http://www.w3.org/2005/Incubator/geo/XGR-geo-ont-20071023/

Apêndice A: Conceitos

1. RDF

O RDF é o arcabouço sobre o qual as ontologias na Web Semântica são baseadas. De acordo com Manola & Miller (2004), o RDF é uma linguagem para representar informações sobre os recursos da World Wide Web. Mais especificamente, segundo Breitman (2005) é uma linguagem declarativa que fornece uma maneira padronizada de utilizar o XML para representar os metadados no formato de declarações sobre propriedades e relacionamentos entre itens na Web. Esses itens, denominados recursos, podem ser virtualmente qualquer objeto (texto, figura, vídeo e outros) desde que possuam um endereço na Web.

O objetivo do RDF é fornecer um padrão para descrever a informação na Web para que ela possa ser trocada entre as aplicações de software (Manola & Miller, 2004). Como através do RDF a informação é descrita de forma padronizada, os projetistas de software podem aproveitar a disponibilidade de analisadores e ferramentas de processamento de RDF. A capacidade de trocar informações entre diferentes aplicações significa que as informações podem ser disponibilizadas para fins diferentes daqueles para os quais elas foram originalmente criadas.

A linguagem RDF baseia-se na ideia de fazer declarações sobre os recursos na forma de *sujeito-predicado-objeto (S-P-O)*. Estas expressões são conhecidas como *triplas RDF (S-P-O)*, onde o sujeito identifica o recurso, o predicado é a parte que identifica a propriedade ou característica do sujeito e o objeto é o valor da propriedade. Para a identificação dos recursos são utilizados identificadores da Web (***Uniform Resource Identiers - URIs***). Isso permite que as declarações sobre recursos em RDF possam ser representadas como um grafo onde os nós e arcos representam os recursos, as suas propriedades e seus valores. Desta forma, os nós do grafo representam os **indivíduos** que podem ser o sujeito ou o objeto de uma tripla RDF e os arcos representam as **propriedades** que ligam sujeito ao valor da

propriedade. O exemplo a seguir, transcrito de Manola & Miller (2004), torna estes conceitos mais claros. Considere a seguinte declaração:

Existe uma **pessoa** que é identificada pela URI "*http://www.w3.org/People/EM/contact#me*", cujo **nome** é "*Eric Miller*", cujo endereço de **e-mail** "*em@w3.org*", e possui o **título** de "*Dr.*". A Figura 53 mostra como este exemplo pode ser representado como um grafo de RDF.

Figura 53 - Exemplo de um grafo RDF (Manola & Miller, 2004)

Na Figura 53 podemos identificar:

- Indivíduos, como por exemplo, Eric Miller, identificado por *http://www.w3.org/People/EM/contact#me*
- Tipos de recursos, por exemplo, **Pessoa**, identificado por *http://www.w3.org/2000/10/swap/pim/contact#Pessoa*
- Propriedades desses recursos, por exemplo, **email**, identificado por *http://www.w3.org/2000/10/swap/pim/contact#mailbox*
- Valores de propriedades, como por exemplo: ***mailto:em@w3.org*** que é o valor da propriedade **mailbox**. O RDF também usa tipos de dados como

string, números inteiros e datas como valores das propriedades, como por exemplo: *"Eric Miller"* e *"Dr."*.

A descrição detalhada da linguagem pode ser encontrada em Beckett, 2004.

2. OWL

Ontology Web Language (OWL) (McGuinness & Harmelen, 2004) é uma linguagem recomendada pelo consórcio W3C que se baseia na extensão do RDFS para descrever ontologias.

OWL é tipicamente usada para definir uma ontologia para um domínio particular. Uma ontologia OWL (Lacy, 2005) é um conjunto de axiomas que descrevem classes, propriedades e o relacionamento entre elas. O RDF/XML é usado como marcação em conformidade com a OWL para as instâncias dos dados.

OWL possui três linguagens (em ordem decrescente de expressividade) (Lacy, 2005):

- OWL Full é o conjunto completo da linguagem OWL que não coloca nenhuma restrição ao RDF. As declarações RDF são misturadas com os construtores da OWL Full.
- OWL DL é um subconjunto da OWL Full computacionalmente mais eficiente. O principal propósito da OWL DL é fornecer uma lógica de descrição (DL) que dê suporte à aplicações de lógica. OWL DL restringe o uso de alguns construtores da OWL Full.
- OWL Lite é um subconjunto da OWL Full que possui um conjunto mínimo de marcações para usuários que querem se beneficiar de ontologias sem um investimento significativo com códigos que representam relacionamentos semânticos complexos.

OWL Lite e OWL DL podem ser entendidas como uma extensão de uma *view* restrita do RDF.

O conjunto completo dos axiomas OWL assim como a descrição detalhada da linguagem pode ser encontrado em http://www.w3.org/2004/OWL/

3. Sistema de Informação Geográfica - SIG

De acordo com Queiroz & Ferreira (2006) o termo sistema de informação geográfica é aplicado para sistemas que realizam o tratamento computacional de dados geográficos. A principal diferença de um SIG para um sistema de informação convencional é sua capacidade de armazenar tanto os atributos descritivos (informações alfanuméricas) como as geometrias dos diferentes tipos de dados geográficos. Assim, para cada lote num cadastro urbano, um SIG guarda, além de informação descritiva como proprietário e valor do IPTU, a informação geométrica com as coordenadas dos limites do lote. A partir destes conceitos, é possível indicar as principais características de SIGs (Queiroz & Ferreira, 2006):

- Inserir e integrar, numa única base de dados, informações espaciais provenientes de meio físico-biótico, de dados censitários, de cadastros urbano e rural, e outras fontes de dados como imagens de satélite, e GPS.
- Oferecer mecanismos para combinar as várias informações, através de algoritmos de manipulação e análise, bem como para consultar, recuperar e visualizar o conteúdo da base de dados geográficos.

4. Formato Vetorial

As estruturas de dados utilizadas em bancos de dados geográficos podem ser divididas em duas grandes classes: estruturas vetoriais e estruturas matriciais.

Segundo (Queiroz & Ferreira, 2006) as estruturas vetoriais são utilizadas para representar as coordenadas das fronteiras de cada entidade geográfica, através de três formas básicas: pontos, linhas, e áreas (ou polígonos), definidas por suas coordenadas cartesianas. Um ponto é um par ordenado (x, y) de coordenadas espaciais. O ponto pode ser utilizado para identificar localizações ou ocorrências no espaço. São exemplos: localização de crimes, ocorrências de doenças, e localização de espécies vegetais. Uma linha é um conjunto de pontos conectados. A linha é utilizada para guardar feições unidimensionais. De uma forma geral, as linhas estão associadas a uma topologia arco-nó, descrita a seguir. Uma área (ou polígono) é a região do plano limitada por uma ou mais linhas poligonais conectadas de tal forma que o último ponto de uma linha seja idêntico ao primeiro da próxima. Observe-se também que a fronteira do polígono divide o plano em

duas regiões: o interior e o exterior. Os polígonos são usados para representar unidades de dados geográficos espaciais individuais (setores censitários, distritos, zonas de endereçamento postal, municípios). Para cada unidade, são associados dados oriundos de levantamentos como censos e estatísticas de saúde.

5. Formato Raster

Segundo (Queiroz & Ferreira, 2006) as estruturas matriciais ou *raster* usam uma grade regular sobre a qual se representa, célula a célula, o elemento que está sendo representado. A cada célula, atribui-se um código referente ao atributo estudado, de tal forma que o computador saiba a que elemento ou objeto pertence a uma determinada célula. Nesta representação, o espaço é representado como uma matriz P(m, n) composto de m colunas e n linhas, onde cada célula possui um número de linha, um número de coluna e um valor correspondente ao atributo estudado e cada célula é individualmente acessada pelas suas coordenadas.

A representação matricial supõe que o espaço pode ser tratado como uma superfície plana, onde cada célula está associada a uma porção do terreno. A resolução do sistema é dada pela relação entre o tamanho da célula no mapa ou documento e a área por ela coberta no terreno.

6. Relacionamentos Topológicos

Existem várias propostas de modelos com o objetivo de descrever os possíveis relacionamentos entre dois objetos (Queiroz & Ferreira, 2006). Os modelos adotados nas implementações da maioria dos SIGs seguem o paradigma das matrizes de interseção introduzida por Max Egenhofer.

No modelo chamado de matriz de 4-interseções (ver a Figura 54), oito relações topológicas binárias são consideradas, representando a interseção entre a fronteira e o interior de duas geometrias (Egenhofer & Franzosa, 1991; apud Queiroz & Ferreira, 2006).

Para definir relacionamentos topológicos entre geometrias com estruturas mais complexas, como regiões com ilhas e separações, é necessário estender a matriz de 4-Interseções para também considerar o exterior de uma geometria. O novo modelo, chamado de matriz de 9-Interseções (ver Figura 55), considera então o resultado da interseção entre as fronteiras, interiores e exteriores de duas

geometrias. Maiores detalhes sobre relações topológicas entre regiões com ilhas podem ser encontrados em (Egenhofer & Herring, 1991; apud Queiroz & Ferreira, 2006).

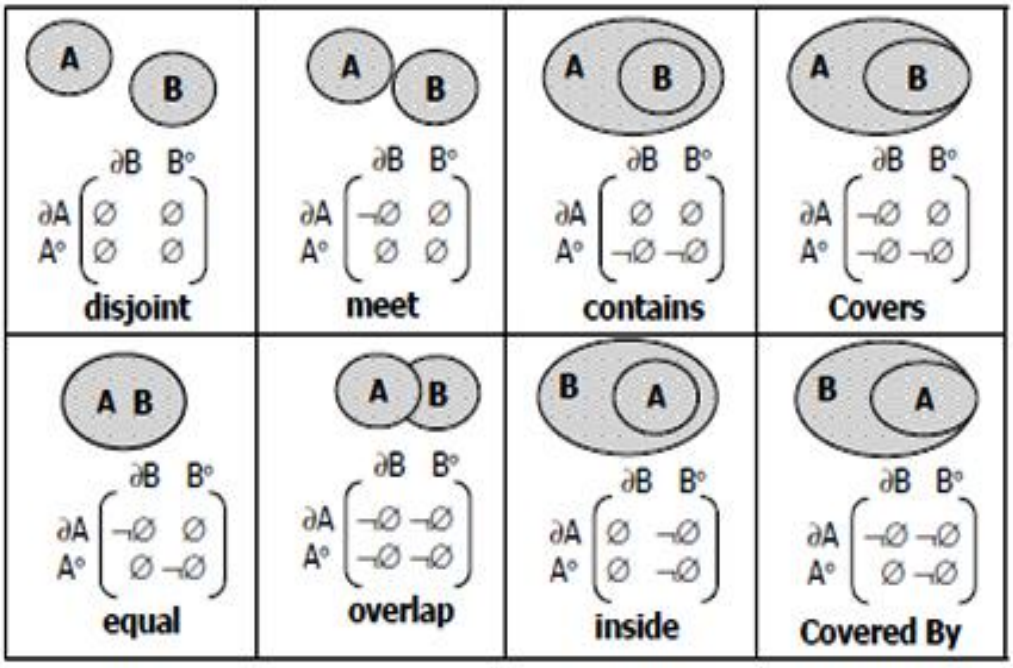


Figura 54 - Matriz de 4-Interseções para relações entre duas regiões. Fonte: (Egenhofer & Franzosa, 1991 apud Queiroz & Ferreira, 2006).

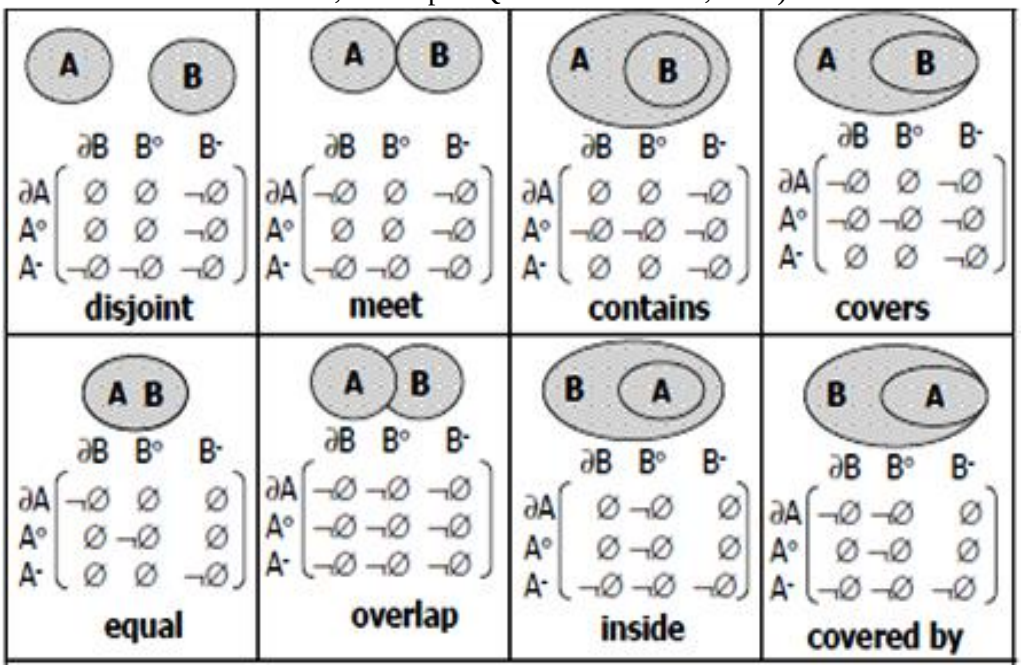


Figura 55 - Matriz de 9-Interseções para relações entre duas regiões.
Fonte:(Egenhofer & Franzosa, 1991, apud Queiroz & Ferreira, 2006).

7. Cubos de dados estatísticos

Os cubos de dados são utilizados na modelagem dimensional de um *Data Warehouse* (DW). A modelagem de dados para um DW é diferente da modelagem utilizada em banco de dados convencionais. Nos bancos de dados convencionais é utilizado o MER (Modelo de Entidades e Relacionamentos), enquanto que num DW a modelagem dimensional é uma técnica que suporta o ambiente para análise multidimensional dos dados (Machado, 2000, apud Menolli, 2004).

O modelo dimensional permite a visualização de dados na forma de um **cubo**, onde cada dimensão do cubo representa o contexto de um determinado **fato** e a interseção entre as dimensões representa as **medidas**. Matematicamente, o cubo possui apenas três dimensões. Entretanto, no modelo dimensional, a metáfora do cubo pode possuir quantas dimensões forem necessárias para representar um determinado **fato** (Machado, 2000, apud Menolli, 2004). Por este motivo este modelo também é chamado de modelo multidimensional.

A modelagem dimensional tem como conceito, três elementos principais (Machado, 2000, apud Menolli, 2004):

- fatos;
- dimensões;
- medidas.

Um fato é uma coleção de dados. Cada fato representa um item de negócio, uma transação ou um evento de negócio. Os fatos são utilizados para fazer a análise sobre a empresa, ou a instituição.

Dimensões são os elementos que participam de um fato, como por exemplo: tempo, localização e cliente. A dimensão pode ser organizada de maneira hierárquica, sendo constituída de vários níveis. Por exemplo, a dimensão região pode ser constituída de região, estado e cidade.

As medidas são os atributos numéricos que representam um fato. Cada medida é constituída pela combinação de dimensões que estão em um fato. Um exemplo de medida é o número de publicações de um determinado autor em um ano.

8. Sistema PruSM

O sistema PruSM (Prudent Schema Matching) (Nguyen et al., 2010) é um exemplo da abordagem *virtual integration* que alinha esquemas de formulários Web de múltiplas fontes de informação por similaridade. Primeiramente o sistema recebe os dados de uma coleção de formulários previamente coletados. Estes formulários servem como entrada para um módulo de agregação que utiliza técnicas para a preparação dos dados, como por exemplo, a remoção de *stop words*[33]. A próxima etapa é o módulo de *Descoberta de Relacionamentos*, onde são analisados os relacionamentos e a frequência entre os rótulos (ou atributos) do esquema dos formulários. Primeiramente, é calculada a similaridade dos rótulos do formulário, fornecendo uma medida da importância do termo dentro do conjunto, que é baseada na frequência do termo no conjunto. Em seguida, calcula-se a similaridade entre os valores do atributo. Por fim, é realizada a correlação dos valores dos atributos do conjunto, para detectar casos como, por exemplo, *Make* e *Brand*, que apesar de possuírem grafia diferente, possuem o mesmo significado. Ao final do processo, os atributos com baixa frequência têm seus pesos recalculados através de um algoritmo chamado STF (*Singular Token Frequence*). Em seguida, aplica-se um algoritmo conhecido como 1NN (*1-Nearest-Neighbor Clustering*), que agrupa o atributo raro ao seu vizinho mais próximo de maior frequência. Estes passos geram um novo cluster de atributos considerados representativos. Finalmente, incorpora-se este novo cluster ao cluster confiável, utilizando um algoritmo denominado HAC (*Hierarchical Agglomerative Clustering*), onde os atributos são representados em uma estrutura de árvore e são aglomerados aos mais próximos sucessivamente. Então é executada a identificação de conjuntos de alta confiança, ou seja, clusters de atributos afins.

[33] Palavras ou termos que podem ser retirados do texto pelos mecanismos de busca como, por exemplo: no, o, uma, quem, etc.

Apêndice B: Esquema do Banco de Dados W-RayS

Para facilitar o entendimento do esquema do BD W-RayS a Figura 56 foi dividida em duas partes. A primeira (letra B em vermelho) modela os conceitos do *esquema das views* e os parâmetros necessários para: o mapeamento RDB-to-RDF; o mapeamento OWL-to-LN; e a geração do site W-Ray. A segunda (letra A em vermelho) modela: os vocabulários carregados e os alinhamentos entre os vocabulários e *ontologia da aplicação* que será gerada.

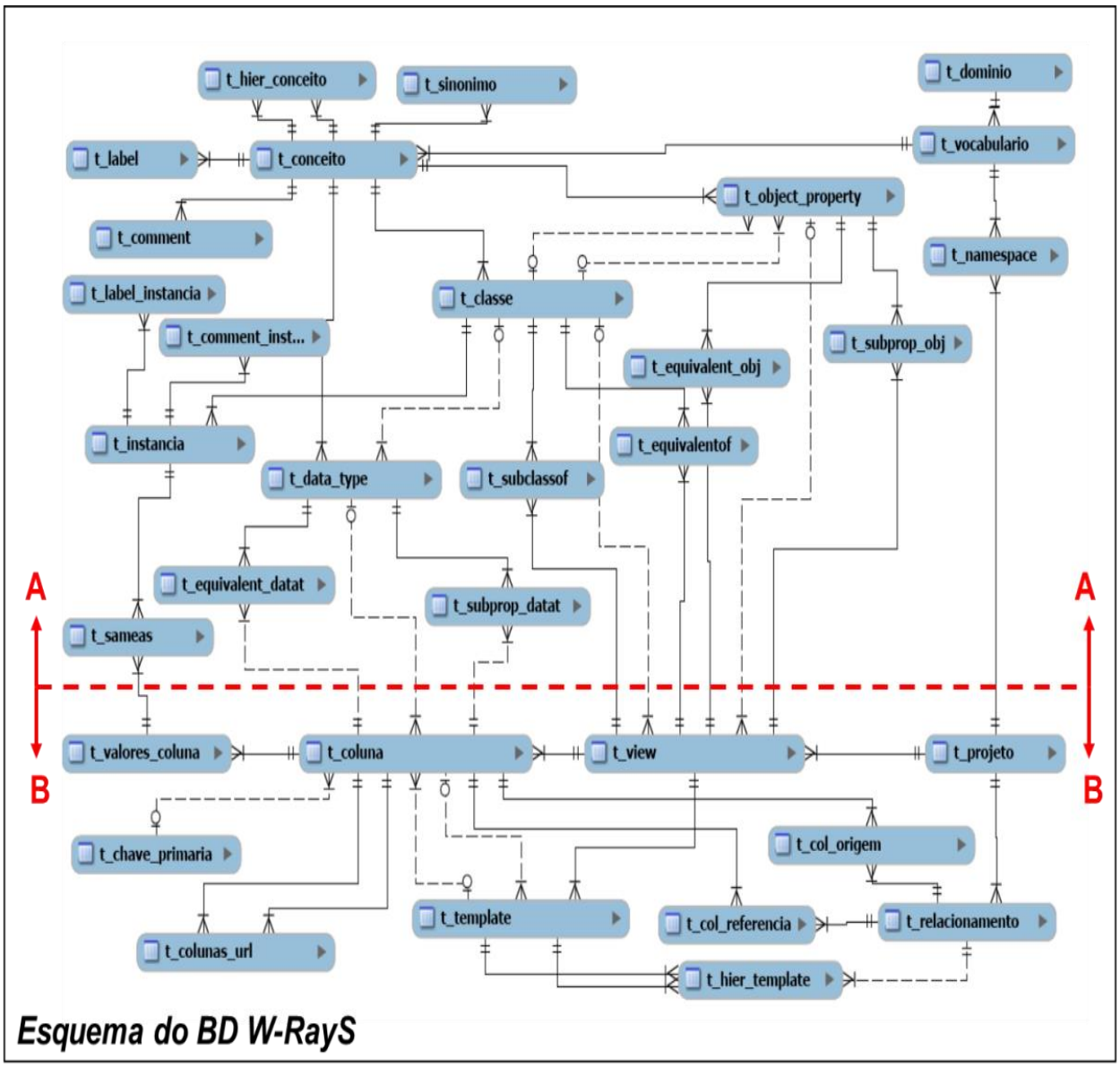


Figura 56 - Esquema do banco de dados W-RayS

Apêndice C: Passo a passo da ferramenta W-RayS

A Figura 57 mostra o menu principal da ferramenta que é composto por cinco módulos principais:

1. Módulo de Carga das Views - onde o projetista deve carregar o conjunto de *views* dos dados que serão publicados na Web.
2. Módulo de Projeto da Ontologia da Aplicação – ajuda o projetista a selecionar os vocabulários apropriados para criar um esquema RDF, que modele o esquema das *views* de dados e seja alinhado com outras ontologias;
3. Módulo de Projeto de *Template* de Linguagem Natural – ajuda o projetista a melhorar os *templates* predefinidos para o mapeamento OWL-to-LN;
4. Módulo de Design do Site - ajuda o projetista a definir a estrutura do site de acordo com as diretrizes da abordagem W-Ray;
5. Módulo de Geração do Site - responsável pela geração do site W-Ray de acordo com o projeto especificado.

Figura 57- Menu principal da Ferramenta W-RayS

A seguir descrevemos o passo a passo para cada um dos módulos.

1 Módulos de Carga

A etapa de carga das *views* e vocabulários deve ser efetuada antes do início dos ajustes para a geração do site W-Ray. A Figura 58 mostra a entrada para os módulos de carga.

Figura 58 – Entrada para os módulos de carga

1.1 Carga das views geradas no DB usuário

O projetista deve selecionar quais dados serão publicados criando *views* materializadas sobre o banco de dados. As *views* devem exteriorizar dados que fornecem um resumo dos objetos mais importantes e seus atributos, dentro dos limites de privacidade. A abordagem W-Ray oferece uma lista de recomendações que o projetista deve seguir ao definir as *views*, conforme capítulo 3.

O banco de dados W-RayS armazena todas as informações sobre as *views* geradas pelo usuário para o projeto W-Ray (chaves estrangeiras, chaves primárias, tipo de dados, nomes das colunas e valores). Para efetuar a carga das *views* no BD W-RayS o usuário deve informar o nome do driver-ODBC e criar um projeto (formulários não apresentados aqui). Desta forma, a ferramenta pode acessar o catálogo do BD do usuário e disponibilizar a lista de *views* existentes a fim de possibilitar a seleção das respectivas *views* para a carga no BD W-RayS (Figura 59).

Figura 59 - Formulário de carga das *views* do BD do usuário

A Figura 58 também oferece a possibilidade de carga de tabela. Isto se deve ao fato de que algumas vezes o usuário pode considerar que não existe a necessidade de criação de uma *view*, por exemplo, todos os dados da tabela são relevantes para a publicação via W-RayS. Assim, o usuário pode carregar diretamente a tabela do BD desde que respeite todas as restrições de integridade envolvidas (Figura 60).

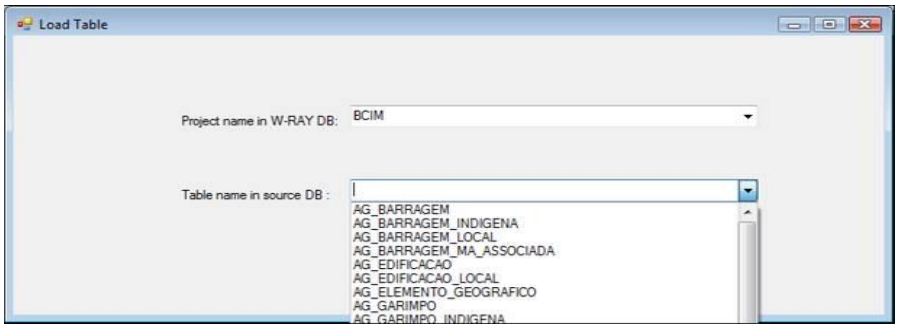


Figura 60 - Formulário de carga de tabelas do BD do usuário

1.2 Carga dos vocabulários

Após a fase de concepção das *views*, o designer pode selecionar um ou mais vocábulos que pertencem ao domínio da aplicação em questão. Entende-se por vocabulário: uma ontologia, um tesauro ou mesmo um simples glossário. Os vocabulários selecionados devem ser carregados no BD W-RayS para uso posterior (Figura 61). O BD W-RayS possui o WordNet.owl já carregado.

As ontologias são "enxugadas", ou seja, o programa filtra, antes da carga, apenas os nomes de *classes*, *datatype properties*, *object properties* e as respectivas descrições (se existirem).

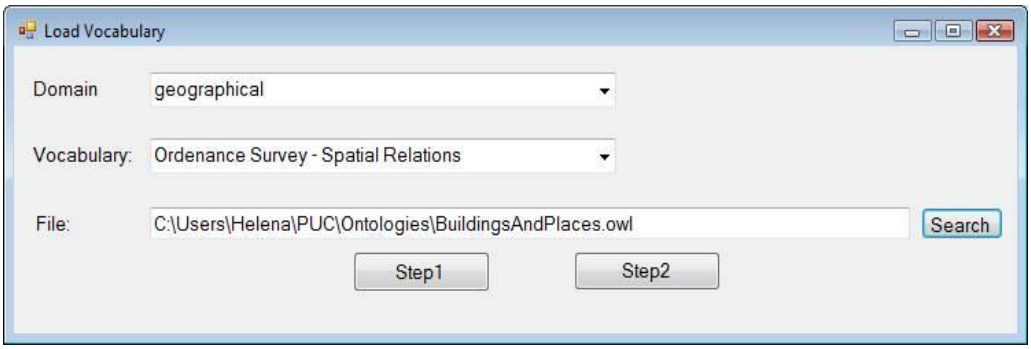


Figura 61 - Formulário de carga de vocabulários para futuros alinhamentos

2 Módulos de projeto da ontologia

Para a geração de triplas RDF e a inclusão de RDFa em páginas Web é necessário o mapeamento do esquema *views* para o esquema RDF. Este mapeamento deve estar alinhado com, ou reutilizar, os termos dos vocabulários já carregados. O esquema RDF resultante é denominado "ontologia de aplicação". O *Módulo Projeto da Ontologia* ajuda o designer a selecionar os vocabulários apropriados e alinhar as ontologias.

Figura 62 - Sub-módulos de alinhamento de ontologias

Todos os sub-módulos de alinhamento de ontologias listados na Figura 62 oferecem facilidades de busca para ajudar o projetista na tarefa de alinhamento dos termos da *ontologia da aplicação* com os termos de outras ontologias. O módulo também permite que o projetista procure sinônimos com a ajuda da ontologia WordNet.

2.1 Sub-módulo de Alinhamento de Classe

A Figura 63 mostra o sub-módulo que permite o alinhamento de uma classe da *ontologia da aplicação* com uma classe de outra ontologia através das primitivas *rdfs:subClassOf e owl:equivalentClass*. A Figura 64 permite o reuso de classe de outra ontologia. A Figura 65 apresenta as facilidades de busca para alinhamento de classe.

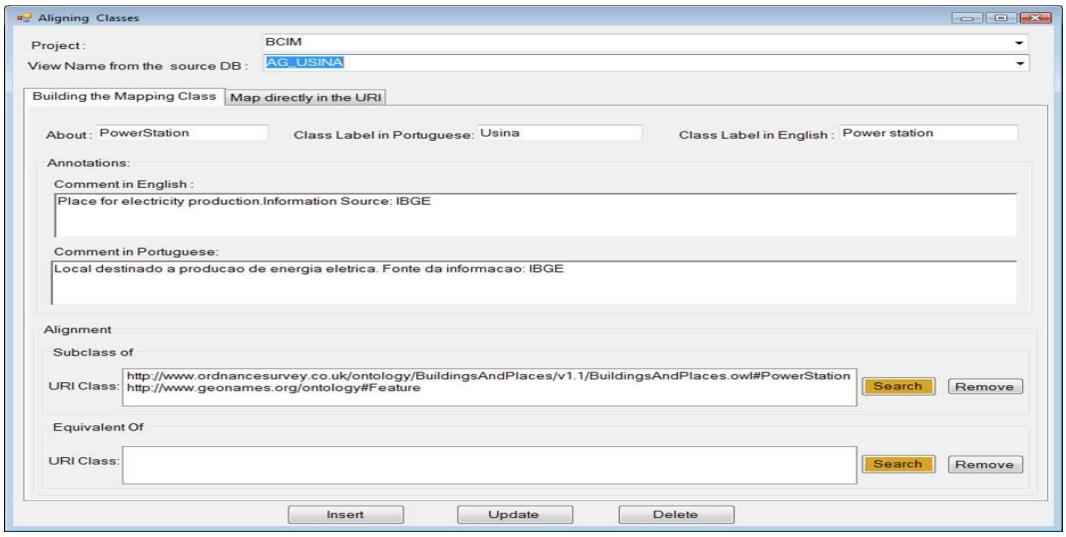


Figura 63- Sub-módulo de alinhamento de Classe

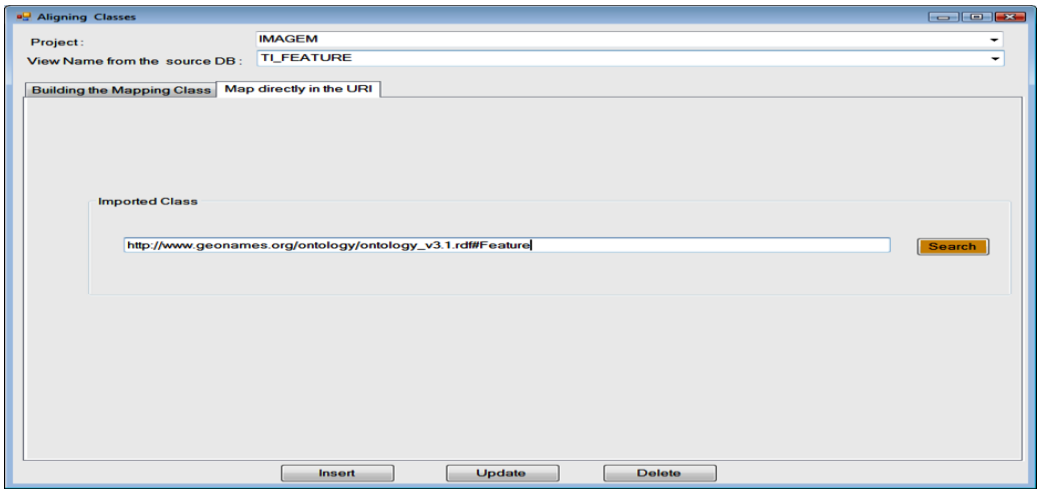


Figura 64 - Como informar a reutilização de uma classe para o mapeamento

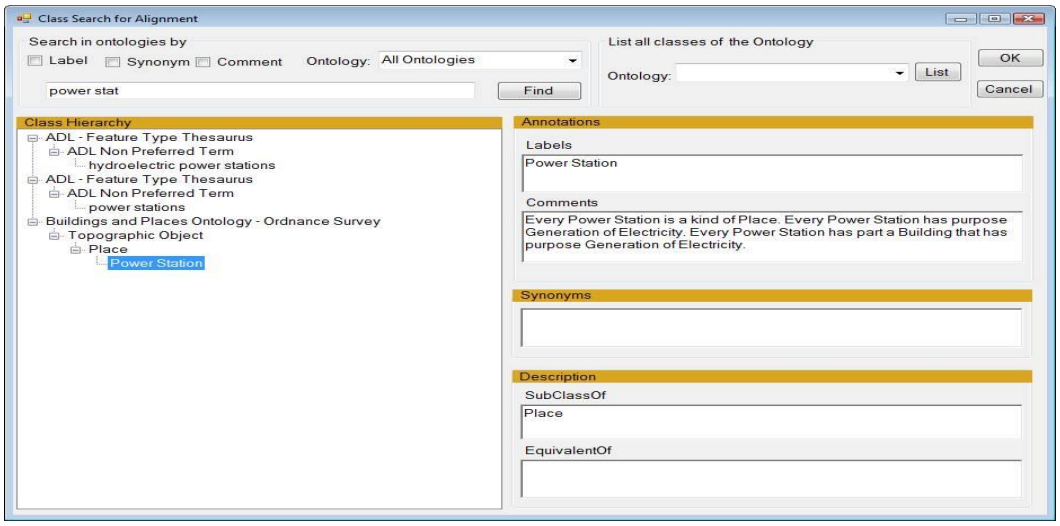


Figura 65- Facilidade de busca para alinhamento de Classe

2.2 Sub-módulo de Alinhamento de Object Property

A primeira aba da Figura 66 permite o alinhamento com uma *object porperty* de outra ontologia através das primitivas *rdfs:subPropertyOf e owl:equivalentProperty*. A segunda aba da Figura 67 permite o reuso de uma *object property* de outra ontologia. A Figura 68 mostra a facilidade de busca de *object porperties* de outras ontologias.

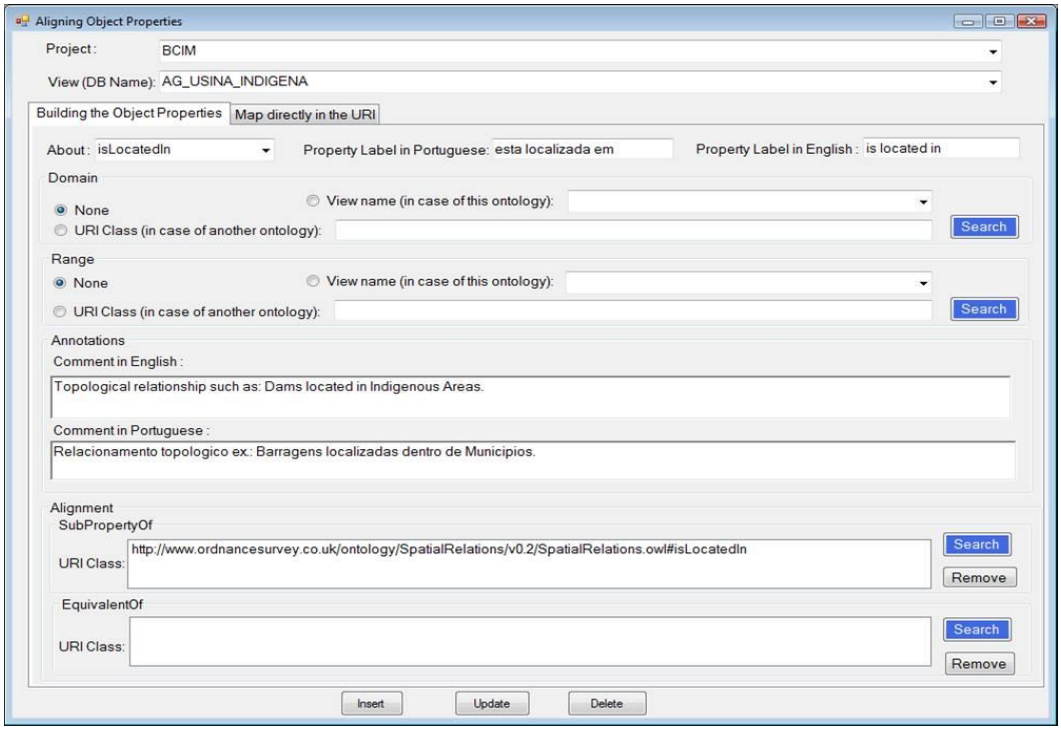


Figura 66 - Sub-módulo de alinhamento de *object property*

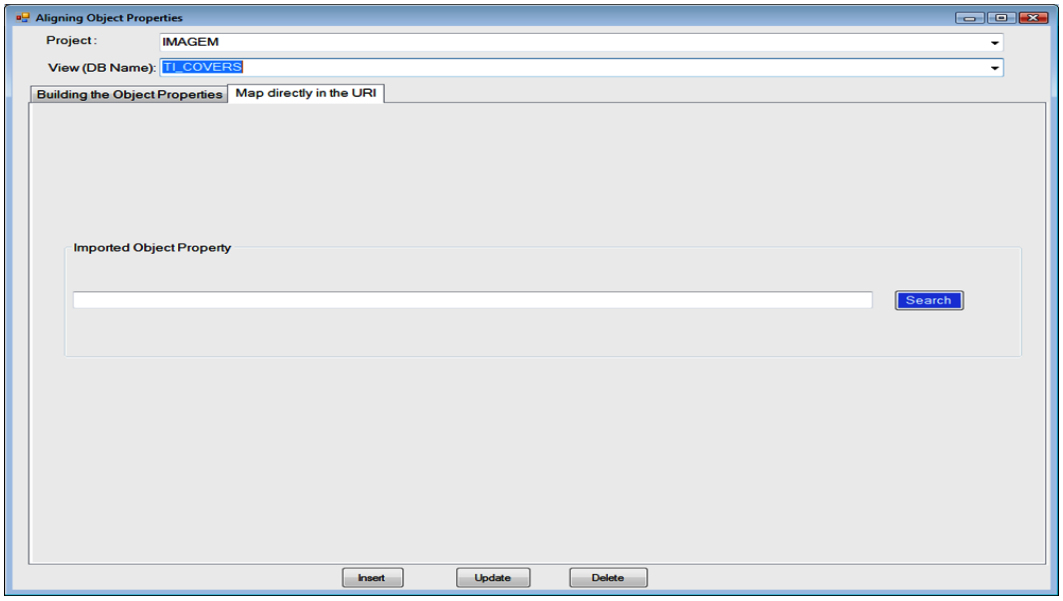


Figura 67 - Aba que permite o reuso de *object porperty* de outro vocabulário

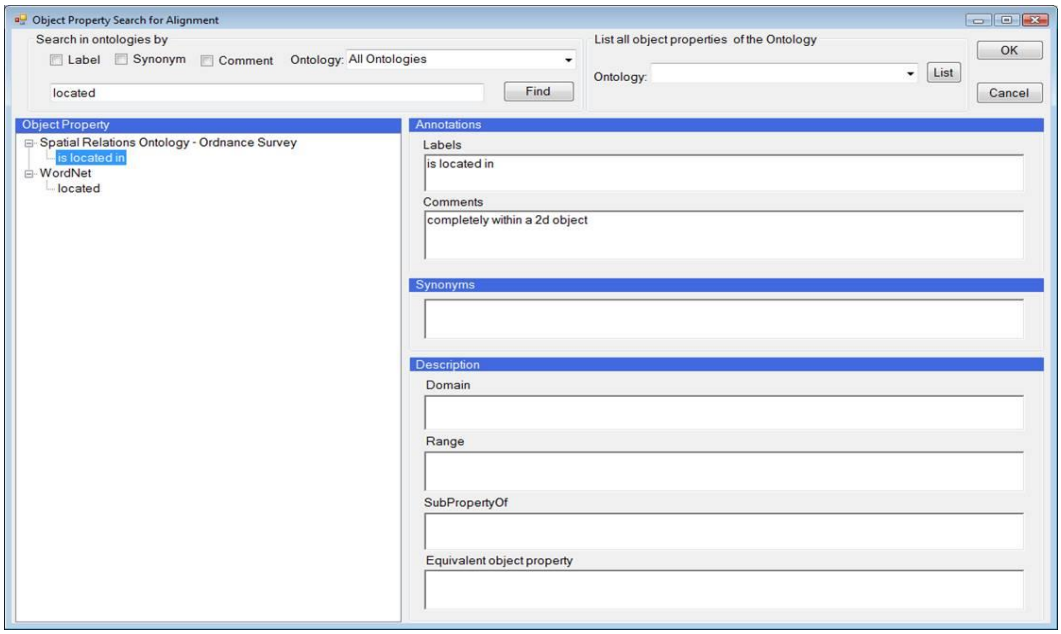


Figura 68 - Facilidade de busca de *object properties*

2.3 Módulo alinhamento de *datatype property*

A primeira aba da Figura 69 permite o alinhamento com uma *datatype porperty* com outra ontologia através das primitivas *rdfs:subPropertyOf e owl:equivalentProperty*. A segunda aba da Figura 70 permite o reuso de uma *datatype property* de outra ontologia. A Figura 71 mostra a facilidade de busca de *datatype porperties* de outras ontologias.

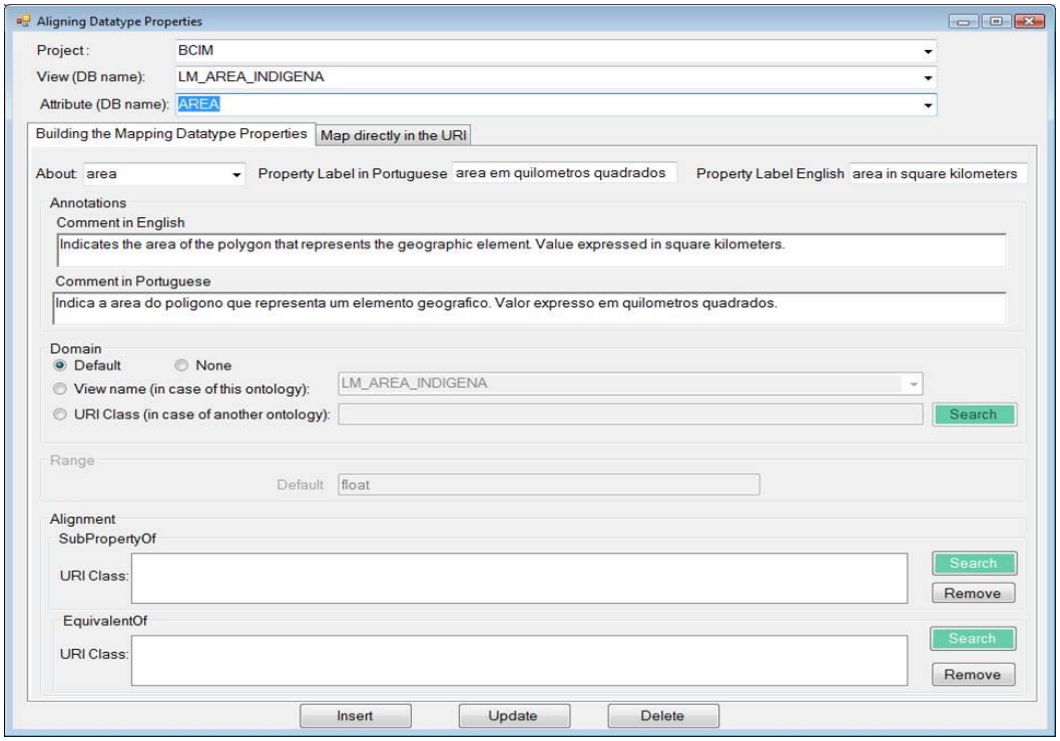


Figura 69- Sub-módulo de alinhamento de *datatype property*

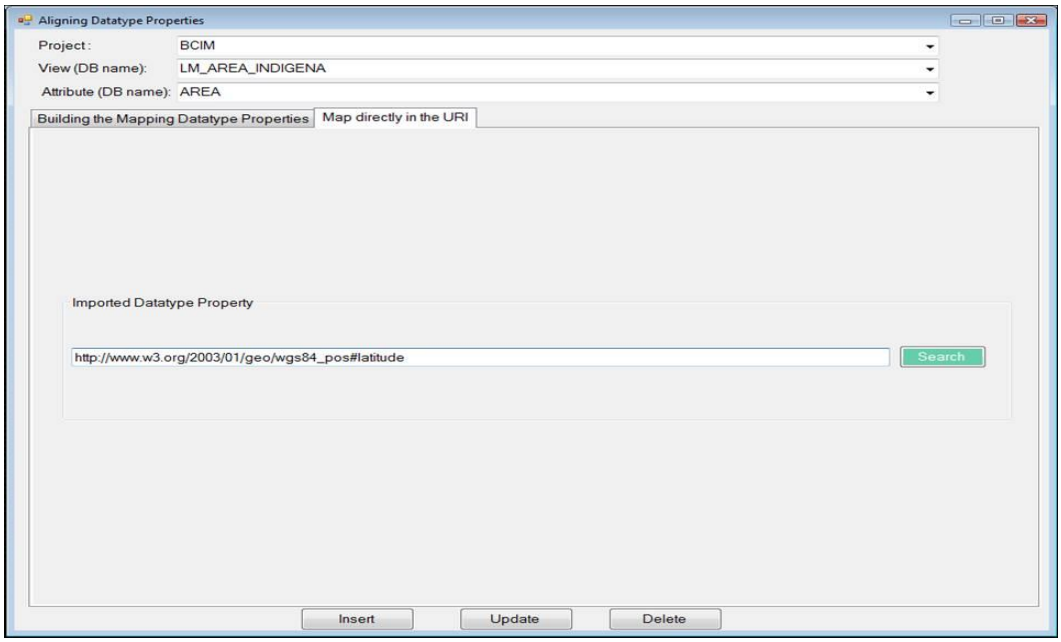


Figura 70 - Aba que permite o reuso de *datatype porperty* de outro vocabulário

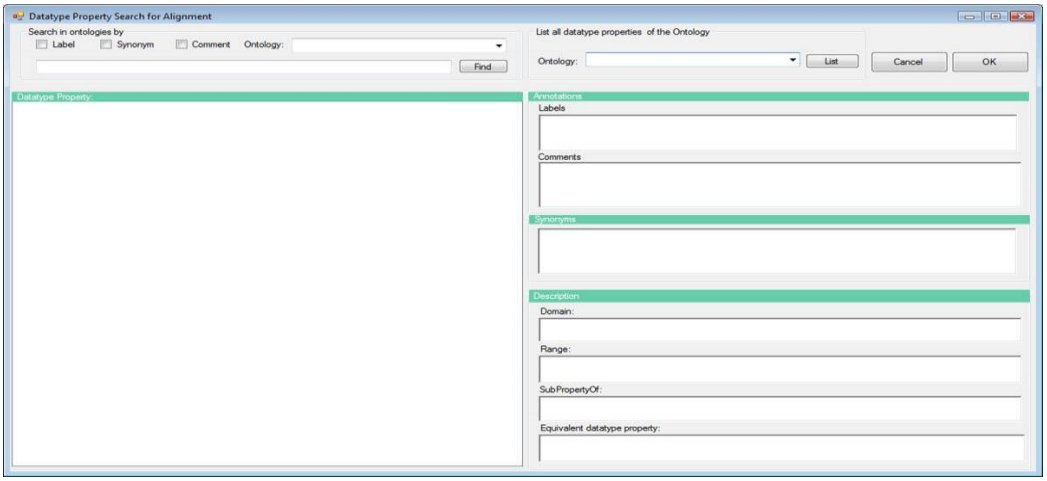


Figura 71 - Facilidade de busca de *object properties*

2.4 Sub-módulo alinhamento de indivíduos

Este sub-módulo permite o alinhamento de um indivíduo da *ontologia da aplicação* com outro de outra ontologia através da primitiva owl:sameAs (Figura 72). A Figura 73 71 mostra a facilidade de busca de *indivíduos*.

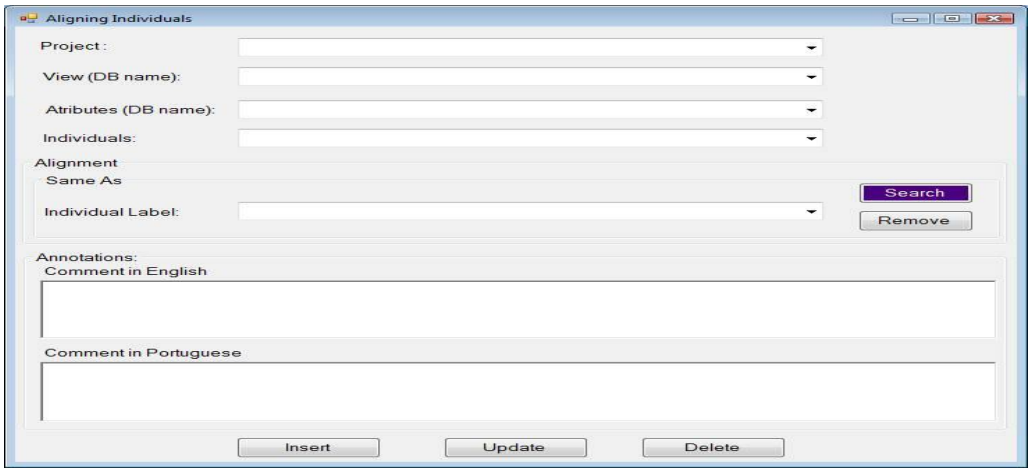


Figura 72 - Sub-módulo de alinhamento de *indivíduos*

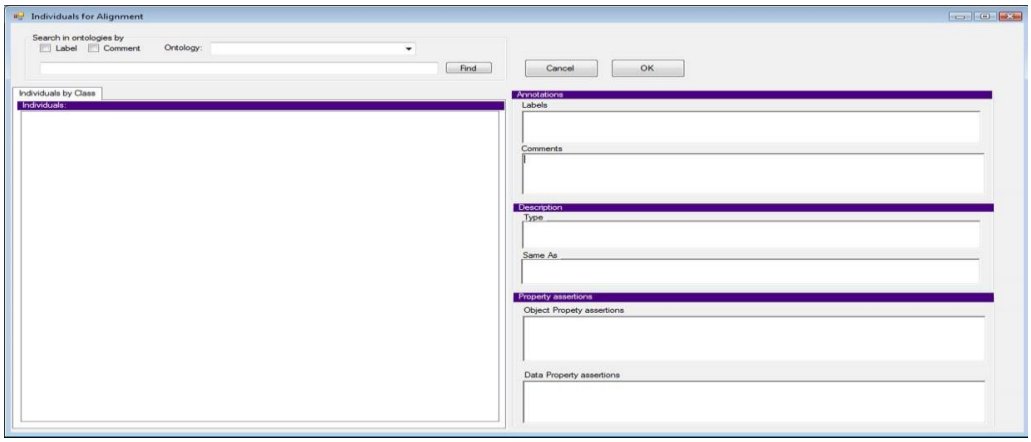


Figura 73 - Facilidade de busca de *indivíduos*

3 Módulo de Projeto de Templates

A Figura 74 mostra a entrada para os sub-módulos onde podem ser definidos alguns parâmetros a fim de ajudar a melhorar a legibilidade da linguagem natural. Vale observar que as sentenças podem ser geradas automaticamente sem a definição desses parâmetros.

Cada sub-módulo possui uma ferramenta de busca (Figura 76) nos vocabulários carregados no BD W-RayS. Desta forma, um alinhamento mais simples pode ser executado sem a necessidade do conhecimento de toda a tecnologia da Web Semântica. Como o módulo de busca de termos de outros vocabulários é o mesmo para *views*, colunas e relacionamentos apenas uma figura será apresentada a título de ilustração. Porém, seu código é capaz de identificar o contexto da busca.

Figura 74 - Sub-módulos de ajuste de Linguagem Natural

3.1 Sub-módulo de ajustes nos nomes das *Views*

A Figura 75 mostra um exemplo envolvendo a *view* AG_USINA. Note que o usuário tem apenas acesso ao nome do termo do vocabulário e ao conceito do termo selecionado na Figura 76. Quando o usuário faz a seleção do termo no formulário de busca (Figura 76), o programa armazena esta informação no banco de dados. Para o exemplo mostrado nos formulários abaixo, quando a *ontologia da aplicação* (no caso do exemplo a *BCIM*) é gerada, o alinhamento é executado da seguinte maneira: uma classe *PowerStation* é criada na *ontologia da aplicação* como subclasse da classe *PowerStation* da ontologia *Buildings and Places – Ordnance Survey*.

As caixas de texto do formulário (Figura 75) (Portuguese URL, English URL e URI) são apenas informativas. Mostram que a URI da classe e o link para o conceito de *PowerStation* no site W-Ray foram armazenados no BD.

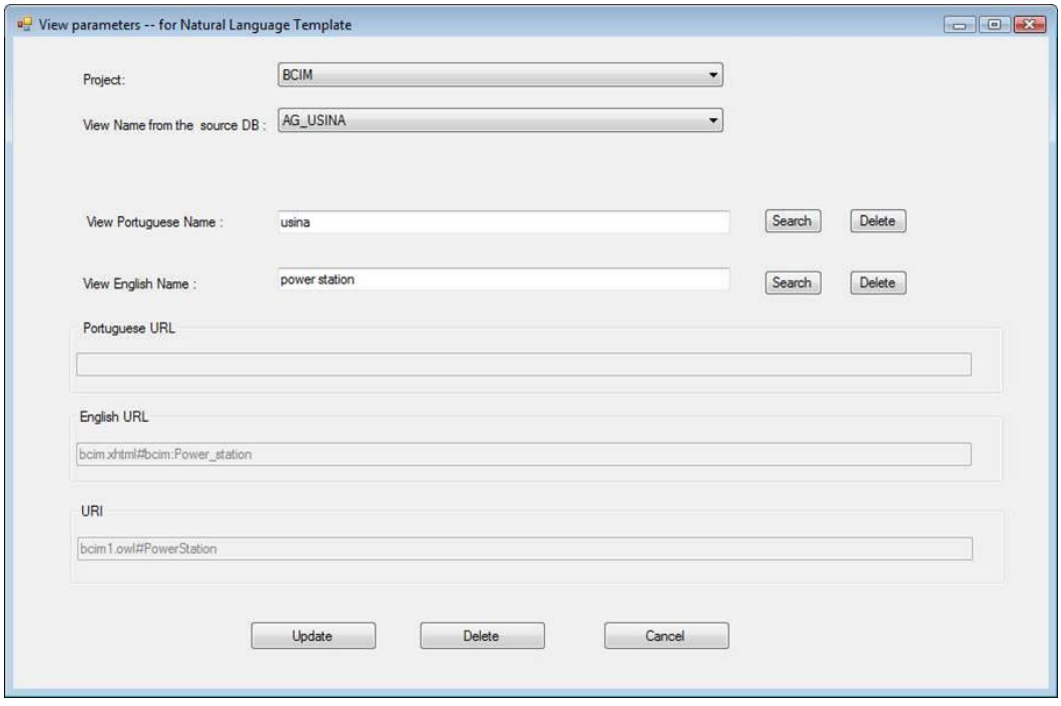


Figura 75 – Sub-módulo de ajuste nos nomes das *views*

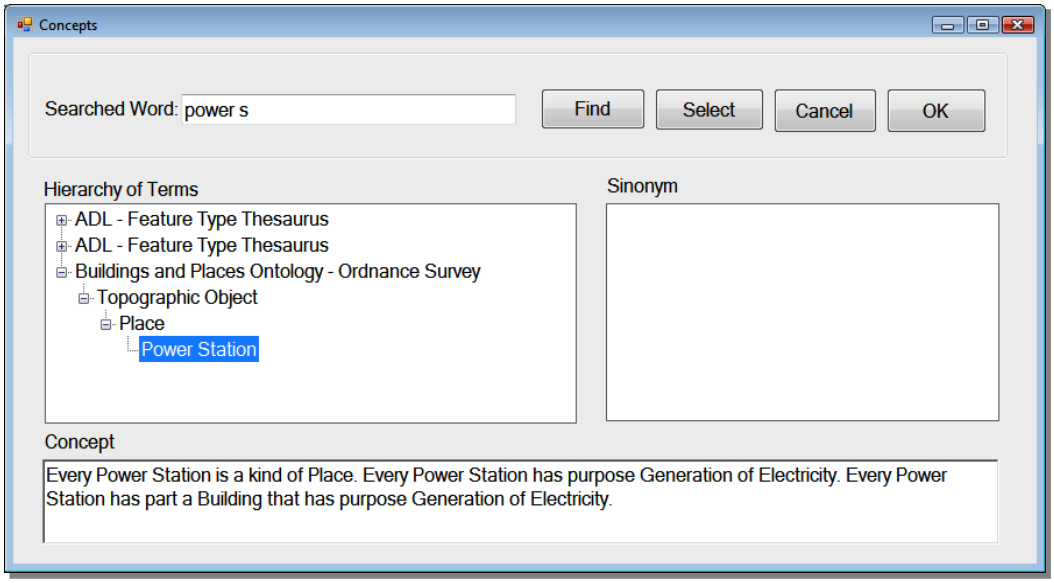


Figura 76 - Facilidade de busca que permite o alinhamento com outras ontologias

3.2 Sub-módulo de ajustes nos nomes das Colunas

Este sub-módulo (Figura 77) permite melhorar a legibilidade da sentença através da atribuição de parâmetros do tipo: se o atributo é sujeito da sentença; se o valor deve ficar entre aspas, negrito ou itálico; se após o nome do atributo deve ser escrito *igual a*; etc. Vale observar que todos os parâmetros possuem valores predefinidos pela ferramenta.

Na segunda aba (Figura 78) o usuário pode fazer ajustes no nome da Coluna. O procedimento executado por este sub-módulo é o mesmo descrito no item anterior para nomes de *views*.

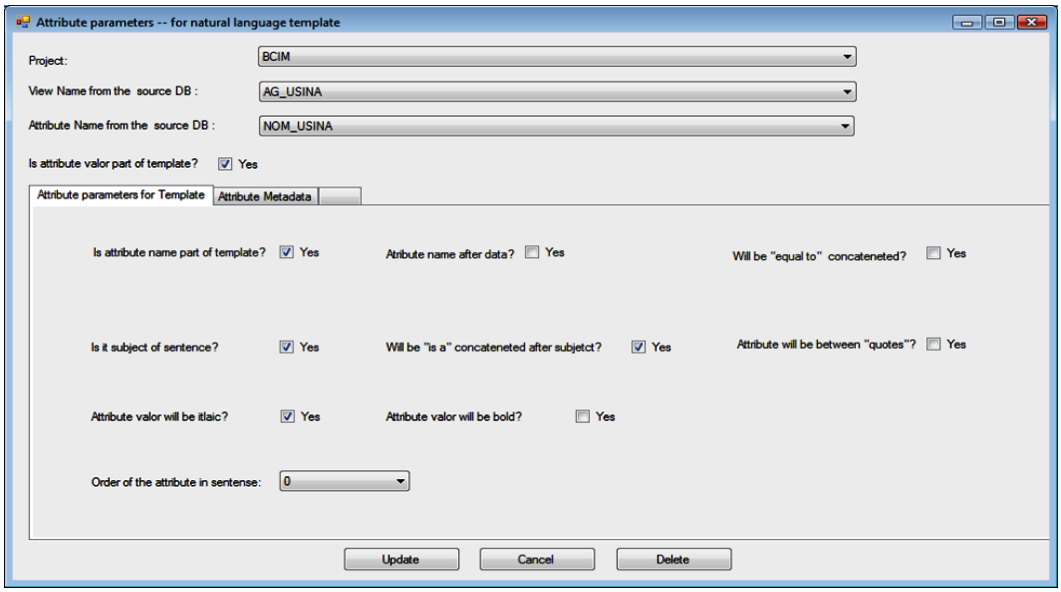


Figura 77 – Aba de parametrização dos nomes das colunas

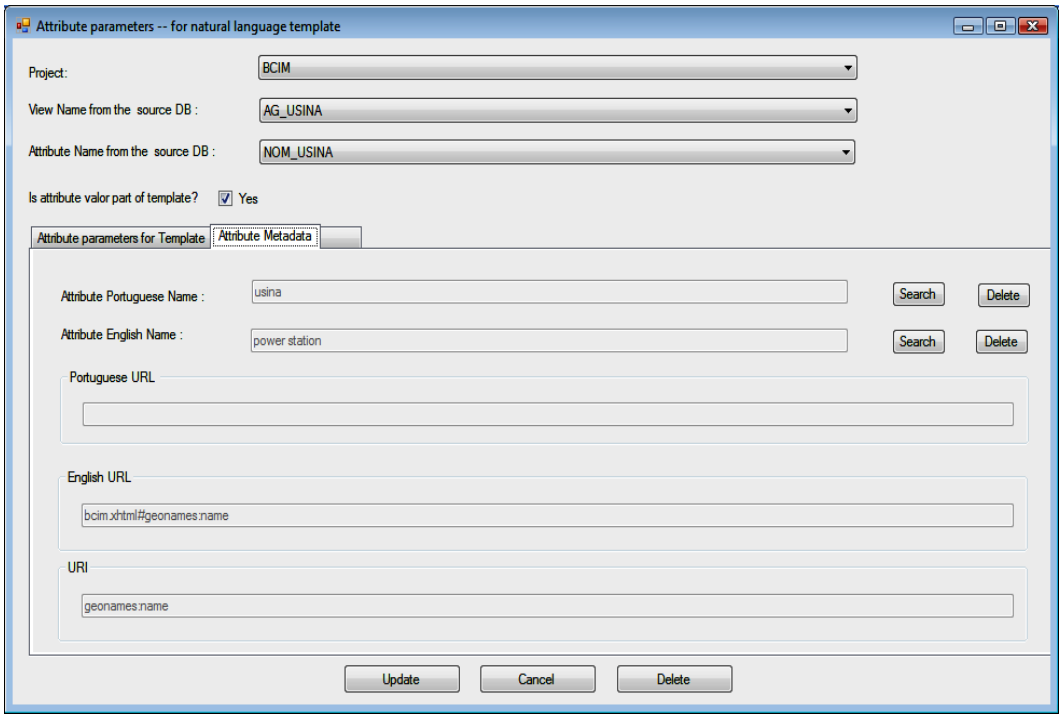


Figura 78 – Aba de ajuste de nomes de colunas

3.3 Sub-módulo de ajustes nos nomes dos Relacionamentos

Particularmente no caso de relacionamentos n_ários as chaves estrangeiras (aqui denominadas relacionamento) podem ser renomeadas e alinhadas com *object properties* de outras ontologias. Neste caso a determinação do sujeito é fortemente recomendada para evitar a geração de sentenças em todos os sentidos do relacionamento n-ário (Figura 79).

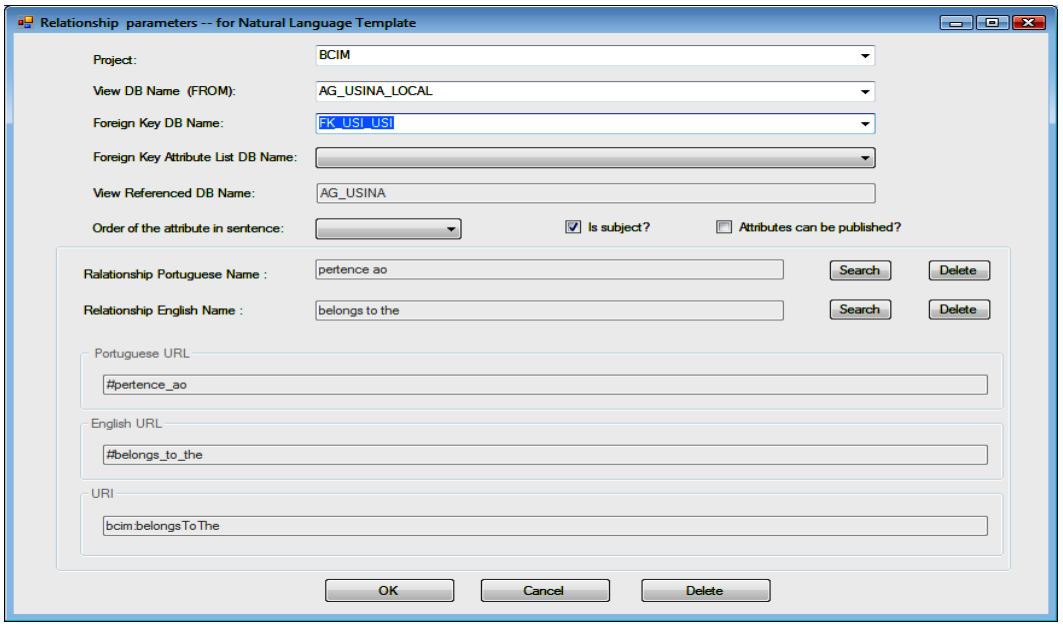


Figura 79 – Sub-módulo de ajuste de nomes de colunas

4 Módulo de Projeto do Web Site

A Figura 80 mostra a entrada para o módulo de Projeto do Web Site W-Ray. Este módulo ajuda o projetista a definir a estrutura do site de acordo com as diretrizes da abordagem W-Ray. Estas diretrizes são formadas por um subconjunto das recomendações para acessibilidade na Web sugeridas pelo W3C e outro subconjunto de recomendações da Google para construção de sites e estão descritas no capítulo 3.

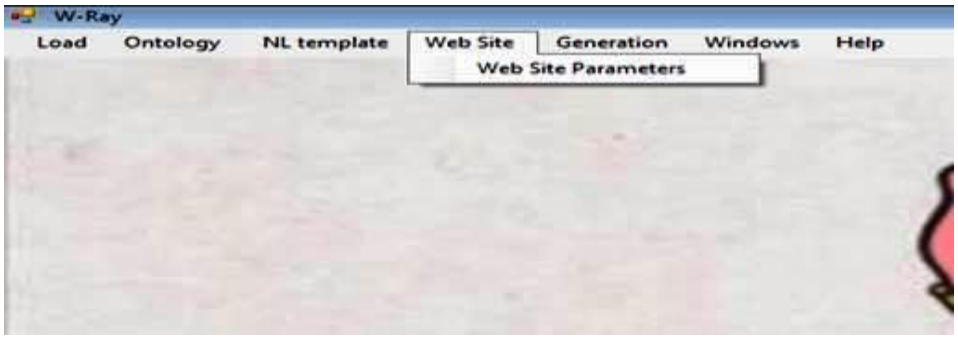


Figura 80 – Entrada para o Módulo de Projeto do Web Site

A Figura 81 mostra a aba que permite a definição de parâmetros que organizam as sentenças na página HTML, tais como: a ordem que diferentes tipos de *templates* aparecerão na página; se as sentenças serão publicadas em uma mesma página ou se existirá quebra; se o tipo de *template* é um cabeçalho, uma sentença comum ou uma descrição de um termo da sentença. A Figura 82 mostra a aba que permite fazer a ligação dos valores das chaves primárias de uma *view* com os respectivos formulários de consulta à *Deep Web*. A Figura 83 mostra como é possível definir uma hierarquia de *templates* para a publicação de "sentenças aninhadas" numa mesma página.

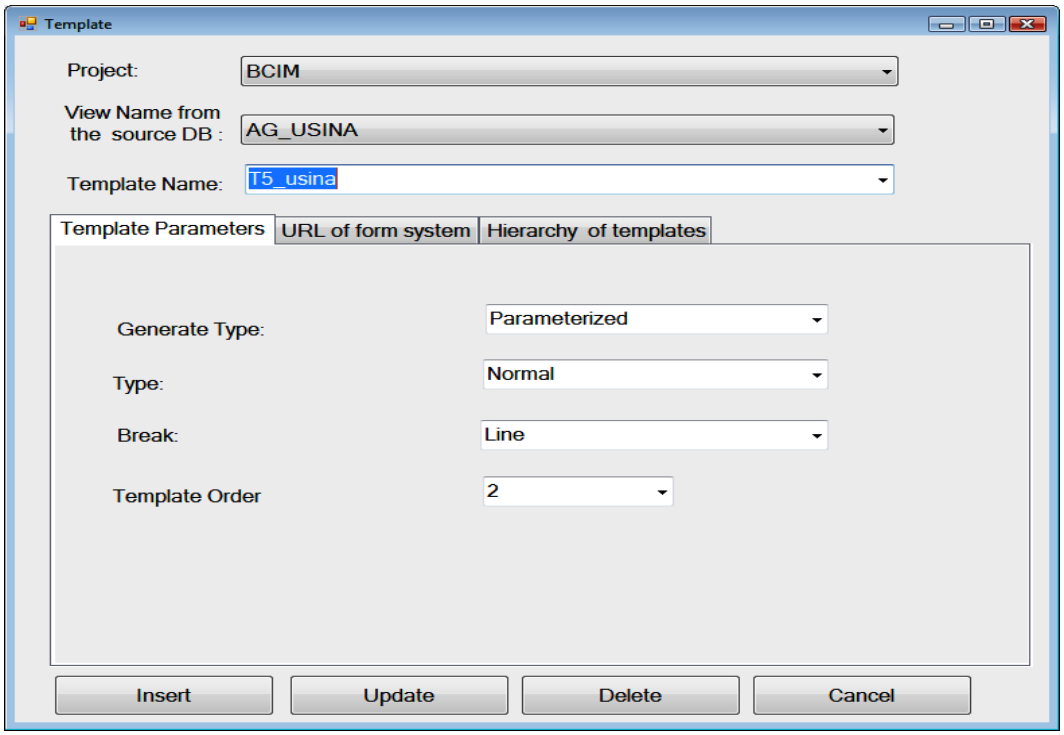


Figura 81 – Aba de parâmetros do *template*

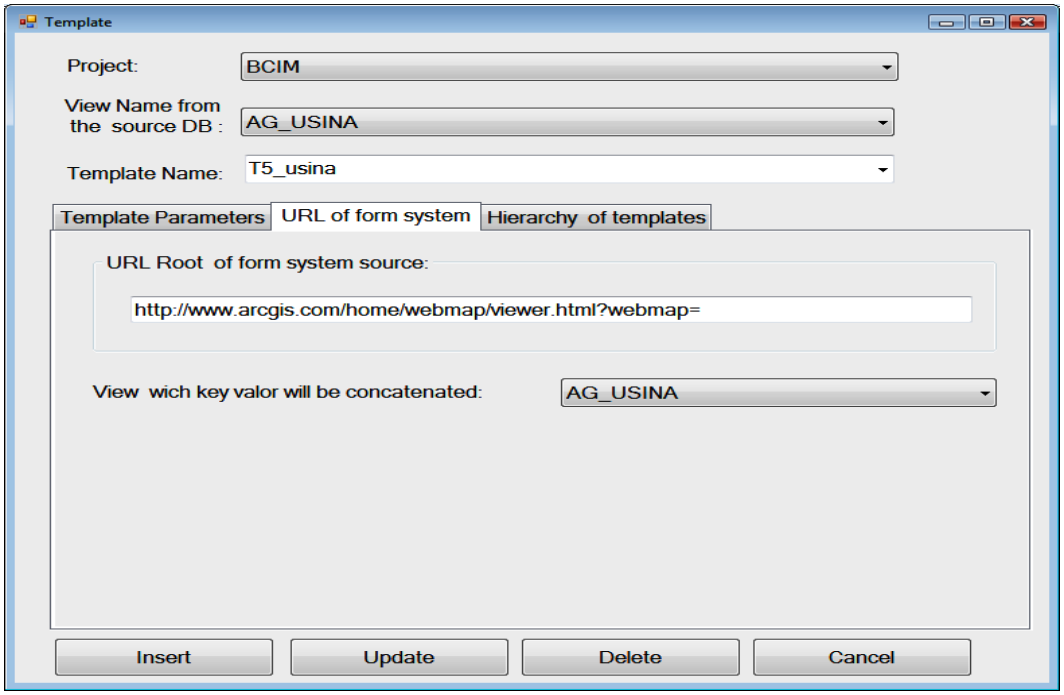


Figura 82 - Aba que permite a ligação com os formulários da *Deep Web*

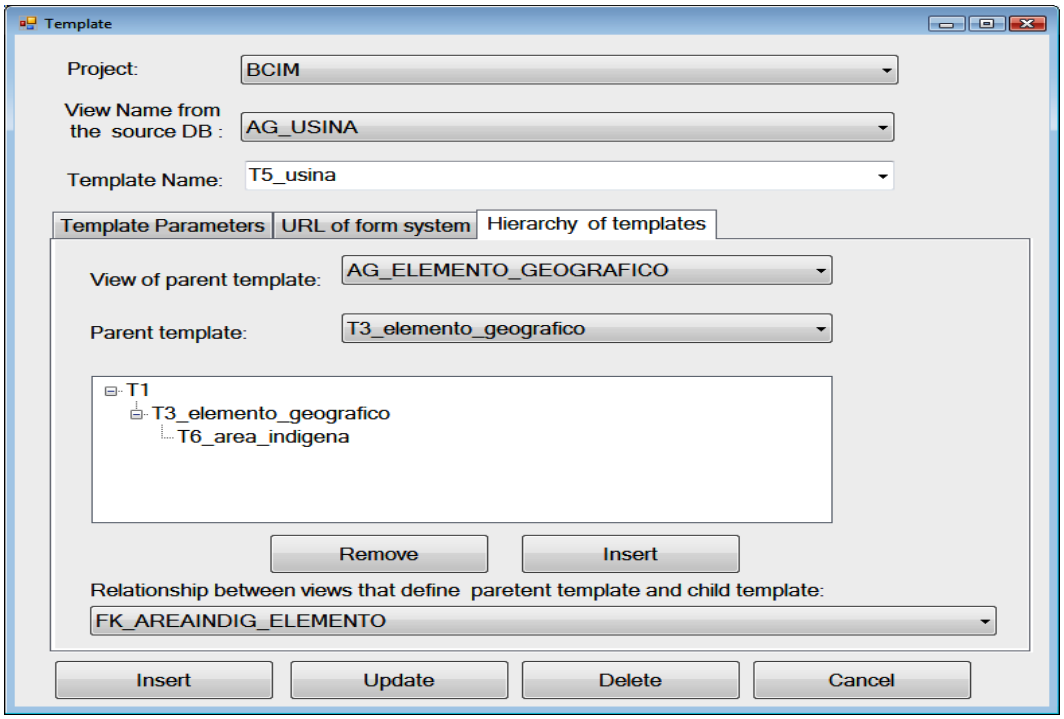


Figura 83 – Aba que permite a criação de uma hierarquia de *templates*

5 Módulos de Geração

A Figura 84 mostra a entrada para os seguintes sub-módulos: geração da *ontologia da aplicação* (Figura 85); geração das sentenças em uma página HTML (Figura 86); geração de sentenças contendo RDFa embutido (Figura 87) e Triplificação dos dados (Figura 88).

Figura 84 – Entrada para os módulos de geração da abordagem W-Ray

5.1 Sub-módulo Geração de Ontologia de Aplicação

A geração de triplas RDF e a inclusão de RDFa em páginas Web tira partido da definição de um esquema RDF, que modela os dados a serem exibidos, de preferência usando vocabulários já conhecidos. Referimo-nos a este esquema RDF, como uma **ontologia de aplicação**. Este sub-módulo (Figura 85) é o responsável pela geração da *ontologia da aplicação* que é mapeada a partir do

esquema composto pelas *views* e do alinhamento executado pelo usuário. A ferramenta W-RayS implementa uma estratégia de mapeamento onde: tabelas são mapeadas em classes, atributos das *views* em *datatype properties* e relacionamentos binários (definido através de chaves estrangeiras) em *object properties*. URIs são geradas com o auxílio de chaves primárias. As relações n-árias são tratadas de acordo com a recomendação do W3C (Noy & Rector, 2006).

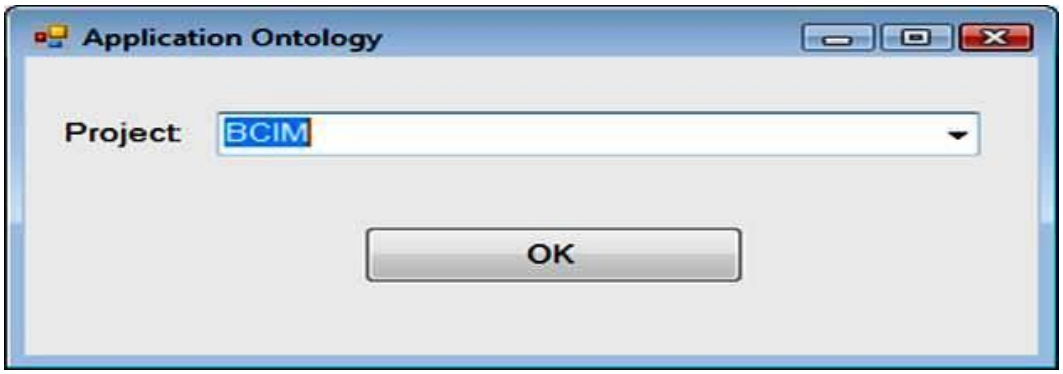


Figura 85 – Sub-módulo Geração de Ontologia de Aplicação

5.2 Sub-módulo Geração de Paginas HTML

Este sub-módulo (Figura 86) é responsável pela geração das sentenças em linguagem natural e das páginas HTML. O módulo implementa OWL-to-LN baseado em *templates* pré-definidos e em todo o projeto do usuário. Se os parâmetros opcionais não foram configurados, o procedimento padrão é executado.

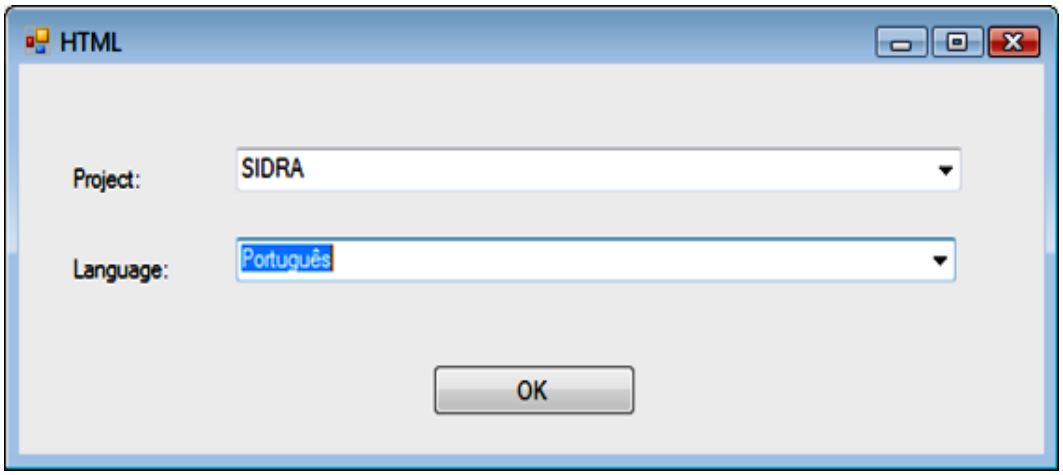


Figura 86 - Sub-módulo Geração do Site em HTML

5.3 Sub-módulo Geração de Paginas HTML com RDFa embutido

Este sub-módulo (Figura 87) é a maior inteligência de todo o sistema. Pode ser considerado como uma extensão do *Módulo de Geração de Páginas HTML*. No que se refere à geração do site W-Ray os procedimentos são os mesmos. A diferença se encontra no RDFa embutido que torna o processo muito parecido com o de triplificação dos dados.

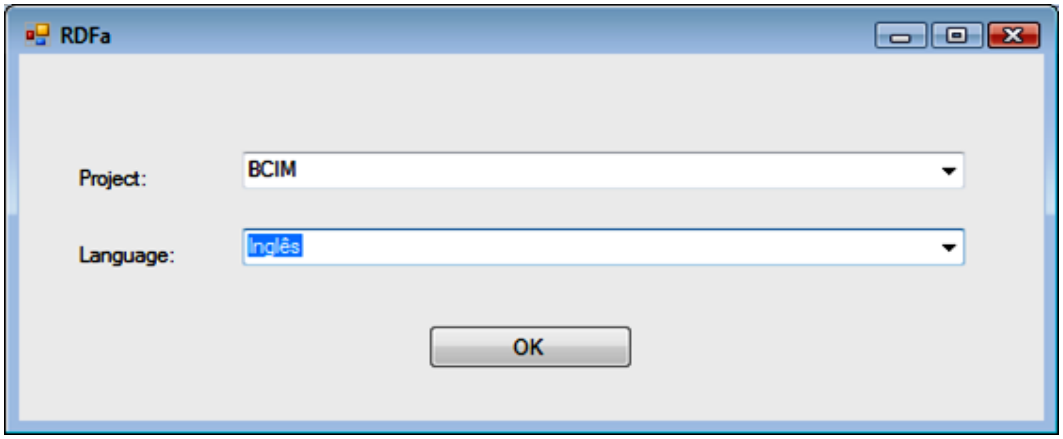


Figura 87 - Sub-módulo Geração do Site em HTML com RDFa embutido

5.4 Sub-módulo de Triplificação

Este sub-módulo (Figura 88) é uma simplificação do módulo *Geração de Paginas HTML com RDFa embutido*. Os indivíduos são gerados com os mesmos procedimentos aplicados para a geração do RDFa. As triplas são geradas como um *dump* em um arquivo RDF.

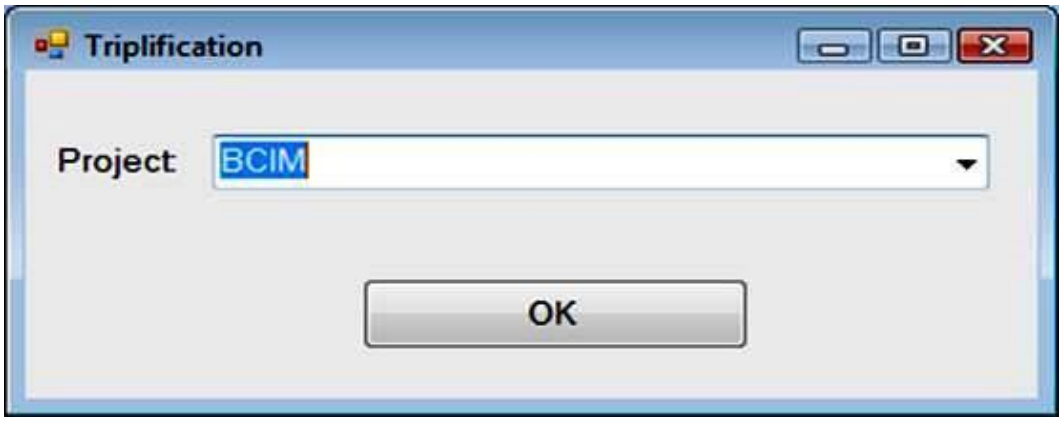


Figura 88 - Sub-módulo de triplificação dos dados das *views*

Apêndice D: Desvio automático para a Deep Web

O desvio automático para o respectivo formulário dinâmico da *Deep Web* tem como objetivo saltar as páginas geradas pela abordagem W-Ray e direcionar o usuário diretamente para a Deep Web. Desta forma as páginas W-Ray são acessadas apenas pelos mecanismos de busca ou por usuários que conhecem o endereço do site. O procedimento é executado da seguinte maneira:

1. Antes de uma página W-Ray ser carregada, é identificada a URL de onde partiu o acesso.
2. Se esta URL é proveniente do Google, ou seja, se o usuário fez uma consulta por palavra chave no mecanismo de busca Google e chegou numa página W-Ray através da página do Google, faz-se um desvio baseado nas palavras-chave identificadas na pesquisa via Google. Um exemplo de uma URL proveniente do mecanismo de busca Google pode ser observado a seguir. As palavras em vermelho compõem a query feita pelo usuário no mecanismo de busca Google. As palavras em azul indicam o desvio feito pelo mecanismo de busca Google para a página que contem as descrições geradas automaticamente pela aplicação W-Ray para o mapa de Climas do Brasil.

> http://www.google.com/url?sa=t&rct=j&q=climas+do+brasil+semi-arido&source=web&cd=38&cad=rja&ved=0CE4QFjAHOB4&url=http%3A%2F%2Ftomcat.inf.puc-rio.br%3A8080%2Fmuralmaps%2Fclima.html&ei=XvE5Ucn2FYPK9QS1j4HoBA

3. Identifica-se a sentença que possui o maior número de palavras-chave pesquisadas

4. Recupera-se href do sujeito da sentença e desvia-se automaticamente para o objeto correspondente da Deep Web. Caso haja empate, desvia-se para o objeto da primeira sentença localizada. No caso do exemplo acima o

desvio é feito automaticamente para o mapa de climas com um zoom na primeira região de clima semiárido encontrada nas descrições W-Ray.

Uma observação importante é que este procedimento só é válido para o protocolo http. Quando o protocolo é https, o mecanismo de busca Google não disponibiliza a query feita pelo usuário.

Apêndice E: Pseudocódigo GeraRDFa

```
Procedimento GeraRDFa

Entrada: FlorestaTemplates f, conexão BD-WRAY
Saída: String pagHTML[ ]
{
  Para cada arvoreTemplates a em f faça
    VarreArvore(a)
  Fim-para
}

/*varre a árvore que representa a hierarquia de templates das
sentenças que serão publicadas na página HTML com RDFa embutido */
Procedimento VarreArvore(a)
Entrada: arvoreTemplates a (cada árvore da floresta de
                templates representa um conjunto de
                sentenças que serão publidas em uma página
                HTML)
{
/*Declaração de arrays de estrutura */
Declaração:
   Estrutura: no
              Varchar pai
              Varchar filho
              Varchar codView
              Varchar tipoTemplate
   arrayNo[numNos]
Fim-declaração

  Se a.quebra então
    IniciaPaginaNova
  Fim-se
  no = raiz(a)
  TrataNo(no)
}
```

```
/*trata um nó da árvore de templates */

Procedimento TrataNo(no)

Entrada: Estrutura: no
                Varchar pai
                Varchar filho
                Varchar codView
                Varchar tipoTemplate
          arrayNo[numNos]


(um nó da árvore representa um template de uma sentença. A
 hierarquia da árvore define como as sentenças serão
 publicadas na página. Por exemplo: sentenças onde o sujeito
 é Brasil podem funcionar como título, estados como
 subtítulo e municípios como sentenças comuns. Cada nó da
 árvore está relacionado com uma view definida pelo designer
 para o projeto W-Ray)

{

/*Declaração de arrays de estrutura */
Declaração:
  Estrutura: properties
     Varchar nome
     Varchar valor
     Varchar hrefDatatypeProperty
     Varchar datatypeProperty
     Varchar tipoDatatypeProperty
  arrayProperties[numProperties]

  Estrutura: propertiesMulti
     Varchar nome
     String arrayValores[]
     Varchar hrefDatatypeProperty
     Varchar datatypeProperty
     Varchar tipoDatatypeProperty
  arrayPropertiesMulti[numProperties]

  Estrutura: predicadoNM
     Varchar nome
     Varchar uriPredicado
     Varchar classePredicado
     Varchar propertyPredicado
     Varchar tipoPredicado
     Varchar valorPredicado
     Varchar hrefDatatypeProperty
     Varchar hrefPagOrigemPredicado
  arrayPredicadoNM[numPredicado]

  Estrutura: predicadoNM
     Varchar nomePredicado
     Varchar uriPredicado
     Varchar classePredicado
     Varchar propertyPredicado
```

```
      Varchar tipoPredicado
      Varchar valorPredicado
      Varchar hrefDatatypeProperty
      Varchar hrefPagOrigemPredicado
  arrayPredicadoNM[numPredicado]

  Estrutura: RelMN
      Varchar verbo
      Varchar hrefVerbo
      Varchar objectPropertyVerbo
  arrayVerbos[numVerboss]

Fim-Decalracao


    Se no.tipoTemplate = nulo então
        tipoTemplate,viewSujeito=IdentificaTipoTemplate(no)
    senão
        tipoTemplate = no.tipoTemplate
    Fim-se
    Conforme tipoTemplate
      Caso Titulo
        titulo = templateTitulo(no)
        Imprime (titulo)


      Caso SubTitulo
        subTitulo = templateSubTitulo(no)
        Imprime (subTtitulo)


      Caso Simples
        uriSujeito, classeSujeito, hrefSujeito,
            propertySujeito, idiomaSujeito, valorSujeito,
            hrefDeepWebSujeito  =
            IdentificaAtributoSujeito(no,viewSujeito)
          verbo=
          IdentificaVerbo(viewSujeito,tipoTemplate,idiomaSu
          jeito)
          arrayProperties =
            IdentificaPredicado(viewSujeito,
            idiomaSujeito, uriSujeito)
          arrayAlteracoesDesigner = IdentificaAlteracoesTemplate(no)
          ImprimeTemplateSimples(
            uriSujeito, classeSujeito, hrefSujeito,
            propertySujeito, idiomaSujeito, valorSujeito,
            hrefDeepWebSujeito, verbo, arrayProperties,
            arrayAlteracoesDesigner)
```

```
    uriSujeito, classeSujeito, hrefSujeito,
      propertySujeito, idiomaSujeito, valorSujeito,
      hrefDeepWebSujeito  =
      IdentificaAtributoSujeito(no,viewSujeito)


Caso Multivalorado

  uriSujeito, classeSujeito, hrefSujeito,
      propertySujeito, idiomaSujeito, valorSujeito,
      hrefDeepWebSujeito  =
      IdentificaAtributoSujeito(no,viewSujeito)

    verbo=
    IdentificaVerbo(viewSujeito,tipoTemplate,idiomaSu
    jeito)

    arrayProperties= IdentificaPredicado(viewSujeito,
      idiomaSujeito, uriSujeito)

    arrayPropertiesMulti=
      IdentificaPredicadoMulti(viewSujeito,
      idiomaSujeito, uriSujeito)

    arrayAlteracoesDesigner= IdentificaAlteracoesTemplate(no)

    ImprimeTemplateMultivalorado(
      uriSujeito, classeSujeito, hrefSujeito,
      propertySujeito, idiomaSujeito, valorSujeito,
      hrefDeepWebSujeito, verbo, arrayProperties,
      arrayPropertiesMulti, arrayAlteracoesDesigner)

    uriSujeito, classeSujeito, hrefSujeito,
      propertySujeito, idiomaSujeito, valorSujeito,
      hrefDeepWebSujeito  =
      IdentificaAtributoSujeito(no,viewSujeito)


Caso Binario

  uriSujeito, classeSujeito, hrefSujeito,
     propertySujeito, idiomaSujeito, valorSujeito,
     hrefPagOrigemSujeito =
     IdentificaAtributoSujeitoMN(no,viewSujeito)

   verbo, hrefVerbo, objectPropertyVerbo =
      IdentificaVerbo(viewSujeito, tipoTemplate,
      idiomaSujeito)

   arrayPredicadoNM, numPredicado =
      IdentificaPredicadoMN(no, viewSujeito,
      uriSujeito)

   arrayAlteracoesDesigner = IdentificaAlteracoesTemplate(no)

   ImprimeTemplateBinario(            uriSujeito,
      classeSujeito, hrefSujeito, propertySujeito,
      idiomaSujeito, valorSujeito,
      hrefPagOrigemSujeito, verbo, hrefVerbo,
      objectPropertyVerbo, arrayPredicadoNM,
      arrayAlteracoesDesigner)
```

```
      uriSujeito, classeSujeito, hrefSujeito,
         propertySujeito, idiomaSujeito, valorSujeito,
         hrefPagOrigemSujeito =
         IdentificaAtributoSujeitoMN(no,viewSujeito)


   Caso n-ario

     uriSujeito, classeSujeito, hrefSujeito,
        propertySujeito, idiomaSujeito, valorSujeito,
        hrefPagOrigemSujeito =
        IdentificaAtributoSujeitoMN(no,viewSujeito)

      verbo, hrefVerbo, objectPropertyVerbo =
         IdentificaVerbo(viewSujeito,tipoTemplate,idioma
         Sujeito)

      arrayAlteracoesDesigner =IdentificaAlteracoesTemplate(no)

      arrayAtributosRelMN =
         IdentificaAtributosRelMN(viewSujeito,
         idiomaSujeito, uriSujeito)

      ImprimePrimeiraparteTemplateNario(  uriSujeito,
         classeSujeito, hrefSujeito, propertySujeito,
         idiomaSujeito, valorSujeito,
         hrefPagOrigemSujeito, verbo, hrefVerbo,
         objectPropertyVerbo, arrayAtributosRelMN,
         arrayAlteracoesDesigner)

      arrayVerbos=
         IdentificaOutrosRelacionamentos(viewSujeito)

      Para cada verbo do arrayVerbos faça

          uriPredicado, classePredicado, hrefPredicado,
            propertyPredicado, idiomaPredicado,
            tipoPredicado, valorPredicado,
            hrefPagOrigemPredicad =
            IdentificaPredicadoMN(no, verbo, uriSujeito)

          ImprimePredicadosNM(
            arrayVerbos(verbo), uriPredicado,
            classePredicado, hrefPredicado,
            propertyPredicado, idiomaPredicado,
            tipoPredicado, valorPredicado,
            hrefPagOrigemPredicado,
            arrayAlteracoesDesigner)

      Fim-Para

      uriSujeito, classeSujeito, hrefSujeito,
         propertySujeito, idiomaSujeito, valorSujeito,
         hrefPagOrigemSujeito =
         IdentificaAtributoSujeitoMN(no,viewSujeito)
Fim-Conforme


Para cada no=ProximoFilho(no)
  TrataNo(no)
```

```
    Fim-para
  metadados = templateMetadados(no)
  Imprime (metadados)
}


/*identifica o tipo de template */
Procedimento IdentificaTipoTemplate(no)
Entrada: no
Saída: String tipoTemplate, viewSujeito
{
No BD-WRAY:
     view = RecuperaPrimeiraView(no)
     chavePrimaria = IdentificaChavePrimaria(view)
     tipoChavePrimaria=IdentificaTipoChavePrimaria(chavePrimaria,
     view)

     Conforme tipoChavePrimaria
       Caso compostaPorDuasEstrangeiras
       /*chave composta por chaves primárias de apenas duas views/*
         viewSujeito = DescobreViewSujeito(view) /*descobre se
         o designer definiu um sujeito em alguma view dos relacionamentos /*
         Se viewSujeito = nenhum então
           ArbitrarViewSujeito(view, tipoChavePrimaria)
         Senão
           tipoTemplate = "binario"
         Fim-se

       Caso compostaPorMaisDeDuasEstrangeiras
       /*chave composta por chaves primárias de mais de duas views/*
         viewSujeito = DescobreViewSujeito(view)/*descobre se o
         designer definiu um sujeito em alguma view dos relacionamentos/*
         Se viewSujeito = nenhum então
           ArbitrarViewSujeito(view, tipoChavePrimaria)
         Senão
           tipoTemplate = "n-ario"
         Fim-se

       Caso parcialmenteComposta
```

```
        /*chave composta por chaves primárias de outras views, mas possui
        atributos da própria view/*
          viewSujeito = DescobreViewSujeito(view)/*descobre se o
          designer definiu um sujeito em alguma view dos relacionamentos/*
          Se viewSujeito = nenhum então
            viewSujeito = view
            ArbitrarAtributoSujeito(viewSujeito)
          Senão
            tipoTemplate = "multivalorado"
          Fim-se

        Caso simples
        /*chave composta por atributos da própria view/*
          viewSujeito= DescobreViewSujeito(view)/*descobre se o
          designer definiu algum atributo da view como sujeito/*
          participaRel,arrayViewsRel                              =
          ParticipaRelacionamento(view) /* verifica se a view
          participa de algum relacionamento nm, 1n /*
          Se participaRel = nao e viewSujeito <> nenhum então
            tipoTemplate = "simples"
          senão
            Se participaRel = nao e viewSujeito = nenhum então
              tipoTemplate = "simples"
              viewSujeito = view
              ArbitrarAtributoSujeito(viewSujeito)
            Senão
              Se participaRel = sim e viewSujeito=nenhum então
                 ArbitrarViewSujeito(view,tipoChavePrimaria)
              Fim-se
            Fim-se
          Fim-se
      Fim-Conforme
}


Procedimento templateTitulo(no)
Entrada: no
Saída: String titulo
{
     titulo = Recupera Título no BD-WRay
     Se titulo = nenhum então
```

```
      Montar título com o nome do projeto
    Fim-se
}

Procedimento templateSubTitulo(no)
Entrada: no
Saída: String subTitulo
{
    subTitulo = Recupera subTitulo no BD W-Ray
    Se subTitulo = nenhum então
      Montar subTitulo com o Título concatenando com o nome
      da primeira view recuperada para o nó raiz.
    Fim-se
}

/*identifica o atributo que funcionará como sujeito da sentença */
Procedimento IdentificaAtributoSujeito (no)
Entrada: no, viewSujeito
Saída: uriSujeito, classeSujeito, hrefSujeito, propertySujeito,
idiomaSujeito, valorSujeito, hrefDeepWebSujeito
{
    Seleciona a viewSujeito no BD W-Ray e verifica se o
    usuário definiu um atributo como sujeito.
    Se existir atributo definido como sujeito então
      Recupere no BD: o valor da chave primária da
      viewSujeito, nome do atributo, a datatypeProperty
      correspondente ao atributo sujeito (propertySujeito), a
      classe da datatypeProperty (classeSujeito), o idioma
      do dado (idiomaSujeito), o valor do atributo sujeito
      (valorSujeito), link da Ontologia Gerada e a parte
      inicial do link que liga a sentença à Deep Web.
      Inserir hífem no nome do atributo (se separado por
      branco). Inserir "#" na frente do nome do atributo
      (hrefSujeito). O hrefSujeito liga do nome do atributo
      Sujeito com a sua descrição no final da página.
      Concatenar ao link da Ontologia Gerada (armazenado no
      BD W-Ray) o valor da chave primária do sujeito para
      gerar a URI do sujeito (uriSujeito).
      Concatenar ao link inical da Deep Web do projeto
      (amarzenado no BD W-Ray) o valor da chave primária do
      sujeito (hrefDeepWebSujeito).
    Senão
```

```
        atributoSujeito= ArbitrarAtributoSujeito(viewSujeito)

        Recupere no BD: o valor da chave primária da
        viewSujeito, nome do atributo, a datatypeProperty
        correspondente ao atributo sujeito (propertySujeito), a
        classe da datatypeProperty (classeSujeito), o idioma
        do dado (idiomaSujeito), o valor do atributo sujeito
        (valorSujeito), link da Ontologia Gerada e a parte
        inicial do link que liga a sentença à Deep Web.

        Inserir hífem no nome do atributo (se separado por
        branco). Inserir "#" na frente do nome do atributo
        (hrefSujeito). O hrefSujeito liga do nome do atributo
        Sujeito com a sua descrição no final da página.

        Concatenar ao link da Ontologia Gerada (armazenado no
        BD W-Ray) o valor da chave primária do sujeito para
        gerar a URI do sujeito (uriSujeito).

        Concatenar ao link inical da Deep Web do projeto
        (amarzenado no BD W-Ray) o valor da chave primária do
        sujeito (hrefDeepWebSujeito).

    Fim-se

}


/*elege um atributo que funcionará como sujeito da sentença quando
o desiner não define o sujeito explicitamente*/

Procedimento ArbitrarAtributoSujeito(viewSujeito)

Entrada: viewSujeito

Saída: String atributoSujeito

{

    Recuperar primeiro atributo string depois da chave
    primária e defini-lo como sujeito no BD W-Ray
    (atributoSujeito).

}



/*elege uma view como a view que possui o atributo que funcionará
como sujeito da sentença quando o desiner não define o sujeito
explicitamente*/

Procedimento ArbitrarViewSujeito(view, tipoChavePrimaria)

Entrada: view, tipoChavePrimaria

Saída: String viewSujeito, atributoSujeito

{

     Conforme tipoChavePrimaria

       Caso          compostaPorDuasEstrangeiras          ou
       compostaPorMaisDeDuasEstrangeiras

       /*chave composta por chaves primárias de apenas duas views/*
```

```
            Selecionar primeira chave estrangeira que
            participa da chave primaria.
            Descobrir view referente à primeira chave
            estrangeira.
            viewSujeito = view
            atributoSujeito=
            ArbitrarAtributoSujeito(view)
          Caso simples ou parcialmenteComposta
          /*chave composta por atributos da própria view/*
            participaRel,arrayViewsRel=
            ParticipaRelacionamento(view)
            Se participaRel então
                 Para cada view do arrayViewsRel faça
                   recupere a chave primária da view corrente
                   Se a chave primária for totalmente
                   composta por chaves estrangeiras então
                     view = arrayViewsRel.corrente
                     tipoChavePrimaria=
                       "compostaPorMaisDeDuasEstrangeiras"
                     ArbitrarViewSujeito(view,tipoChavePrimaria)
                   Fim-se
                 Fim-Para
            Senão
                 viewSujeito = view
                 atributoSujeito=
                 ArbitrarAtributoSujeito(view)
            Fim-se
       Fim-Conforme
}

/*descobre qual view possui um atributo que será o sujeito da
sentença */
Procedimento DescobreViewSujeito(view, tipoChavePrimaria)
Entrada: view, tipoChavePrimaria
Saída: String viewSujeito
{
    Recupera view no BD W-Ray e verifica se possui um
    atributo definido como sujeito.
     Conforme tipoChavePrimaria
```

```
        Caso          compostaPorDuasEstrangeiras          ou
        compostaPorMaisDeDuasEstrangeiras
        /*chave composta por chaves primárias de apenas duas views/*
          Recuperar views que participam do relacionamento
          Para cada view faça
             Verifica se existe atributo sujeito
             Se exite atributo então
                viewSujeito = view
             Fim-se
          Fim-para
          Se nenhuma view tiver um atributo sujeito então
             viewSujeito = "nenhum"
          Fim-se
        Caso simples ou parcialmenteComposta
        /*chave composta por atributos da própria view */
          Verificar se view possui um atributo definido como
          sujeito
          Se exite atributo então
             viewSujeito = view
          Senão
             participaRel,arrayViewsRel=
             ParticipaRelacionamento(view)
             Se participaRel então
                  Verificar se alguma view do arrayViewsRel
                  possui um atributo definido como sujeito.
                  Se exite alguma view atributo então
                    viewSujeito = view
                  Senão
                    viewSujeito = "nenhum"
                  Fim-se
          Fim-se
      Fim-Conforme
}


/*verifica se a view participa de algum relacionamento nm, 1n */
Procedimento ParticipaRelacionamento(view)
Entrada: view
Saída: String participaRel, arrayViewsRel
```

```
{
    Recuperar chave primária da view
    Verificar se a chave primária compõe a chave primária
    de uma view cuja chave primária é composta por chaves
    estrangeiras de duas ou mais views.
    Se sim então
       participaRel = true
       Para cada chave estrangeira da view-relacionamento
       faça
          Monta arrayViewsRel com todas as views envolvidas
          no relacionamento sinalizando a view que
          representa o relacionamento
       Fim-para
    Fim-se
}
```

```
/*identifica qual o verbo da sentença e recupera a objectProperty
referente ao verbo e o link para a descrição da objectProperty na
própria página de publicação */
Procedimento IdentificaVerbo(viewSujeito, tipoTemplate,
idiomaSujeito)
Entrada: viewSujeito, tipoTemplate, idiomaSujeito
Saída: verbo, objectProperty, hrefSObjectPropery
{
    Conforme tipoTemplate
      Caso Simples ou Multivalorado
         verbo = default (conforme idiomaSujeito)
      Caso Binario ou n-ario
         participaRel,arrayViewsRel=
         ParticipaRelacionamento(viewSujeito)
         Recupere no arrayViewsRel a view responsável pelo
         relacionamento
         Recupere no BD W-Ray:
           O nome da view responsável pelo relacionamento
           (verbo) no idiomaSujeito ou recupere o novo nome
           da view atribuído pelo designer;
           A objectProperty correspondente, o link da
           Ontologia Gerada.
         Insira hífem no nome da view (se separado por
         branco). Inserir "#" na frente do nome da view
         (hrefSujeito). /* O hrefSujeito liga o verbo
```

```
            (objectProperty) com a sua descrição no final da
            página */
        Fim-Conforme
}


/*identifica qual o predicado da sentença, ou seja, todas as
datatype properties para um template Simples */
Procedimento IdentificaPredicado(viewSujeito, idiomaSujeito,
uriSujeito)
Entrada: viewSujeito, idiomaSujeito, uriSujeito
Saída: arrayProperties
/*Declaração de arrays de estrutura */
  Estrutura: properties
      Varchar nome
      Varchar valor
      Varchar href
      Varchar datatypeProperty
      Varchar tipoDatatypeProperty
  arrayProperties[numProperties]
{
    Recupere em idiomaSujeito no BD W-Ray todos os nomes de
    atributos (nome) da viewSujeito em português, valores
    de cada atributo (valor), nome da datatypeProperty e
    tipo da datatypeProperty.
    Insira hífem no nome de cada atributo (se nome separado
    por branco). Insera "#" na frente do nome de cada
    atributo (href). /* O href liga a dataytpeProperty com
    a sua descrição no final da página */
    Preencha o array arrayProperties.
}


/*identifica qual o predicado da sentença composto por atributos
multivalorados, ou seja, aqueles em que estão em outra view devido
à normalização imposta pelo modelo de BD relacional. Recupera
também as datatype properties */
Procedimento IdentificaPredicadoMulti(viewSujeito,
idiomaSujeito, uriSujeito)
Entrada: viewSujeito, idiomaSujeito, uriSujeito
Saída: arrayPropertiesMulti
/*Declaração de arrays de estrutura */
  Estrutura: propertiesMulti
      Varchar nome
      arrayValores[]
      Varchar hrefDataTypeproperty
```

```
     Varchar datatypeProperty
     Varchar tipoDatatypeProperty
  arrayPropertiesMulti[numProperties]
{
    Recupere em idiomaSujeito no BD W-Ray as chaves
    estrangeiras da viewSujeito. Nas views em que cada
    chave estrangeira é chave primária recupere o nome e
    valores do primeiro atributo depois da chave primária
    em português. Recupere também o nome das
    datatypeProperties e seu tipo.

    Insira hífem no nome de cada atributo (se nome separado
    por branco). Insera "#" na frente do nome de cada
    atributo (href). /* O href liga a dataytpeProperty com
    a sua descrição no final da página */

    Preencha o array arrayPropertiesMulti.
}
```

```
/*Identifica qual o predicado da sentença cujo template é binário.
Neste caso, a tabela que representa o relacionamento não possui
atributos e relaciona apenas duas tabelas. Este procedimento é o
responsável por recuperar tudo relativo à segunda tabela do
relacionamento, ou seja, aquela que não é sujeito */

Procedimento IdentificaPredicadoMN(viewSujeito,
idiomaSujeito, uriSujeito)

Entrada: viewSujeito, idiomaSujeito, uriSujeito

Saída: arrayPredicadoNM

/*Declaração de arrays de estrutura */
  Estrutura: predicadoNM
     Varchar nomePredicado
     Varchar uriPredicado
     Varchar classePredicado
     Varchar propertyPredicado
     Varchar tipoPredicado
     Varchar valorPredicado
     Varchar hrefDataTypeproperty
     Varchar hrefPagOrigemPredicado
  arrayPredicadoNM[numPredicado]
{
    participaRel,arrayViewsRel=
    ParticipaRelacionamento(viewSujeito)

    Recupere no arrayViewsRel a view que não é responsável
    pelo relacionamento e que não é sujeito

    Recupere em idiomaSujeito no BD W-Ray:

        o valor da chave primária da view, nomes dos
        atributos (nomePredicado) e datatypeProperties
        correspondentes a cada atributo (propertyPredicado)
        com seus tipos (tipoPredicado), a classe das
```

```
            datatypeProperties (classePredicado), os valores
            dos atributos (valorPredicado), link da Ontologia
            Gerada.

        Inserir hífem no nome de cada atributo (se nome
        separado por branco). Inserir "#" na frente do nome do
        atributo (hrefDataTypeproperty).

        Concatenar ao link da Ontologia Gerada (armazenado no
        BD W-Ray) ao valor da chave primária para gerar a URI
        do predicado (uriPredicado).

        Concatenar o nome da tabela sujeito (sem branco ou
        hífem) com ".html#" e ao valor da chave primária do
        predicado (hrefPagOrigemPredicado).

        Preencha o array arrayPredicadoNM.

}



/*Identifica atributos relativos ao relacionamento, ou seja, aos
atributos da tabela que representa o relacionamento */

Procedimento IdentificaAtributosRelMN(viewSujeito,
idiomaSujeito, uriSujeito)

Entrada: viewSujeito, idiomaSujeito, uriSujeito

Saída: arrayAtributosRelMN

/*Declaração de arrays de estrutura */
  Estrutura: atributosRelMN
     Varchar uriRel
     Varchar classeRel
     Varchar nomeRel
     Varchar hrefRel
     Varchar propertyRel
     Varchar tipoRel
     Varchar valorRel
  arrayAtributosRelMN[numAtributos]

{

    participaRel,arrayViewsRel=
    ParticipaRelacionamento(viewSujeito)

    Recupere no arrayViewsRel a view que é responsável pelo
    relacionamento

    Recupere em idiomaSujeito no BD W-Ray:

        os nomes dos atributos (nomeRel) e
        datatypeProperties correspondentes a cada atributo
        (propertyRel) com seus tipos (tipoRel), a classe
        das datatypeProperties (classeRel), os valores dos
        atributos (valorRel), link da Ontologia Gerada.

    Inserir hífem no nome de cada atributo (se nome
    separado por branco). Inserir "#" na frente do nome do
    atributo (hrefRel).
```

```
    Concatenar ao link da Ontologia Gerada (armazenado no
    BD W-Ray) ao valor da chave primária para gerar a URI
    do predicado (uriRel).

    Preencha o array arrayAtributosRelMN.

}


/*Gera um array com os verbos diferentes do principal (logo após o
sujeito) de um template n-ário*/

Procedimento    IdentificaOutrosRelacionamentos(viewSujeito,
idiomaSujeito)

Entrada: viewSujeito, idiomaSujeito

Saída: arrayVerbos

/*Declaração de arrays de estrutura */
  Estrutura: RelMN
     Varchar verbo
     Varchar hrefVerbo
     Varchar objectPropertyVerbo
  arrayVerbos[numVerbos]

{

    participaRel,arrayViewsRel=
    ParticipaRelacionamento(viewSujeito)

    Recupere a view que é resonsável pelo relacionamento

    Recupere no arrayViewsRel as views que não são
    responsáveis pelo relacionamento e que não são sujeito

    Para cada view não responsável pelo relacionamento
    faça:

        Recupere em idiomaSujeito no BD W-Ray:

        o nome da chave-secundária que relaciona a view com
        a view responsável pelo relacionamento (verbo)

        recupere a objectProperty correspondente a este
        relacionamento (objectPropertyVerbo)

        Inserir hífem no nome de cada chave-secundária (se
        nome separado por branco). Inserir "#" na frente do
        nome do atributo (hrefVerbo)

    Fim-para

    Preencha o array arrayVerbos.

}


Procedimento      IdentificaTipoChavePrimaria(chavePrimaria,
view)

Entrada: chavePrimaria, view
```

```
Saída: String tipoChavePrimaria
{
No BD-WRAY:
      Recupere os relacionamentos da view
      Conforme composição da chave primária
        Caso seja composta por chaves primárias de apenas
        duas views
           tipoChavePrimaria = "compostaPorDuasEstrangeiras"

        Caso seja composta por chaves primárias de mais de
        duas views então
           tipoChavePrimaria=
           "compostaPorMaisDeDuasEstrangeiras"

        Caso seja composta por chaves primárias de outras
        views, mas possui atributos da própria view
           tipoChavePrimaria= "parcialmenteComposta"

        Caso seja chave composta por atributos da própria
        view
           tipoChavePrimaria= "simples"

      Fim-Conforme
}

Procedimento IdentificaAlteraçõesTemplate(no)
Entrada: no
Saída: arrayAlteracoesDesigner[]
{
No BD-WRAY:
      Recupere na Tabela Template o projeto feito pelo
      designer
      Se existirem alterações no template default então
         Popular o arrayAlteracoesDesigner com as alterações
      Fim-se
}
```