UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

RENAN FRANCISCO SANTOS SOUZA

PROCESSO DE PUBLICAÇÃO DE DADOS ABERTOS INTERLIGADOS: APLICAÇÃO A DADOS DE DESEMPENHO DE REDE

Profa. Maria Luiza Machado Campos, Ph. D.
Orientadora

Rio de Janeiro, Dezembro de 2013

Processo de Publicação de Dados Abertos Interligados: Aplicação a Dados de Desempenho de Rede

Renan Francisco Santos Souza

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto de Matemática da Universidade Federal do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Apresentado por:

_____

Renan Francisco Santos Souza

Aprovado por:

_____

Profa. Maria Luiza Machado Campos, Ph.D.
(Orientadora)

_____

Profa. Adriana Santarosa Vivacqua, D.Sc.

_____

Profa. Jonice de Oliveira Sampaio, D.Sc.

_____

Prof. Herli Joaquim de Menezes, M.Sc.

Rio de Janeiro, Dezembro de 2013

AGRADECIMENTOS

Meus sinceros agradecimentos vão a cada um que esteve ao meu lado fisicamente ou não, diretamente ou não, durante toda a minha caminhada desde a época de ensino fundamental até os últimos dias da graduação.

Em especial, gostaria de primeiramente agradecer a Deus por certamente estar ao meu lado em todos os momentos, principalmente por me dar paz quando mais precisei.

Agradeço a toda minha família que sempre me apoiou e acreditou em mim. Particularmente, sou eternamente grato a minha mãe por nunca ter medido esforços em investir da melhor maneira possível em minha educação. Graças a suas sábias palavras e principalmente seu esforço em me educar de modo excelente, consegui chegar até o final de um curso de nível superior de destacada qualidade.

Também agradeço a minha namorada, futura esposa, que desde antes de entrar na universidade sempre foi minha motivação maior. Sou grato por sua dedicação, compreensão, por torcer por mim e acreditar que eu seria capaz durante todo o tempo.

Sou grato a todos os meus amigos de infância, de escola, do técnico, da igreja e do intercâmbio. Especialmente, agradeço aos amigos da faculdade, que também foram essenciais tanto para estudar juntos quanto para diversão durante os árduos períodos da graduação.

Agradeço a minha professora Maria Luiza Machado Campos que me orienta desde os primeiros períodos da graduação. Cada palavra, cada gesto, cada tempo particular investido depois das aulas para me esclarecer algum tópico da matéria. Serei eternamente grato a toda sua dedicação e investimento em minha carreira profissional.

Meus agradecimentos também vão para os professores Les Cottrell e Bebo White, da Universidade de Stanford, que me receberam, apoiaram, orientaram, ensinaram e proveram tudo que puderam para eu desenvolver um bom trabalho no SLAC.

Finalmente, gostaria de agradecer a cada um dos meus professores, individualmente, desde os do ensino fundamental até os da graduação, que dedicadamente passaram a mim e aos meus colegas seus conhecimentos e que creio que contribuíram para um futuro melhor para a sociedade. Em especial, agradeço a todos meus professores do Departamento de Ciência da Computação da UFRJ.

A todos esses, eu dedico o meu esforço e trabalho.

RESUMO

SOUZA, Renan Francisco S. **Processo de Publicação de Dados Abertos Interligados: Aplicação a Dados de Desempenho de Rede**. Rio de Janeiro, 2013. Trabalho de Conclusão de Curso (Bacharel em Ciência da Computação) – Departamento de Ciência da Computação, Instituto de Matemática, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2013.

Atualmente, nas áreas das ciências da Computação e Informação, há um claro reconhecimento do potencial das tecnologias de Web Semântica, embora ainda seja notória a falta de aplicações que evidenciem os benefícios trazidos por sua utilização. Este trabalho tem por objetivo discutir o uso dessas tecnologias, apresentando uma proposta de Processo de Publicação de Dados Abertos Interligados (*Linked Open Data* - LOD), elaborada a partir do desenvolvimento de uma solução para disponibilizar publicamente dados sobre desempenho de ligações de nós na Internet ao redor do mundo, através de um formato padrão na web (LOD). As principais características e vantagens de se ter dados nesse formato serão distintamente enfatizadas no processo, bem como as dificuldades encontradas durante o projeto, evidenciando áreas da Web Semântica que ainda necessitam de mais pesquisa e desenvolvimento.

Palavras-chave: Linked Open Data, Dados Ligados Abertos, Padrões de Dados Abertos, Processo de Publicação de Linked Open Data, Web Semântica, LOD Medida de Monitoramento de Rede, LOD Ping Rede, PingER LOD

ABSTRACT

SOUZA, Renan Francisco S. **Linked Open Data Publication Process: Application in Networking Performance Measurement Data**. Rio de Janeiro, 2013. Undergraduate Conclusion Course Work (Bachelor of Computer Science) – Computer Science Department, Mathematics Institute, Federal University of Rio de Janeiro, Rio de Janeiro, 2013.

In today's time, in Computer and Information sciences, there is a clear acknowledgement of the semantic web technologies' potential, although it is still noticeable a lack of applications that take advantages of their utilization. The goal of this project is to discuss the usage of these technologies, presenting a proposal of a Linked Open Data (LOD) Publishing Process, conceived throughout the development of a solution to publish data about the performance of Internet links around the world, using a standard format on the web (LOD). The main characteristics and advantages of having data in this format will be distinctly emphasized in the process, as well as the difficulties faced during the project, highlighting fields of semantic web that still need more research and development.

Keywords: Linked Open Data, Open Data Standards, Linked Open Data Publication Process, Semantic Web, LOD Networking Monitoring Measurement, LOD Ping Networking, PingER LOD

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
BD	Banco de Dados
CERN	*The Europeran Organization for Nuclear Research*
CRUD	*Create, Read, Update, Delete*
DW	*Data Warehouse*
ETC	Extração, Transformação e Carga
ETSI	*European Telecommunications Standards Institute*
GRECO	Grupo de Engenharia do Conhecimento
HTML	*HypterText Markup Language*
HTTP	*HyperText Transfer Protocol*
IEPM	*Internet End-to-end Performance Monitoring*
ISG	*Industry Specification Group*
LOD	*Linked Open Data*
MOI	*Measurement Ontology for IP traffic*
MOMENT	*Monitoring and Measurement in the Next generation Technologies*
OWL	*Web Ontology Language*
PIB	Produto Interno Bruto
PingER	*Ping End-to-end Reporting*
RDF	*Resource Description Framework*
RDFS	RDF *Schema*
SGBD	Sistema de Gerenciamento de Banco de Dados
SLAC	*Stanford National Accelerator Laboratory*
SPARQL	*SPARQL Protocol and RDF Query Language*
SQL	*Structured Query Language*
UFRJ	Universidade Federal do Rio de Janeiro
URI	*Unified Resource Identifier*
URL	*Unified Resource Locator*
VOID	*Vocabulary of Interlinked Datasets*
WWW	*World Wide Web*
XML	*Extensible Markup Language*
XSD	XML *Schema Definition*

SUMÁRIO

1 INTRODUÇÃO

1.1 O problema da Web atual

Nos dias de hoje, apesar de muitos avanços, ainda se gasta muito tempo tentando encontrar exatamente o que se busca na web. David Siegel (2010) exemplifica que pesquisa e marketing estão intimamente relacionados. No setor de marketing, além de serviço e qualidade de produtos, a pesquisa e o fato de encontrar o produto pesquisado também são essenciais. Marketing é apenas um dos muitos setores da sociedade que utilizam e dependem da pesquisa na web. Em torno de 25% do tempo de expediente de um profissional do conhecimento são gastos pesquisando informações essenciais ao seu trabalho (SIEGEL, 2010).

Os americanos fazem cerca de 7,5 bilhões de pesquisas nos principais mecanismos de busca (por exemplo, Google e Bing) por mês e cerca de um terço, ou seja, 2,48 bilhões de pesquisas não trazem resultado esperado (SIEGEL, 2010). Qualquer um que utilize a Internet hoje em dia já se deparou com alguma pesquisa que ou não retorna nada ou que necessite de outras pesquisas além de cliques em vários outros *links* até encontrar o que se busca. O problema se intensifica se for levado em consideração que muitos dos *websites* espalhados pela Internet possuem um campo de "busca". Em outras palavras, as buscas estão por todo lado, não se restringindo ao Google ou Bing.

Adicionando-se ao fato das pesquisas não retornarem algum resultado, ainda existe o contratempo ocasionado porque os mecanismos de busca, em geral, desconsideram a semântica do que está sendo buscado. Siegel (2010) explica, ainda, o conceito de resultado falso positivo nas buscas: "é quando aparece uma resposta que não deveria estar lá." Isso fica evidenciado quando se busca por palavras para as quais existam homônimos. Por exemplo, se alguém busca por `países do globo`, sendo `globo` um sinônimo para mundo, e os primeiros resultados apresentam *links* para países em que a TV Globo atua. Isso poderia ser resolvido se os mecanismos de busca levassem em conta a semântica das palavras buscadas para distinguir os resultados e apresentar exatamente o que atende ao interesse do usuário.

Ademais, outro problema da web atual é a dificuldade dos mecanismos de busca têm de lidar com muitas palavras-chave nas pesquisas. Experimentalmente, alguém que tente realizar uma única pesquisa com vários parâmetros normalmente não consegue chegar ao resultado esperado. Isso se evidencia quando se busca, por exemplo, algo bem específico e preciso: `quais são os endereços de farmácias próximas onde posso encontrar agora paracetamol 750 mg a menos de R$ 3,50?`. Nessa pesquisa, o mecanismo de

busca precisaria entender o que é `próximo`, `agora`, `menos de`, `R$`, `mg`, `paracetamol` e `endereços de farmácia`. Nota-se a diversidade de natureza de informação e a semântica por detrás de cada palavra-chave tornando a pesquisa extremamente complexa para um mecanismo deste tipo.

É bem difícil precisar quantos *websites* existem, porém sabe-se que são muitos. Alpert e Hajaj (2008) informaram no blog oficial do Google que 1 trilhão de URLs únicas eram processadas de uma só vez pelo mecanismo de busca. Ou seja, é razoável afirmar que a informação necessária para responder precisamente à pergunta exemplificada acima já existe, de alguma forma, na web. Todavia, considerando as tecnologias em amplo uso hoje, recuperar precisamente a informação necessária a uma busca mais complexa ainda é extremamente difícil.

Entretanto, de modo a amenizar todos esses problemas, para algum tipo de mecanismo de busca ser capaz de responder precisamente a uma pergunta tão específica, a informação necessária para trazer o resultado esperado precisa idealmente ser qualificada de três formas:

a) Organizada. Analogamente ao que acontece no cenário da computação, procurar algum objeto em um ambiente bem estruturado e organizado costuma ser muito mais rápido e eficiente do que procurar o mesmo objeto em um ambiente completamente desorganizado. Adicionalmente, a semântica, ou seja, o significado de cada informação poderia ser utilizado para auxiliar a organização de toda a informação espalhada ao redor da web. Em outras palavras, se a informação na web possuir uma melhor organização, além de humanos, os mecanismos de busca especialmente terão mais facilidade de encontrar e retornar o resultado esperado.
b) Conectada. Pela natureza extremamente diversificada da web, é impraticável ter toda a informação mantida em um único lugar, em especial, um único banco de dados, isolado de outras fontes de dados. Os bancos de dados precisam estar interconectados de alguma forma, de modo que, ao perguntar algo tão específico, seja possível buscar a informação em vários bancos diferentes ao mesmo tempo maximizando, assim, o domínio da busca. Essa ideia apoia a possibilidade de existirem vários bancos de dados tão conectados e integrados que poderiam ser tratados como um único banco de dados gigante.
c) Aberta. Algumas vezes a informação até está bem organizada e estruturada, porém não é aberta; fica retida em bancos de dados privados das organizações. É necessário que a informação na web seja pública e de fácil acesso. Essa última

> qualidade alude à ideia de web colaborativa e pública, em prol da sociedade como um todo. Porém, essa é uma das maiores discussões filosóficas neste ramo uma vez que as empresas privadas, muitas vezes, não têm interesse em publicar suas informações, sendo um empecilho para a evolução mais rápida dessas ideias.

Portanto, atualmente, os mecanismos de busca são definitivamente importantes chegando a ser necessários no dia-a-dia de qualquer pessoa conectada à Internet. Pelos problemas citados, fica evidente, no entanto, que as buscas precisam ser melhoradas significativamente. A Web Semântica, ainda que esteja muito longe de realmente resolver, é uma proposta que traz uma alternativa para solucionar as dificuldades mencionadas. Adicionalmente, como parte dessa proposta promissora, o mundo da Web Semântica claramente considera as três qualidades de informação citadas acima.

1.2 A Web Semântica

Tim Berners-Lee (1998), conhecido como inventor da web por ter sido um dos autores do projeto do qual se originou a *World Wide Web* (BERNERS-LEE; CAILLIAU, 1990) propôs um projeto que enfatiza a necessidade da informação na web ser compreensível por máquinas e nomeou-o de Web Semântica. Lee explica que a web foi designada para ser um espaço de informação útil não apenas para comunicação humano-humano, mas também para que máquinas fossem capazes de interpretar, participar e colaborar. Idealmente, a web Semântica seria uma rede de dados (em Inglês, *Web of Data*), promovendo o uso de padrões comuns e incentivando sua aplicação de modo a transformar toda a informação contida na web, predominantemente não estruturada, em um grande banco de dados global, com suas características de organização e descrição das informações ali mantidas (BERNERS-LEE, 1998).

Lee é o diretor do World Wide Web Consortium (W3C), organização fundada em 1994 para fomentar padrões e tecnologias interoperáveis, através da publicação de especificações, guias, softwares e ferramentas para a web. A missão amplamente declarada do W3C é levar a web ao seu "potencial completo" (BERNERS-LEE, 2013). Especialmente, o W3C formaliza recomendações, conhecidas como "Recomendação do W3C" *("W3C Recommendation")*, o nível mais maduro, entre vários outros, nos estágios de desenvolvimento de padrões para a web. Padrões que recebem a marca de "Recomendação do W3C" são mais respeitados porque são aprovados pelos membros do W3C e seu diretor após extensas discussões e construções de consenso entre os membros internos do consórcio e

também entre os membros e a comunidade web (WORLD WIDE WEB CONSORTIUM, 2005).

A Web Semântica e suas tecnologias são recomendações do W3C e é a visão que o Consórcio tem de web de dados interligados. De acordo com o W3C, as tecnologias de Web Semântica possibilitam que as pessoas criem bancos de dados na web, construa vocabulários e escreva regras para lidar com os dados (WORLD WIDE WEB CONSORTIUM, 2005).

A ideologia da Web Semântica inspirou muitas pessoas a se engajarem e pesquisarem essa área. Atualmente, a comunidade é bem forte e todos os dias são desenvolvidas novas técnicas, tecnologias e aplicações que contribuem para a maximização do potencial da web, que é a missão do W3C (SEMANTIC WEB, 2013).

Entretanto, apesar de todos os esforços principalmente vindos dos acadêmicos, ainda há um longo caminho até chegar à ideologia proposta por Berners-Lee e o W3C. Lee, Shadbolt e Hall (2006) afirmaram naquele ano: "esta ideia simples continua amplamente não realizada". Atualmente, sete anos depois, esta afirmação ainda é pertinente. O autor da página principal da comunidade de Web Semântica a descreve como uma "visão utópica" (SEMANTIC WEB, 2013), evidenciando o quão longe da realidade atual a maioria das propostas ainda se encontra. Isso evidencia o quanto essas ideias ainda precisam ser difundidas e motivadas de modo a tornar a Web Semântica tão popular ou importante quanto à World Wide Web. Ainda há um caminho muito longo a ser percorrido, tanto no desempenho das tecnologias quanto na divulgação da ideologia. De qualquer maneira, fato é que as tecnologias existentes atualmente já constatam claramente o grande poder da Web Semântica. Este trabalho mostrará algumas delas.

Em relação às áreas da Ciência da Computação em que a Web Semântica se enquadra, pode-se dizer que em duas grandes áreas: Banco de Dados e Inteligência Artificial. Como visto, a Web Semântica foi essencialmente concebida para facilitar o entendimento das informações na web não só por humanos, mas também por computadores. A Inteligência Artificial estuda desenvolver algoritmos que fazem com que uma máquina seja inteligente o suficiente para fazer inferências. Ou seja, deduzir informações que não estão explicitadas na base de conhecimento. Existe um ramo na Web Semântica que só foca, estuda e desenvolve práticas em inferências lógicas. Essa subárea da Web Semântica é conhecida por "Inferências Semânticas" (*Semantic Reasoning*) (FRANCONI, 2002).

A outra grande área da Ciência da Computação que estuda Web Semântica é Banco de Dados. O foco é analisar como os dados são armazenados, organizados, como se ligam, se interoperam e quais as melhores técnicas de acesso a esses dados.

É importante ressaltar que este trabalho tem uma visão essencialmente voltada para Banco de Dados, embora não desconsidere a parte de Inteligência Artificial.

1.3 SLAC e PingER

1.3.1 História do SLAC como motivação

A oportunidade de aplicar os conceitos de Web Semântica em um cenário real foi proporcionada graças a uma parceria entre o Grupo de Engenharia do Conhecimento (GRECO)[1], na Universidade Federal do Rio de Janeiro representado pela Profa. Dra. Maria Luiza Machado Campos, e o SLAC Stanford National Accelerator Laboratory (SLAC)[2], operado pela Universidade de Stanford, representado por Prof. Dr. Bebo White e Prof. Dr. Les Cottrell. Este trabalho teve origem como parte do programa de intercâmbio do Governo Federal do Brasil, conhecido como Ciência sem Fronteiras[3]. Nesse programa, o aluno em intercâmbio completa um ano letivo em uma universidade do exterior e depois permanece três meses em um programa de estágio supervisionado em expediente integral, o qual foi desenvolvido no SLAC.

O SLAC é um respeitado laboratório especializado em aceleração de partículas atômicas. Esses tipos de pesquisa científica, em especial de ciências físicas, historicamente demonstram impulsionar o desenvolvimento tecnológico de novas áreas, especialmente da computação. A propósito, foi também de um laboratório renomado especializado em física de partículas, o CERN (European Organization for Nuclear Research)[4], – atualmente na mídia pelo prêmio Nobel de Física de 2013 pelo descobrimento do bóson de Higgs (NOBEL PRIZE, 2013) –, que se originou o projeto conhecido como o nascimento da World Wide Web proposto em 1990 por Berners-Lee e R. Cailliau, sendo Lee quem propôs a Web Semântica (BERNERS-LEE, 1998).

Adicionalmente, também em 1990, cientistas do SLAC receberam o prêmio Nobel de Física pelas investigações pioneiras para o desenvolvimento do modelo da partícula subatômica *quark* (WHEELER, 2006). Ademais, Les Cottrell – líder da equipe de computação do time responsável por esse Nobel e chefe atual do departamento de redes e telecomunicação do SLAC – e Bebo White – especialista em ciências da web e conselheiro no SLAC – fizeram parte da equipe responsável pela instalação do primeiro servidor World Wide

[1] Grupo de Engenharia do Conhecimento: http://greco.ppgi.ufrj.br
[2] SLAC Stanford National Accelerator Laboratory: http://slac.stanford.edu
[3] Ciência sem Fronteiras: http://www.cienciasemfronteiras.gov.br
[4] CERN Europeran Organizaion for Nuclear Research: cern.ch

web no SLAC e primeiro fora da Europa. Além disso, Cottrell é o líder e idealizador do projeto utilizado como cenário por este trabalho, o projeto PingER.

Cottrell, Bebo e Maria Luiza e equipe supervisionaram e auxiliaram diretamente este trabalho.

Sendo assim, além de estar na historicamente prestigiada Universidade de Stanford, todos esses fatores e ambiente colaboraram efetivamente para a motivação deste trabalho, cujo objetivo é aplicar os conceitos de Web Semântica em um cenário real (do PingER) e abrir os dados no formato de Dados Abertos Interligados (*Linked Open Data* – LOD).

1.3.2 Cenário real: PingER

O cenário deste trabalho é o projeto PingER (Ping End-to-end Reporting), que monitora o desempenho de *links* de Internet ao redor do mundo. Foi desenvolvido pelo Grupo IEPM (Internet End-to-end Performance Monitoring) [5] no SLAC e é liderado pelo departamento de redes e telecomunicações, na divisão de Computação no SLAC. Les Cottrell, também orientador deste trabalho enquanto no SLAC, é o principal líder do projeto PingER. Os dados do PingER são relacionados ao monitoramento da qualidade de *links* de Internet desde 1998 até os dias de hoje. O projeto descreve medições de cerca de 80 nós monitores para mais de 800 nós monitorados ao redor do mundo (cerca de 8200 pares) em mais de 160 países, podendo informar a quase totalidade da população mundial conectada à Internet sobre a qualidade da rede (COTTRELL, 2001). Uma descrição mais aprofundada sobre o domínio do PingER, o que seus dados medem e como são acessados será apresentada na seção 4.1 quando analisaremos o domínio do PingER.

Cottrell (2011) explica que os dados obtidos nas medições do projeto PingER provê várias aplicações de cunho:

- Técnico. Dados sobre fácil monitoramento de métricas de rede como *throughput,* perda de pacotes, tempos de resposta, medidas de um *link* em particular, etc.
- Econômico. Baseados nas descobertas proporcionadas pelos dados do PingER, uma recomendação pode ser feita para auxiliar no suporte à decisão de uma pessoa ou instituição de aumentar a banda de Internet de um determinado lugar.
- Detecção de problemas. Os dados podem ser utilizados para identificar algum problema relacionado à rede de Internet de algum lugar específico. Quando e onde

[5] http://www-iepm.slac.stanford.edu/

ocorreu o problema, se ainda está ocorrendo, etc. Análises quantitativas e qualitativas são disponibilizadas.

- Colaborativo. Para a colaboração entre cientistas e acadêmicos, um certo grau de qualidade de *link* de Internet é necessário. Diversas métricas providas pelos dados do PingER possibilitam medir a qualidade dos *links*.
- Quantificação do impacto de eventos. Os dados do PingER têm sido usados para mostrar o impacto da Internet em cortes de cabos submarinos, terremotos e tsunamis, impacto de novas conexões, etc.
- Roteamento. O PingER pode ser usado para auxiliar a identificação de roteamentos inapropriados estendendo *Round Trip Times*. Também pode ser utilizado para escolher a melhor rota.

Adicionalmente, PingER possui vários casos de estudo que demonstram na prática as aplicações dos dados colhidos pelo projeto. Por exemplo, alguns objetos de estudo que puderam ser identificados utilizando exclusivamente os dados do PingER (COTTRELL, 2013):

- Em setembro de 2013, os dados apontavam para uma desconexão súbita do Sudão da Internet;
- Os dados possibilitaram identificar que a Síria ficou *off-line* por 20 horas entre os dias 7 e 8 de maio de 2013;
- Em 27 de fevereiro de 2010, os dados de monitoramento da Internet no Chile foram perturbados consideravelmente, condizendo com o terremoto de escalas desastrosas que aconteceu neste dia. Isso contribui para quantificar o impacto desse tipo de evento;
- Os dados do PingER demonstram uma melhoria linear da Internet no Brasil entre Janeiro e Julho de 2001;
- Utilizando os dados, é possível distinguir que áreas da África tem uma conexão pior e que precisam de esforços mais urgentemente.

Analisando esses aspectos, conclui-se que os dados providos pelo projeto PingER são consideravelmente úteis e importantes. Por medirem a qualidade da Internet em várias perspectivas ao redor do mundo, são capazes de identificar eventos, situações ou lugares críticos que precisam da Internet e do acesso à informação. Isso tudo contribuiu para o crescimento tecnológico, social e educacional da sociedade ao redor do mundo. Dessa forma, a importância do projeto também atuou como motivador deste trabalho.

1.4 Objetivo: PingER Linked Open Data

Este trabalho tem por objetivo, após reunir os elementos conceituais e tecnológicos da Web Semântica, desenvolver um projeto para publicar dados do PingER em formato Linked Open Data. A esse projeto deu-se o nome de PingER Linked Open Data[6]. Como consequência do desenvolvimento do projeto, foi proposto um Processo de Publicação de Linked Open Data o qual reúne orientações de como publicar dados em LOD, enfatizando as vantagens proporcionadas pela publicação nesse formato.

O projeto PingER Linked Open Data será descrito detalhadamente, desde o início da sua concepção até a fase de consumo e aplicações dos dados. Todos os benefícios proporcionados pela utilização das tecnologias de Web Semântica serão enfatizados neste projeto, evidenciando o que ela se propõe a resolver e melhorar. Entretanto, as dificuldades encontradas até o produto final também serão ressaltadas.

Quanto à estrutura desta monografia, no capítulo 2 será feita uma revisão da literatura da Web Semântica, focando essencialmente nos aspectos teóricos e práticos fundamentais e essenciais com o objetivo de publicar dados em LOD. No capítulo 3, será apresentada a proposta de abordagem do Processo de Publicação de LOD. No capítulo 4, todo o processo apresentado no capítulo 3 será aplicado ao domínio do PingER, apresentando na prática os aspectos tecnológicos da abordagem, ratificando o sucesso das soluções bem como toda a dificuldade até os resultados finais. Finalmente, o capítulo 5 concluirá este trabalho, sintetizando os principais pontos dos capítulos 2, 3 e 4, resumindo os benefícios proporcionados por este projeto de pesquisa e desenvolvimento. Também listará trabalhos futuros relacionados tanto à Web Semântica em geral quanto ao projeto PingER LOD.

[6] www.pingerlod.slac.stanford.edu

2 FUNDAMENTOS DA WEB SEMÂNTICA

2.1 Definições fundamentais de Ciência da Web

Antes de prosseguirmos para os detalhes técnicos da Web Semântica, é necessário definir alguns termos fundamentais em Ciência da Web.

a) Dado

"Informação factual, especialmente organizada para análise ou usada para tomadas de decisão" [7]. Para Ciência da Computação, é qualquer registro do mundo real que possa ser processado por um computador. Por exemplo, o número `59` ou o conjunto de palavras `Silvia Maria`.

b) Informação

É um dado interpretado de uma maneira que seja útil no mundo real. Por exemplo, entender ou interpretar que o conjunto de caracteres `Brasil` é o **nome** de um lugar que tem uma população e que o número `201.032.714` é a **população** desse lugar. Apesar dessa interpretação ser tão natural para humanos, é necessário haver algo a mais para auxiliar os computadores a interpretarem dados, tornando-os em informação útil.

c) Metadado

"Meta-" é um prefixo grego que significa "depois", "além", "junto de", "sobre" [8]. Em Ciência da Web, entende-se que metadado é "dado que descreve dado" e é essencial para auxiliar a organização dos dados na web. Nos exemplos anteriores, **população** seria um metadado para o dado `201.032.714`. É importante destacar a importância que os metadados têm para trazer a semântica aos dados.

d) Estruturação dos dados

Diz-se que os dados em relação a um domínio estão estruturados quando eles estão organizados em entidades (ou classes) que compartilham as mesmas características (ou propriedades ou atributos), quando as entidades estão classificadas de acordo com alguma taxonomia e indivíduos pertencentes a uma mesma classificação possuem as mesmas propriedades. Por exemplo, criar a classe `Lugar com População`, dizer que todos os lugares com população possuem as propriedades nome e população. Criar a classe `País`, dizer que

[7] http://www.thefreedictionary.com/data
[8] http://www.thefreedictionary.com/meta-

todo país é um lugar com população e que todo país `tem capital`. Criar os indivíduos `Brasil` e `Copacabana` e dizer que, por serem `Lugares com População`, têm `nome` e `população`. Porém, somente para `Brasil`, uma instância da classe `País`, faz sentido dizer qual é sua `capital`. Infelizmente, a grande maioria dos documentos na web hoje é predominada por dados não estruturados, ou seja, os documentos HTML em geral só contêm dados e textos, mas nada que os descreve ou os dá alguma estrutura. Observação: Nesta monografia, o termo "organizado" é relaxadamente às vezes utilizado como sinônimo para estruturado.

e) Modelo

"Representação externa e explícita de parte de uma realidade vista pelas pessoas que vão utilizar o modelo para entender, mudar, gerenciar e controlar aquela parte da realidade" (PIDD, 2000).

f) Esquema

Uma especificação formal da descrição de toda a estrutura dos dados do domínio. Utiliza os metadados para definir a estrutura, as relações entre as classes, a taxonomia e as restrições (ex. o metadado idade descreve dados do tipo número inteiro).

g) Navegação, busca e consulta

Nos sistemas de hipermídia (documentos web ligados por *links*), **navegar** significa percorrer os *links* até chegar aonde se deseja. As **buscas** utilizam algoritmos e mecanismos estudados na disciplina Recuperação da Informação para encontrar o que se deseja. Facilita significativamente encontrar o que se procura e até hoje é essencial na web; tanto é que o Google, a grande referência dos mecanismos de busca, é a ferramenta computacional mais utilizada no mundo. As **consultas** são comuns em sistemas de banco de dados e é consideravelmente mais preciso do que as buscas já que atuam em um ambiente mais estruturado, facilitando encontrar *exatamente* o que se procura. Na Web Semântica, queremos realizar consultas aliadas às já estabelecidas buscas, tornando a precisão dos resultados muito melhor (CAMPOS, 2013).

h) Web 3.0

Entende-se que a web pode ser dividida em três fases. Resumidamente, nos primórdios da WWW, a fase web 1.0, tida como "web da publicação", os *websites* tinham uma característica "*read-only*". Ou seja, as informações eram simplesmente expostas em

páginas estáticas em HTML puro e conectadas a outras através de "*links*". A maneira mais comum de se encontrar o que procurava era navegando por esses *links*. Na Web 2.0, conhecida como "web da interação", anos depois, muito mais pessoas passaram a ter a capacidade de participar dinamicamente na publicação de conteúdo, deixando a web com uma característica *"read-write"* e enfatizando sua força e popularidade. A maneira mais comum de se encontrar o que procura é utilizando os mecanismos de busca. A terceira fase, a Web 3.0, às vezes é utilizada como sinônimo de Web Semântica. Nessa fase, considera-se uma web integrada e mais inteligente de modo a, automaticamente, entender o significado dos dados e consumi-los cruzando e explorando toda a diversidade de natureza e amplitude dos dados publicados na web (WILLIAMS, 2011). Conrad Wolfram, irmão do criador do conhecido mecanismo de conhecimento computacional *Wolfram Alpha*[9], argumenta que na Web 3.0 o computador, ao invés dos humanos, gera novas informações (KOBIE, 2010; JOHNSON, 2013).

i) Web Semântica

Uma extensão da Web 2.0 que visa dar significado ao conteúdo das páginas web, facilitando entendimento, troca e geração automática de dados por um computador, não só por pessoas.

j) Web de Dados

(Em Inglês, *Data Web* ou *Web of Data*). Uma visão idealizada da web que visa converter a web predominada por documentos não estruturados e semi-estruturados em uma web mais estruturada. É mais um nome que aparece na literatura como sinônimo de "Web Semântica" (BLOOMBERG BUSINESSWEEK, 2007).

k) Ciência da Web

(Em Inglês, *Web Science*). É a ciência social e tecnológica que estuda como a *World Wide Web* afeta, reflexivamente, os humanos, considerando todos os aspectos tecnológicos que envolvem essa relação (SHNEIDERMAN, 2007).

2.2 Ontologias

2.2.1 Definição de ontologia

> Thomas Gruber (apud CAMPOS, 2013) define ontologia como sendo "uma especificação explícita e formal de uma conceituação compartilhada". É uma

[9] http://www.wolframalpha.com/

> especificação porque é um modelo abstrato de um fenômeno no mundo; é explícita porque os tipos de conceitos usados e suas restrições devem estar explicitamente definidos; é formal porque a ontologia deve ser processada por máquina; é compartilhada, pois as ontologias devem capturar o conhecimento aceito por consenso pelas comunidades que delas fazem uso.

Para Guarino (1998), o uso mais frequente do termo "ontologia" na área de Inteligência Artificial se refere a um "artefato de engenharia, constituído de um vocabulário específico usado para descrever uma certa realidade, mais o conjunto de pressupostos explícitos relacionados a um significado pretendido do vocabulário". Em palavras mais simples, "uma ontologia descreve uma hierarquia de conceitos relacionados pela reunião de relacionamentos" (GUARINO, 1998; GUARINO; GIARETTA; CARRARA, 1993).

Para a web de dados, a ontologia funciona como o modelo que define a estrutura de um domínio, incluindo a taxonomia, caracterização dos conceitos, relação entre eles e restrições específicas do domínio. Na Web Semântica, a ontologia adiciona relações mais semânticas e inteligentes entre documentos ou qualquer outro recurso na web.

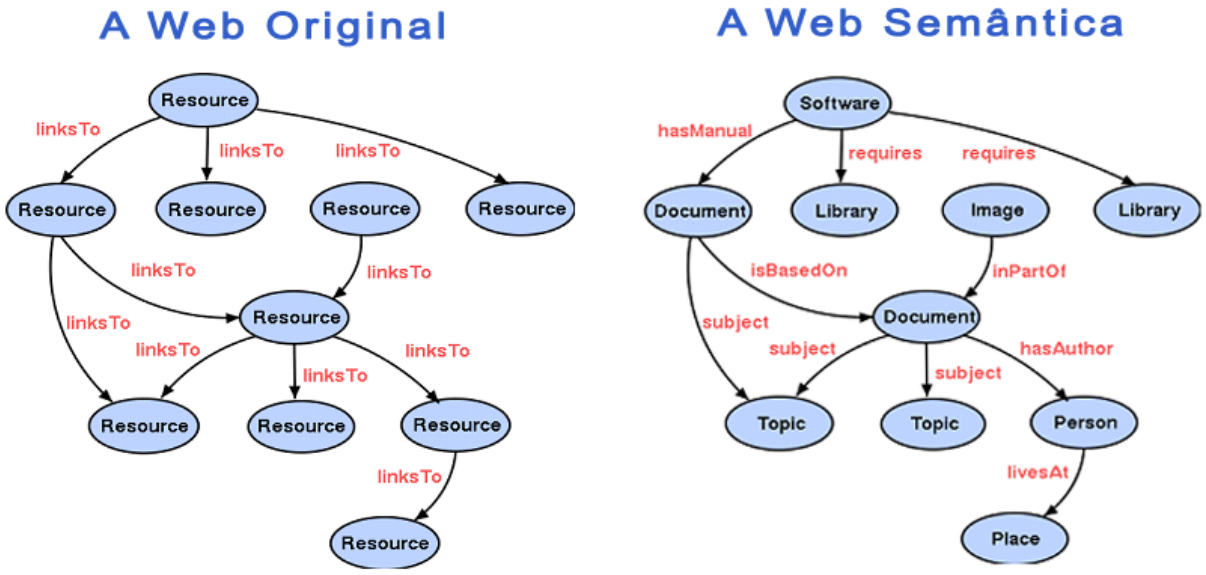


Figura 1 – Comparação entre a web de documentos e a web de dados, com ligações semânticas[10].

Deve-se enfatizar sua característica de ser de fácil entendimento não só para humanos, mas para máquinas processarem. Além disso, também acrescenta uma camada de inferência sobre objetos da web (GRUBER, 1993). Em adição, ontologias estabelecem níveis de interoperabilidade entre as aplicações de Web Semântica (BERNERS-LEE, 2006) e acrescentam significado, representação e expressividade sobre as camadas atuais da web (DJURIC; GAŠEVIC; DEVEDŽIC, 2006).

[10] http://ciencialaultima.blogspot.com.br/2012/12/web-30.html

Resumidamente, é a maneira de modelar conceitos no mundo da Web Semântica. Em analogia com o mundo do banco de dados (BD) relacional, a ontologia do projeto de Web Semântica seria o modelo Entidade-Relacionamento de um projeto de BD relacional.

Entretanto, apesar da analogia com o modelo do banco de dados relacional, as ontologias são bem mais flexíveis que os esquemas tradicionalmente estruturados dos BDs relacionais. Ou seja, alterar a estrutura de um modelo de ontologia costuma ser bem menos penoso quando comparado ao mesmo tipo de alteração em um esquema de banco de dados relacional. Esse fato é notório quando se adiciona ou remove relações ou entidades no banco de dados nos formatos da Web Semântica já povoado com milhões de registros. Isso é mais um benefício proporcionado pela utilização de tecnologias de Web Semântica.

Uma ontologia é composta por:

- Vocabulário controlado, que são listas dos termos do domínio enumerados explicitamente. Todos os termos devem ser não ambíguos e não redundantes. Um glossário contendo uma definição não ambígua e não redundante de cada um desses termos é essencial para a descrição do vocabulário controlado;
- Taxonomia, que são coleções de vocabulários controlados organizados em uma estrutura hierárquica. Por exemplo, `Lugar com População` se especializa em `País` e `Cidade`. Cidade poderia ainda se especializar em `Cidade Comum` e `Capital`. Cada especialização tem suas características específicas.
- Conjunto de Relações, que especifica as relações entre os termos do vocabulário controlado. Ou seja, além das relações de hierarquia providas pela taxonomia, o Conjunto de Relações adiciona relações semânticas de associação entre os conceitos do domínio. Por exemplo, definir a relação `tem capital` e declarar que indivíduos do tipo `País` se relacionam com indivíduos do tipo `Capital` pela relação `tem capital`.
- Axiomas, regras e restrições que podem ser usadas para deixar o esquema ainda mais expressivo e próximo da realidade do domínio, além dar suporte a inferências.

2.2.2 URIs, Indivíduos, Recursos, Literais e Prefixos

No mundo da Web Semântica e ontologias, alguns termos e conceitos são utilizados frequentemente e precisam ser definidos:

a) URI

Uniform Resource Identifier. Na Web Semântica, URI é uma *string* utilizada para identificar o nome de um recurso na web. Analogamente, é como a "chave primária" do banco de dados relacional, porém, ao invés de identificar uma instância naquele domínio, a URI deve identificar um recurso em toda a *World Wide Web*, ou seja, deve ser **única** em toda a WWW. Não deve ser confundida com URLs (*Uniform Resource Locator*), que é um endereço para uma página na web. As URIs são identificadores e não necessariamente precisam indicar um endereço para uma página web.

b) Indivíduo

É uma instância, ou seja, um exemplar de uma classe. Exemplos: O recurso Rio de Janeiro é um indivíduo da classe Cidade.

c) Recurso

Qualquer "coisa" ou entidade que possa ser identificada, nomeada ou endereçada de alguma forma na web de dados. Em um modelo de ontologias, as classes, as relações entre elas e os indivíduos são recursos. Exemplos: a classe `País`, o indivíduo `Brasil` (uma instância de `País`) e a relação `tem capital` são recursos.

d) Literal

É um valor que uma determinada propriedade pode assumir. Um literal é de algum tipo, geralmente primitivo, de dado. Por exemplo, "República Federativa do Brasil" é um literal do tipo cadeia de caracteres (*string*), que é o valor da propriedade `Nome Oficial` da classe `País`, do indivíduo `Brasil`.

e) Namespaces

Também conhecidos como prefixos. É como um apelido para um caminho ou endereço, em especial, uma URL. Provê meios de interpretar, acessar e definir os termos e indivíduos de forma não-ambígua, especificando a que domínio ou ontologia cada um desses termos pertencem. Além disso, auxilia a leitura das ontologias e diminui o tamanho das URIs. Por exemplo, um dos prefixos mais comuns, utilizado amplamente para definir a que tipo um indivíduo pertence, é o *rdf* que substitui a URI "http://www.w3.org/1999/02/22-rdf-syntax-ns#". Para definir que o indivíduo `Brasil` é do tipo `País`, pode-se declarar `(Brasil,` **`http://www.w3.org/1999/02/22-rdf-syntax-ns#type`**`, País)` ou, utilizando o prefixo *rdf*, `(Brasil,` **`rdf:type`**`, País)`.

2.2.3 Elementos básicos de ontologias

De modo simplista, uma ontologia pode ser modelada definindo-se apenas as classes do domínio, as propriedades que as classes podem ter e as relações entre as classes. Por isso, devido à importância desses dois conceitos (classe e propriedade), esta seção é dedicada à definição e à explicação desses termos.

a) Classes (e subclasses)

Em uma ontologia, é necessário descrever e definir as **classes** a que os indivíduos pertencem e quais propriedades eles terão por pertencerem à determinada classe. Adicionalmente, devido às técnicas de inferência e ao poder das ontologias, é possível afirmar que indivíduos são pertences a determinadas classes, mesmo se não forem diretamente declarados como sendo pertencentes a elas.

Utilizando os conceitos de taxonomia (vistos na seção 2.1.1), é possível ainda introduzir o conceito de herança de classes ou, mais comumente, **subclasse** de classes. Foi visto que em uma possível modelagem, pode-se declarar a classe mais genérica `Lugar com População`, que possui as propriedades `nome` e `população`. Poderia ainda especializar essa classe nas subclasses `País`, que possui a propriedade `tem capital`, e `Cidade`. Por serem subclasses de `Lugar com População`, `País` e `Cidade` herdam as propriedades `nome` e `população`. Porém, somente para `País` a propriedade `tem capital` faz sentido; para `Cidade` não. Em relação às inferências, um indivíduo da classe `País` é também um indivíduo da superclasse `Lugar com População`, mesmo que não seja explicitamente declarado assim.

a) Propriedades (e subpropriedades)

Além de definir as classes e suas relações de taxonomia, as ontologias também consideram as características das classes, ou seja, aspectos que diferenciam uma classe de outra. Para isso, utiliza-se as **propriedades**.

Em ontologias, as propriedades podem conter descrição de valores ou podem relacionar indivíduos. Por exemplo, `nome` é uma propriedade que assume valores do tipo *string* e `tem capital` é uma propriedade que associa indivíduos da classe `País` a indivíduos da classe `Cidade`.

As propriedades também podem ser hierarquizadas em taxonomias. Por exemplo, a propriedade `hasParent` pode ser especializada em `hasMother` e `hasFather`.

2.2.4 OWL

A OWL *(Web Ontology Language)* tem o objetivo de prover uma linguagem que pode ser utilizada para descrever as classes e as relações entre elas que são inerentes a documentos e aplicações web. Essa linguagem é capaz de formalizar conceitos de um domínio, definindo as classes e suas propriedades; definir os indivíduos; e inferir sobre essas classes e indivíduos utilizando as semânticas da providas pela OWL. É uma "Recomendação do W3C" para escrever ontologias na Web Semântica (WORLD WIDE WEB CONSORTIUM, 2004a).

Os vocabulários mais utilizados em uma ontologia OWL possuem os seguintes prefixos:

- `owl: <http://www.w3.org/2002/07/owl#>`
- `rdf: < http://www.w3.org/1999/02/22-rdf-syntax-ns#>`
- `rdfs: < http://www.w3.org/2000/01/rdf-schema#>`
- `xsd: < http://www.w3.org/2001/XMLSchema#>`

A OWL possui um conjunto de termos e regras bem abrangentes e genéricos capazes de definir uma modelagem significativa de uma considerável diversidade de domínios. Enfatizando a ideia de haver máquinas facilmente processando de forma sistemática os dados. Adicionalmente, ontologias em OWL possuem várias regras que dão suporte importante para inferências lógicas, isto é, fazer o computador gerar informações não diretamente instanciadas na base de conhecimento.

Entretanto, esta monografia cobre apenas um subconjunto desses termos. A saber, os termos mais importantes para uma modelagem focada em Banco de Dados e que foram amplamente utilizados no projeto cenário deste trabalho. A especificação de OWL, incluindo os termos cobertos neste trabalho e outros que dão mais suporte a inferências e tornam o modelo ainda mais próximo da realidade do domínio pode ser encontrado no manual da OWL do W3C (WORLD WIDE WEB CONSORTIUM, 2004a). Nesta seção serão utilizados exemplos de OWL escritos em XML. Os termos que se destacam para um modelo de ontologia escrito em OWL são:

a) Para classes e taxonomias: `owl:Class`, `rdfs:subClassOf` e `owl:Thing`

No mundo OWL, todas as classes são subclasses da classe mais genérica possível `owl:Thing`.

O termo `owl:Class` define uma classe. Exemplo, definir a classe `Populated Place`:

```
<owl:Class rdf:ID="PopulatedPlace"/>
```

O termo `rdfs:subClassOf` define uma subclasse. Exemplo, definir a classe `Country` como sendo subclasse de `Populated Place`.

```
<owl:Class rdf:ID="Country">
  <rdfs:subClassOf rdf:resource="#PopulatedPlace" />
</owl:Class>
```

b) Para propriedades: `owl:ObjectProperty` e `owl:DatatypeProperty`

OWL separa as propriedades em categorias:

Propriedades de objeto (object properties), que ligam recursos a recursos. Por exemplo, definir a propriedade `hasCapital` que liga um recurso do tipo `Country` a um recurso do tipo `City`:

```
<owl:ObjectProperty rdf:ID="hasCapital">
  <rdfs:domain rdf:resource="#Country"/>
  <rdfs:range rdf:resource="#City"/>
</owl:ObjectProperty>
```

Propriedades de tipo de dado (datatype properties ou simplesmente data properties), que ligam recursos a literais. Por exemplo, definir a propriedade `population` da classe `PopulatedPlace` que tem um valor do tipo `xsd:integer` (número inteiro do vocabulário XML Schema Definition[11]) .

```
<owl:DatatypeProperty rdf:ID="population">
  <rdfs:domain rdf:resource="#PopulatedPlace" />
  <rdfs:range  rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>
```

Além dessas duas, também existem propriedades do tipo `owl:AnnotationProperty` que são usadas para adicionar informações sobre a ontologia ou seus componentes. Ex: comentários, descrição do criador, um *link* para `seeAlso`, etc.

Adicionalmente, para definir subpropriedades, utiliza-se o vocabulário `rdfs:subPropertyOf`. Para exemplificar o que foi visto na seção anterior (2.2.3), uma ontologia poderia ser modelada em OWL/XML desta forma:

```
<owl:ObjectProperty rdf:ID="hasMother">
   <rdfs:subPropertyOf rdf:resource="#hasParent" />
   <rdfs:domain rdf:resource="#Person"/>
   <rdfs:range rdf:resource="#Woman" />
</owl:ObjectProperty>
```

c) Para domínios e alcances: `rdfs:domain` e `rdfs:range`

Em OWL, o domínio (`rdfs:domain`) de uma propriedade representa as classes que podem ter aquela propriedade. O alcance (`rdfs:range`) de uma propriedade representa as

[11] http://www.w3.org/standards/xml/schema

classes dos recursos a que a propriedade se liga (no caso de object properties) ou o tipo de dado que a propriedade pode assumir (no caso de datatype properties).

d) `owl:sameAs`

Na Web de Dados, um indivíduo em um domínio pode referir-se a um mesmo indivíduo em um outro domínio, enfatizando o fato de eles serem referentes à mesma "coisa" no mundo.

Por exemplo, existe um banco de dados no mundo sobre o domínio de política mundial. Esse banco de dados precisa conter instâncias de todos os países do mundo. Em especial, contém a instância Brasil, a qual contém informações sobre população, investimentos públicos, nome do presidente e nome da capital. Já outro banco de dados, foca no domínio de geografia física mundial, que também precisa de instância de todos os países. Porém, no caso do Brasil, esse banco registra informações sobre relevo, clima, vegetação e território físico do país. É claro que ambos os indivíduos referem-se à mesma "coisa" no mundo, porém sob perspectivas diferentes, dependendo do domínio.

Se alguém quiser investigar a relação entre um país sob perspectiva política e o mesmo país sob a perspectiva geográfico-física, precisará de algum artifício para ligar esses dois indivíduos diferentes, mas que se referem à mesma coisa. Em OWL, esse artifício se dá pela utilização da object property `owl:sameAs` que liga um recurso do tipo `owl:Thing` a um outro recurso do tipo `owl:Thing`.

Essa é a maneira mais comumente usada para ligar mesmas instâncias de domínios completamente diferentes, ou seja, é um componente importante para construir uma das qualidades focadas na Web Semântica: a informação precisa estar conectada (visto na seção 1.1).

e) Classes Anônimas e Classes Equivalentes

Em OWL, uma Classe Anônima pode ser entendida como um conjunto de instâncias caracterizadas por alguma regra (ou **restrição**) que elas têm em comum. Esse conjunto, por ser uma classe, é instanciável. Uma restrição descreve uma classe de indivíduos baseando-se na relação da qual os membros da classe participam (HORRIDGE, 2011). Por exemplo, classe definida pelo conjunto de indivíduos que possuem pelo menos 1 relação `temIrmão`.

Em OWL, duas classes são equivalentes se elas contêm exatamente o mesmo conjunto de indivíduos. (Ex: `PresidenteDosEUA` e `PrincipalResidenteDaCasaBranca`).

É comum juntar esses dois conceitos para declarar que uma determinada classe A é equivalente a uma classe anônima definida por uma restrição. Dessa forma, os indivíduos que satisfazem tal restrição será membro daquela classe A. Veremos este exemplo na seção 4.2.

2.2.5 Ferramentas de modelagem de ontologias

Em relação a tecnologias utilizadas para modelar ontologias, as mais utilizadas são Protégé[12] e Neon Toolkit[13]. Utilizando-as, um ontologista não precisaria escrever códigos OWL em um editor de textos porque essas ferramentas oferecem interfaces amigáveis para desenhar e especificar a ontologia e exportá-la em OWL, com todos seus devidos formalismos e especificações.

Essas ferramentas contam com *plug-ins* de visualização gráfica do modelo, o que facilita o entendimento, análise e desenvolvimento das ontologias.

Além disso, o Protégé oferece facilidades para consultar (utilizando SPARQL que será visto na seção 2.4) e realizar inferências sobre os indivíduos e classes da ontologia. Já o Neon Toolkit, apresenta uma interface amigável e bem parecida com o Eclipse[14], conhecida IDE de Java e outras linguagens.

2.3 RDF, um modelo de dados em grafo

A ontologia define o esquema para o modelo conceitual do domínio. Porém, ainda se faz necessário definir um formato ou padrão para descrever o armazenamento dos dados propriamente ditos. Isto é, um padrão para declarar os recursos instanciados com suas propriedades e as ligações entre eles.

O padrão "Recomendação do W3C" para descrever os recursos na Web Semântica e para troca de dados na web é o RDF (*Resource Description Framework*)[15]. RDF utiliza URIs para identificar as instâncias e as relações entre recursos. É um dos principais componentes da Web Semântica e suporta significativamente os dados lidos por máquinas através da web, fortalecendo a ideia da estruturação dos dados.

Uma coleção de declarações RDF é estruturada e representada em um grafo direcionado e rotulado, como aquele da Web Semântica visto na Figura 1.

[12] http://protege.stanford.edu/
[13] http://neon-toolkit.org/wiki/Main_Page
[14] http://www.eclipse.org/
[15] http://www.w3.org/standards/techs/rdf#w3c_all

Vale ressaltar ainda que RDF, em específico, RDFS *(RDF Schema)* pode e é utilizado, juntamente com OWL, para descrever vocabulários e dar suporte a ontologias[16].

2.3.1 Triplas

Assim como o mundo relacional se baseia em tabelas e chaves identificadoras, RDF se baseia em declarações (em Inglês, *statements*) do tipo `sujeito, predicado, objeto`, ou `recurso, propriedade, valor`, conhecidas como **triplas** que, juntas, formam um grafo. O conceito de triplas é extremamente importante no mundo de Web Semântica, porque, atualmente, grande parte da representação dos dados se dá nesse formato **triplificado**. O sujeito da tripla significa o recurso a ser descrito, o predicado é a propriedade sobre o sujeito e expressa a relação entre o sujeito e o objeto. Por exemplo, na declaração `Brasil tem capital Brasília`, o sujeito é `Brasil`, o predicado é `tem capital` e o objeto é `Brasília`. Outro exemplo de tripla é (`Brasil, tem população, 201.032.714`). Na representação como grafo rotulado direcionado, o sujeito é o nó fonte, o objeto é o nó destino e o predicado é o rótulo da aresta que os conecta.

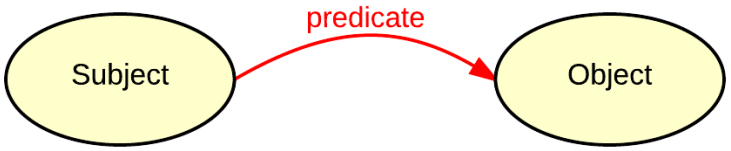


Figura 2 – Tripla RDF representada como grafo direcionado rotulado

Essa forma de representar a informação foi amplamente abraçada pela comunidade e é possível ver várias vantagens de sua utilização. A principal delas é que é um formato de declaração extremamente simples, mas genérico e significativo. Genericamente, qualquer tipo de coisa no mundo pode ser expressa utilizando a forma sujeito, predicado e objeto. Com uma coleção dessas declarações simples, pode-se construir bancos de informações bem expressivos de domínios de qualquer natureza. Além disso, as triplas permitem expressar informações extremamente específicas e detalhadas. Essa característica de simples, porém genérica e significativa das triplas RDF é chamada nesta monografia de **característica granular das triplas**[17]. Em contrapartida, por ser tão simples e genérica, expressar um domínio ou uma

[16] http://www.w3.org/TR/rdf-schema/

[17] Uma referência a **grão**, conceito muito usado em Data Warehousing, para indicar o detalhamento de uma informação. Quanto mais granular, mais detalhada é a informação. Uma tripla pode ser entendida como uma informação muito granular.

entidade utilizando triplas costuma trazer uma sobrecarga de informação necessária para especificar o que está sendo modelado ou instanciado.

2.3.2 Banco de dados importantes em RDF

Já existem diversos bancos de dados publicados em formatos de triplas e, com eles, já é possível realizar diversas consultas, experimentos e aplicações úteis capazes de auxiliarem tomadas de decisão. Alguns desses bancos são extremamente importantes e foram amplamente utilizados neste projeto, sendo, por essa razão, aqui descritos.

a) DBPedia[18]

É uma tentativa de estruturar, em formato de Web Semântica, parte dos dados da Wikipedia. É o banco de dados em RDF mais referenciado por outros. Ou seja, a grande maioria dos bancos de dados RDF existentes se ligam à DBPedia.

b) Geonames[19]

É um dos BD RDF de dados geográficos mais utilizados, aberto e público. Contém dados de mais de 8 milhões de "lugares", incluindo cidades, estados, países até lagos, oceanos, ruas, aeroportos, mercados e muito mais.

c) Freebase[20]

É mantido pelo Google, oferecendo dados abertos e ligados sobre mais de 39 milhões de tópicos.

d) The World Bank [21]

Oferece acesso público e livre a dados sobre desenvolvimento de países ao redor do globo.

e) Factforge[22]

Em um único gigantesco BD de RDF, inclui os seguintes conjuntos de dados: DBpedia, New York Times, MusicBrainz, Lingvoj, Lexvo, CIA World Factbook, WordNet, Geonames, Freebase.

[18] http://dbpedia.org
[19] geonames.org
[20] freebase.com
[21] data.worldbank.org
[22] factforge.net

2.3.3 Formatos de RDF e seus *MIME Types*

RDF é um framework, ou seja, um padrão para representar triplas na Web Semântica. Esse padrão pode ser escrito em diversos formatos, dentre os quais os mais populares são **RDF/XML**, **JSON**, **N-Triples** e **Turtle.** É importante também introduzir o conceito de *Internet Media Type*, mais conhecidos como *MIME Types*. São frequentemente utilizados na web em geral e servem para identificar o formato de arquivos sendo manipulado por alguma aplicação, geralmente especificando o tipo de arquivo na hora de realizar requisições e envios sobre a web. O exemplo mais comum é o "text/html", o *MIME Type* dos documentos HTML, isto é, a grande maioria dos *websites* disponíveis hoje na web ((ROBERTS, 2012; WORLD WIDE WEB CONSORTIUM, 2002).

Em RDF, os formatos especificados acima também possuem *MIME Type* associados, definidos pelo W3C. Eles auxiliam as aplicações e, em especial, os navegadores, a lidarem melhor com o tipo de dado em questão (WORLD WIDE WEB CONSORTIUM, 2008a).

Vale ressaltar que esses formatos são facilmente traduzidos entre si. Existem diversas ferramentas que fazem esse processo e o RDF Translator[23] é um exemplo.

Nesta seção, será explicado brevemente cada um desses formatos e ao fim serão mostrados exemplos de comparação entre os formatos citados.

a) RDF/XML

É um arquivo que segue as regras de um XML[24]. Vide abaixo para analisar como um RDF/XML é representado. Em particular, é o formato mais comum para descrever dados em RDF. O W3C oferece toda a especificação de como os dados devem ser escritos em RDF/XML (WORLD WIDE WEB CONSORTIUM, 2004b).

MIME Type: "application/rdf+xml"

b) JSON

Formato que segue as especificações definidas para JSON (JavaScript Object Notation)[25]. É um formato bem simples, implementado em várias linguagens e é de fácil troca de dados entre aplicações. É muito utilizado em aplicações web já que é nativo de JavaScript.

MIME Type: "application/json"

c) N-Triples

[23] http://rdf-translator.appspot.com/
[24] http://www.w3.org/XML/
[25] http://www.w3schools.com/json/

Provavelmente o formato mais simples de ser entendido. Nenhum conhecimento prévio em XML ou JSON é preciso e nem utiliza sintaxes mais complexas. Basicamente, é uma coleção de triplas linearmente listadas, cada uma delimitada por um ponto (“.”). A desvantagem é que é notória a necessidade de redundância nas declarações. No exemplo que será visto a seguir em N-Triples, o recurso sendo descrito tem que ser necessariamente repetido em todas as declarações. Além disso, esse formato não utiliza namespaces, perdendo todas as vantagens vistas anteriormente na seção 2.2.2.

MIME type: “text/plain” ou “application/n-triples”

d) Turtle

Terse RDF Triple Language (Turtle)[26] é talvez o formato mais interessante para representar RDF. Como será visto no exemplo, não ocorre o problema de redundância do recurso sendo descrito e ainda pode-se utilizar prefixos, diminuindo significativamente a quantidade de informação fisicamente armazenada em disco. Por esses motivos, Turtle é um dos formatos preferidos se for armazenar dados RDF em arquivos simples já que é o mais compacto em relação a tamanho de armazenamento em disco.

MIME type: “text/turtle”

2.3.3.1 Exemplos dos formatos

Considerando a descrição de um recurso hipotético `<my:Brasil>` (sendo `my` um prefixo) nos formatos citados, estas são as triplas a serem escritas:

```
<my:Brasil> <owl:sameAs> <dbpedia:Brazil>
<my:Brasil> <owl:sameAs> <geonames:3469034>
<my:Brasil> <owl:sameAs> <freebase:m.015fr>
<my:Brasil> <rdfs:label> "Brasil"@pt .
<my:Brasil> <rdfs:label> "Brazil"@en .
<my:Brasil> <dbpedia-owl:foundingDate> "1822-09-07"^^xsd:date .
<my:Brasil> <dbpprop:populationEstimate> "193946886"^^xsd:integer .
```

a) Exemplo em RDF/XML:

```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dbpedia-owl="http://dbpedia.org/ontology/"
      xmlns:dbpprop="http://dbpedia.org/property/"
      xmlns:owl="http://www.w3.org/2002/07/owl#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >
```

[26] http://www.w3.org/TR/turtle/

```
            <rdf:Description
rdf:about="http://example.org/my/resource/Brasil">
                  <owl:sameAs
rdf:resource="http://dbpedia.org/resource/Brazil" />
                  <owl:sameAs
rdf:resource="http://sws.geonames.org/3469034/" />
                  <owl:sameAs
rdf:resource="http://rdf.freebase.com/ns/m.015fr" />       <rdfs:label
xml:lang="pt">Brasil</rdfs:label>
                  <rdfs:label xml:lang="en">Brazil</rdfs:label>
                  <dbpedia-owl:foundingDate
rdf:datatype="http://www.w3.org/2001/XMLSchema#date">1822-09-07</dbpedia-
owl:foundingDate>
                  <dbpprop:populationEstimate
rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">193946886</dbpprop:
populationEstimate>
            </rdf:Description>
      </rdf:Description>
      </rdf:RDF>
```

b) Exemplo em JSON:

```
      {
            "http://example.org/my/resource/Brasil": {
                  "http://www.w3.org/2002/07/owl#sameAs": [
      {"type":"uri", "value":"http://dbpedia.org/resource/Brazil"},
      {"type":"uri", "value":"http://sws.geonames.org/3469034/"},
      {"type":"uri", "value":"http://rdf.freebase.com/ns/m.015fr"}
                  ],
                  "http://www.w3.org/2000/01/rdf-schema#label": [
                        {"type":"literal", "value":"Brasil", "lang":"pt"},
                        {"type":"literal", "value":"Brazil", "lang":"en"},
                  ],
                  "http://dbpedia.org/ontology/foundingDate>": [
      {"type":"literal", "value":"1822-09-07",
"datatype":"http://www.w3.org/2001/XMLSchema#date"},
                  ],
                  "http://http://dbpedia.org/property/populationEstimate>":
[
      {"type":"literal",
"value":193946886,"datatype":"http://www.w3.org/2001/XMLSchema#integer"},
                  ]
            }
      }
```

c) Exemplo em N-Triples:

```
      <http://example.org/my/resource/Brasil>
<http://www.w3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Brazil> .

      <http://example.org/my/resource/Brasil>
<http://www.w3.org/2002/07/owl#sameAs> <http://sws.geonames.org/3469034/> .

      <http://example.org/my/resource/Brasil>
<http://www.w3.org/2002/07/owl#sameAs>
<http://rdf.freebase.com/ns/m.015fr> .
```

```
      <http://example.org/my/resource/Brasil>
<http://www.w3.org/2000/01/rdf-schema#label> "Brasil"@pt .

      <http://example.org/my/resource/Brasil>
<http://www.w3.org/2000/01/rdf-schema#label> "Brazil"@en .

      <http://example.org/my/resource/Brasil>
<http://dbpedia.org/ontology/foundingDate> "1822-09-
07"^^<http://www.w3.org/2001/XMLSchema#date> .
      <http://example.org/my/resource/Brasil>
<http://http://dbpedia.org/property/populationEstimate>
"193946886"^^<http://www.w3.org/2001/XMLSchema#integer> .
```

d) Exemplo em Turtle:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dbpedia-owl: <http://dbpedia.org/ontology/> .
@prefix dbpprop: <http://dbpedia.org/property/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix my: <http://http://example.org/my/resource/> .

my:Brasil   owl:sameAs  <http://dbpedia.org/resource/Brazil> ,
                        <http://sws.geonames.org/3469034/> ,
                        <http://rdf.freebase.com/ns/m.015fr> ;

            rdfs:label  "Brasil"@pt ,
                        "Brazil"@en .

my:Brasil   dbpedia-owl:foundingDate "1822-09-07"^^xsd:date .
my:Brasil   dbpprop:populationEstimate 193946886 .
```

2.3.4 Sistemas de Gerenciamento de Banco de Dados RDF

Repositório RDF, Repositório de Triplas (*Triple Store*), Repositório Semântico, e etc são todos termos que se referem ao objeto de estudo da disciplina responsável por analisar como as triplas estão armazenadas, desde o modelo físico até o lógico, e como realizar operações de CRUD (criar, consultar, atualizar e deletar) sobre as triplas RDF. Esse objeto de estudo pode ser chamado de Sistema de Gerenciamento de Banco de Dados RDF (SGBD RDF) (ONTOTEXT, 2013).

Assim como no mundo dos bancos de dados relacionais existem os SGBD Relacionais (ex. Oracle, MS SQL Server, MySQL, etc.), no mundo de Web Semântica existem os SGBD RDF, que são conjuntos de ferramentais que oferecem as operações citadas acima. Diversas alternativas podem ser encontradas e, para fazer a melhor escolha do repositório para o projeto, vale investir tempo no processo de escolha, de modo que se analisem as funcionalidades, escalabilidade e outras características.

No capítulo 3 serão listadas as algumas alternativas de SGBD RDF existentes atualmente, enfatizando a importância de uma boa escolha do repositório RDF.

2.4 SPARQL: Consulta e Serviço

2.4.1 Consultas SPARQL

SPARQL é um acrônimo recursivo para *SPARQL Protocol for RDF Query Language*. É a linguagem padrão para consultar dados em RDF. Sua primeira versão, SPARQL 1.0, tornou-se Recomendação da W3C em 2008 (WORLD WIDE WEB CONSORTIUM, 2008c) e sua segunda versão, SPARQL 1.1, que possibilitou todas as operações de CRUD embutidas na sintaxe da própria linguagem, virou Recomendação da W3C em março de 2013 (WORLD WIDE WEB CONSORTIUM, 2013a).

Apesar do SPARQL ter sido feito para consultar dados em formatos de triplas, esta linguagem possui muita semelhança com o SQL (*Structured Query Language*)[27] do mundo de banco de dados relacional. A construção de uma consulta, as funções, os agregadores, as palavras-chaves e outros aspectos são bem semelhantes ao SQL. Isso contribuiu com o entendimento da linguagem e facilitou o aprendizado de quem já tinha um pré-conhecimento de SQL e SGBDs relacionais.

Esta é uma área relativamente nova. Note que SPARQL 1.1, com suporte a CRUD completo, só tornou-se oficialmente recomendação do W3C em março deste ano. Ainda há muito o que fazer e pesquisar na área de consultas SPARQL. Existem estudos focados somente na área de otimização de consultas e estratégias mais eficientes da camada física dos dados triplificados, isto é, armazenamento e recuperação com melhor desempenho (SCHMIDT, 2010; HARTH; DECKER, n.d.). Entretanto, algumas consultas ainda são consideravelmente lentas, o que motiva ainda mais a pesquisa nessa área.

2.4.2 Serviços SPARQL e SPARQL Endpoint

Há uma Recomendação do W3C que especifica padrões para a implementação do protocolo SPARQL como um serviço web para receber requisições HTTP e enviar respostas HTTP para um cliente. A URI que "ouve" serviços de protocolo SPARQL é conhecida como **SPARQL Endpoint** (WORLD WIDE WEB CONSORTIUM, 2008b).

[27] http://www.w3schools.com/sql/

Os SPARQL Endpoints têm um papel fundamental na comunidade de *Linked Open Data*, pois eles são como uma porta para acessar e consumir os dados de um SGBD RDF, através das consultas SPARQL. Por isso, além de ter os dados armazenados em um formato padrão de fácil troca entre instituições e de fácil processamento por máquinas, os dados triplificados só serão facilmente interoperados entre quaisquer organizações de naturezas mais diferentes possíveis se existir um SPARQL Endpoint. Logo, no durante o Processo de Publicação de Linked Open Data, estabelecer um SPARQL Endpoint é fundamental, como será visto no capítulo seguinte.

Alguns dos BDs importantes mencionados na seção 2.3.4 oferecem SPARQL Endpoints. Por exemplo, DBPedia[28], Factforge[29] e World Bank[30].

Para as instituições que vão começar a publicar dados em RDF e devem prover um SPARQL Endpoint, existem diversas tecnologias que implementam as especificações desse protocolo, tornando essa importante tarefa muito mais simples para os desenvolvedores (WORLD WIDE WEB CONSORTIUM, 2013d). Geralmente, os próprios SGBDs RDF implementam esses mecanismos e oferecem bibliotecas e APIs (Interface de programação de aplicações) para serem utilizadas dentro das linguagens de programação. Em outras palavras, é possível executar consultas, vindas de uma *string*, sobre os dados e tratar os resultados como uma estrutura de dados inerente da linguagem de programação sendo utilizada. Além, também, de poder inserir, atualizar ou remover triplas através dessas APIs.

É importante ressaltar que se a consulta vier do navegador, por meio de métodos HTTP, mais comumente GET, é necessário que a *string* passada à aplicação pelo navegador no método GET deve conter o atributo `query`, como definido na Recomendação do W3C (WORLD WIDE WEB CONSORTIUM, 2008b). Além disso, uma consulta SPARQL deve retornar um resultado utilizando o MIME Type "application/sparql-results+xml"[31].

A ferramenta SPARQLES (*SPARQL Endpoint Status*)[32] monitora a disponibilidade, desempenho, interoperabilidade e facilidade de descoberta de uma lista de mais de 400 SPARQL Endpoints. Para medir esses indicadores, a ferramenta utiliza, dentre outros recursos, serviços que obtêm a descrição da base de dados RDF e do SPARQL Endpoint como, por exemplo, vocabulário VOID e SPARQL Service Description (seções 2.4.3.1 e 2.4.3.2).

[28] http://dbpedia.org/sparql
[29] http://factforge.net/sparql
[30] http://worldbank.270a.info/sparql
[31] http://www.w3.org/2001/sw/DataAccess/rf1/#mime
[32] http://sparqles.okfn.org/

2.4.3 Descrição da base de dados e do SPARQL Endpoint

Para a boa manutenção da comunidade LOD, é importante, além de tornar o SPARQL Endpoint público, descrevê-lo para tornar sua acessibilidade e facilidade de descoberta maior.

> Para um consumidor encontrar informação relevante sobre um endpoint (ex. que tipo de dados ele contém, onde pode ser acessado o *RDF Dump*, etc.), o publicador dos dados deve publicar metadados fundamentais sobre as políticas, características e conteúdo do endpoint e *dataset*. A forma padrão e recomendada de tornar o endpoint possível de ser descoberto é através do uso do vocabulário VOID (*Vocabulary of Interlinked Datasets*) em adição aos metadados de descrição do próprio SPARQL Endpoint (SPARQLES, 2013).

2.4.3.1 VOID

VOID (*Vocabulary of Interlinked Datasets*)[33,34] foi designado para descrever BDs RDF interligados, definindo termos e padrões de descrição de *datasets*, criando a ponte entre produtores e consumidores de dados RDF. Com esse vocabulário, a descoberta e reúso de BDs RDF interligados podem ser executados de maneira eficiente (CYGANIAK, 2009).

VOID possui uma extensa lista de vocabulários úteis para descrição de *datasets*. Nesta seção listaremos um exemplo de caso de uso de apenas alguns termos do vocabulário, aplicando-os no projeto desta monografia, a base de dados RDF do PingER Linked Open Data. Definiremos, no exemplo (escrito em RDF/Turtle), a fonte dos dados (`source`), a URI do SPARQL Endpoint (`sparqlEndpoint`), a URI do *data dump* (`dataDump`) e assuntos, metadado que apoia a descrição semântica do conteúdo da base de dados. Os prefixos são definidos no Apêndice B:

```
:PingERLOD
      a                 void:Dataset;
      foaf:homepage     <http://www.pingerlod.slac.stanford.edu>;
      dc:source         <http://www-wanmon.slac.stanford.edu/pinger>;
      void:sparqlEndpoint <http://pingerlod.slac.stanford.edu/sparql>;
      void:dataDump     < http://www-
iepm.slac.stanford.edu/pinger/lod/data/dump.turtle>;
      dc:description "RDF Database in Linked Open Data formats for Semantic
Web public and standard access to PingER data."
      dc:subject  <http://en.wikipedia.org/wiki/Network_traffic_measurement>;
      dc:subject  <http://en.wikipedia.org/wiki/Ping_(networking_utility)>;
```

Também é possível utilizar VOID para definir a que outros bancos de dados externos a base se liga, utilizando o vocabulário `Linkset`. Isso auxilia a determinar *links* em nível de instância, isto é, especificar que 2 instâncias de bases de dados diferentes se referem ao

[33] http://www.w3.org/TR/void/
[34] http://semanticweb.org/wiki/VoID

mesmo indivíduo. Vimos que isso é geralmente feito utilizando-se o predicado de *links* `owl:sameAs` (seção 2.2.4), mas pode ser utilizada qualquer outra object property.

```
:DBPedia
      a                 void:Dataset;
      foaf:homepage     <http://dbpedia.org/>;
:Geonames
      a                 void:Dataset;
      foaf:homepage     <http://sws.geonames.org/>;
:Freebase
      a                 void:Dataset;
      foaf:homepage     <http://www.freebase.com/>;


:PingERLOD_DBpedia            a void:Linkset;
      void:linkPredicate      owl:sameAs;
      void:linkPredicate      PingER-ont:DBPediaLink;
      void:target             :PingERLOD;
      void:target             :DBPedia;

:PingERLOD_Geonames           a void:Linkset;
      void:linkPredicate      owl:sameAs;
      void:linkPredicate      PingER-ont:GeonamesLink;
      void:target             :PingERLOD;
      void:target             :Geonames;

:PingERLOD_Freebase           a void:Linkset;
      void:linkPredicate      owl:sameAs;
      void:linkPredicate      PingER-ont:GeonamesLink;
      void:target             :PingERLOD;
      void:target             :Geonames;
```

2.4.3.2 *SPARQL Service Description*

SPARQL 1.1 Service Description[35] é uma "Recomendação W3C" para tornar o SPARQL Endpoint mais informativo, permitindo que um cliente ou usuário final descubra detalhes importantes sobre o endpoint e o conjunto de dados. Tem objetivo de prover descrições essencialmente para serem lidas por máquina.

A descrição é feita em RDF e usa o vocabulário *Service Description*[36]. Essas descrições são disponibilizadas através da URI do SPARQL Endpoint e é retornada por padrão quando nenhuma consulta é especificada no parâmetro `query`.

Pela recomendação, serviços SPARQL disponibilizados via protocolo SPARQL devem retornar um documento RDF (RDF/XML normalmente, mas pode ser Turtle ou NTriples também).

[35] http://www.w3.org/TR/sparql11-service-description/
[36] http://www.w3.org/ns/sparql-service-description#

Adicionalmente, também é sugerido que se utilizem anotações RDFa [37] com vocabulários de *Service Description* na página HTML recebida pelo navegador ao executar um HTTP GET na URI do SPARQL Endpoint.

2.4.4 Consultas SPARQL Federadas

Consultas SPARQL Federadas são consultas executadas em mais de um SPARQL Endpoint ao mesmo tempo. São construtos extremamente úteis e poderosos capazes de realmente tratar a nuvem de LOD como um único gigante banco de dados RDF. Em outras palavras, é possível consumir dados de quaisquer fontes de dados RDF em uma mesma consulta, simultaneamente (WORLD WIDE WEB CONSORTIUM, 2013b).

SPARQL 1.1 implementa a palavra-chave `service` a qual aponta para um Endpoint externo. Por exemplo, utilizando o Endpoint do World Bank[38], podemos utilizar a DBPedia como um Endpoint externo na consulta federada para recuperar informações sobre a UFRJ (prefixo no Apêndice B):

```
select * where {
    SERVICE <http://dbpedia.org/sparql> {
       <dbp-rsrc:Federal_University_of_Rio_de_Janeiro> ?b ?c
    }
}
limit 10
```

Assim, podemos mesclar informações específicas do World Bank com informações da DBPedia.

Infelizmente, ainda existem muitas limitações principalmente no tempo levado para executar as consultas. O Apêndice E mostra um exemplo de Consulta Federada aplicado ao projeto PinER LOD que demorou mais de 3 horas para retornar o resultado até que o processo foi cancelado. Apesar de já existirem pesquisas focadas em otimizar esse tipo de consulta, claramente precisa ainda de mais investimento para desenvolver a tecnologia (ARAÚJO, 2012).

2.5 Linked Open Data

2.5.1 Visão Geral

> A Web Semântica não é apenas para colocar dados na web. É para fazer *links* de modo que uma pessoa ou máquina possa explorar a web de dados. Com dados

[37] http://www.w3.org/TR/rdfa-syntax/

[38] http://worldbank.270a.info/sparql

> ligados, quando você tem um pouco de determinada coisa, você pode encontrar muito mais relacionado a essa coisa.
> (BERNERS-LEE, 2009)

Com um presunçoso objetivo de criar um banco de dados global de proporções gigantescas que só a web poderia dar, Tim Barners-Lee conceituou o *Giant Global Graph* (GGG), com uma visão genérica de um modelo semântico de uma web de dados focada em conectividade entre dados (WEB NEXT, 2007).

Podemos reunir essa abordagem, conceitos e fundamentos da Web Semântica estudados até agora e discutir com maior profundidade o conceito que resume o objetivo: Dados Abertos Interligados ou *Linked Open Data* (LOD). LOD pode ser visto como uma "nova" maneira de publicar e consumir dados, considerando, em especial, os seguintes aspectos:

- Poder das ligações tipadas. Isto é, com semântica expressiva e classificação embutida;
- Poder dos dados abertos. Os dados devem ser publicados de uma maneira de fácil acesso, pública, livre de licenças, patentes, copyright, etc.;
- Poder da colaboração. Se os dados estiverem abertos e facilmente acessíveis, poderão ser úteis para um número bem significativo de pessoas. Além disso, favorece aplicações e novas ideias de *mashups* (explicado na próxima subseção);
- Os consumidores dos dados podem também ser publicadores;
- Dados potencialmente "ligáveis" são gerados a todo instante;
- Pelo avanço tecnológico dos sensores, abriu-se todo um novo leque de possibilidades de "coisas" do mundo real que podem ser significativamente representadas virtualmente no computador e consequentemente interligadas. Em outras palavras, pode-se relaxadamente afirmar que, daqui a não muito, será possível interligar e interoperar qualquer coisa no mundo, de qualquer natureza, formando uma grande "Web das Coisas" (GUINARD; TRIFA, 2009).
- Processamento dos dados por computadores, tornando o próprio computador capaz de gerar informação através de inferências;
- Exploração de dados e metadados de forma uniforme;
- Navegação, busca e consultas na web podem ser utilizadas, explorando-se as vantagens do melhor dessas três formas de acesso na web.

No cenário atual de utilização e aderência ao LOD encontramos campos de domínio altamente diversificados. Por exemplo, existem várias iniciativas de governo de países ao

redor do mundo, inclusive no Brasil, investindo nesta área de modo a abrir e ligar seus dados. Instituições privadas como New York Times, BBC e Globo também estão utilizando essas tecnologias. Os dados em LOD ainda são de predomínio do ambiente acadêmico, destacando-se os dados de ciência biomédica. O grau de amadurecimento da comunidade está cada vez maior o que motiva o estudo dessa recente e crescente área. As tecnologias e ferramentas estão em constante pesquisa e desenvolvimento de modo que evoluem e trazem mais usuários. Entretanto, ainda é necessário mais investimento expressivo de iniciativas privadas para alavancar as tecnologias de LOD.

O diagrama da nuvem de LOD (Lod Cloud)[39] é uma aplicação que desenha um grafo com muitos dos bancos de dados em RDF mais expressivos existentes. Existem versões do diagrama desde 2007 até 2011, possibilitando analisar visualmente como (quantitativa e qualitativamente) os bancos de dados em RDF evoluíram na Web de Dados.

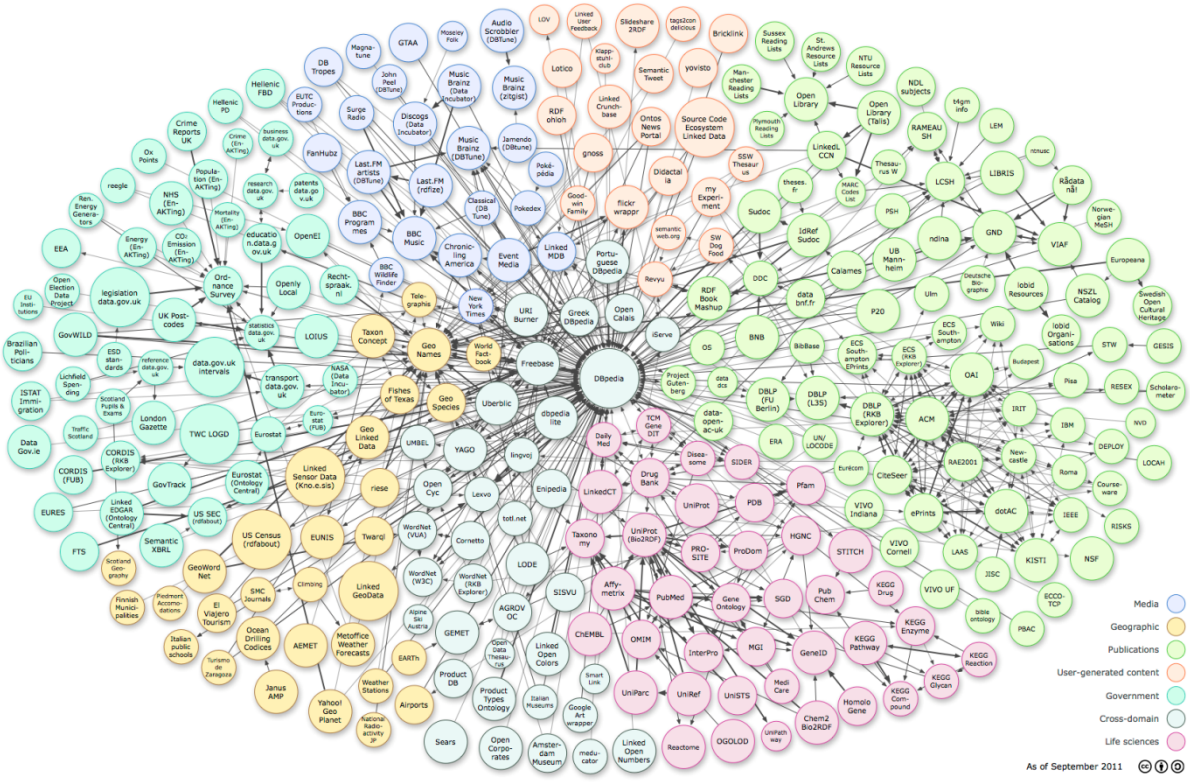


Figura 3 – Diagrama da nuvem de Linked Open Data, por Richard Cyganiak e Anja Jentzsch

2.5.2 *Mashups*

Uma das grandes vantagens de LOD está no poder de interoperar com os dados. *Mashup*, neste contexto, é o nome que se dá quando queremos cruzar dados de modo a gerar

[39] http://lod-cloud.net/

aplicações e relatórios úteis que dão suporte à tomada de decisão. Devido ao fato dos dados estarem todos ligados não só entre os dados do domínio, mas também com algum outro banco de dados em RDF existente na nuvem de LOD, é possível cruzar dados de natureza completamente diferentes e de fontes de dados completamente diferentes. Além disso, com muitas pessoas utilizando e interoperando os dados com diversificadas bases de dados, é possível criar aplicações e usabilidade para os dados nunca antes imaginada.

Um exemplo prático disso será analisado nas aplicações dos dados do projeto PingER Linked Open Data (seção 4.5). Serão cruzados, por exemplo, dados de medida de rede do PingER com dados na DBPedia sobre universidades e dados sobre investimento dos países em tecnologia.

2.5.3 Níveis de publicação de dados como Linked Open Data

Vimos a importância de publicar dados utilizando os padrões de LOD. Tim Berners-Lee (2009) criou uma convenção para classificar em níveis – como os dados podem ser publicados, introduzindo o conceito de **Dados Abertos 5 Estrelas:**

Quadro 1 - Dados Abertos 5 estrelas

★	Disponível na web (em qualquer formato), mas com uma licença aberta, para ser Open Data.
★★	Disponível como dados estruturados processáveis por máquina (ex. Excel ao invés de uma figura escaneada de uma tabela).
★★★	Todos acima, mais um formato aberto, sem licenças (ex: CSV ao invés de Excel).
★★★★	Todos os acima, mais o uso dos padrões abertos recomendados pelo W3C (RDF e SPARQL) para identificar coisas, de modo que outras pessoas possam apontar para os dados sendo publicados.
★★★★★	Todos os acima, mais ligar os dados sendo publicados aos dados de outras pessoas já na nuvem de LOD para prover contexto.

Além de publicar os dados observando o conceito de Dados Abertos 5 Estrelas, é importante verificar quão fácil é para outras pessoas descobrirem os dados publicados. W3C orienta que as URIs utilizadas como identificadores devem ser resolvíveis, isto é, devem ser URIs HTTP de modo que pessoas possam olhar esses identificadores (BERNERS-LEE, 2009).

Adicionalmente, SPARQLES, LOD Cloud e outras ferramentas para monitoramento de bancos de dados em RDF requerem que o BD triplificado esteja registrado no DataHub[40], que é uma plataforma de gerenciamento de dados livres (BUIL-ARANDA *et al.*, 2013).

Também se recomenda utilizar meios de descrever a base de dados para deixá-la mais fácil de ser descoberta pela comunidade; e isso é feito de forma padrão através do vocabulário VOID (2.4.3.1) e pela descrição do serviço SPARQL (2.4.3.2). Ademais, também é recomendável prover um *Dump RDF*, isto é, todos os dados da base em RDF em um dos formatos citados na seção 2.3.3.

2.5.4 Pilha do LOD2

Desde o surgimento dos conceitos de Dados Abertos, foram estabelecidos ciclos de vida para administrar as dificuldades enfrentadas durante o Processo de Publicação e interligação de dados na Web de Dados. Um ciclo de vida relevante foi definido pelo projeto LOD2, denominada Pilha do LOD2 (*LOD2 Stack*) (AUER *et al*., 2012a).

A Pilha do LOD2 consiste de uma arquitetura extensível formada por um conjunto de ferramentas distribuídas e integradas para apoiar todas as fases do ciclo de vida definido pelo projeto LOD2. Algumas dessas ferramentas são:

- D2R Server[41] - Utilizado para extrair e triplificar dados provenientes de banco de dados relacionais.
- OpenLink Virtuoso[42] – Sistema de Gerenciamento de Banco de Dados RDF.
- OntoWiki[43] - Ferramenta Wiki cujos recursos são utilizados para definição de autoria do conteúdo semântico.
- Silk[44] - Framework para descoberta e criação de interligações entre diferentes fontes de dados.
- Linked Data Integration Framework (LDIF)[45] – Integração de dados provenientes de diferentes conjuntos de dados da Web de Dados.

[40] http://datahub.io/about
[41] http://d2rq.org/d2r-server
[42] http://virtuoso.openlinksw.com/
[43] http://ontowiki.eu/Welcome
[44] http://lod2.eu/Project/Silk.html
[45] http://ldif.wbsg.de/

Figura 4 – Pilha do LOD2. Fonte: http://stack.lod2.eu/

O ciclo da Pilha do LOD2 é uma poderosa esquematização para publicar dados em LOD, baseando-se em um forte apoio ferramental pré-definido e que pode ser reutilizado. Contudo, este trabalho propõe, no próximo capítulo, um Processo de Publicação de Linked Open Data simplificado, elaborado empiricamente, abstraindo o ferramental técnico especializado, e que facilita o entendimento didático de cada uma das principais vantagens, distintamente, de publicar dados em LOD.

3 Processo de Publicação de Linked Open Data

Uma vez introduzida a ideologia da Web Semântica, estudado seus fundamentos e aspectos tecnológicos e, principalmente, entendido o conceito de *Linked Open Data*, podemos estabelecer uma visão prática do **Processo de Publicação de Linked Open Data**, proposto neste trabalho de conclusão de curso. Depois de construir o projeto PingER Linked Open Data, foram analisadas diversas dificuldades, foram estudadas várias tecnologias e boas práticas foram reunidas. A partir da reunião de todos esses tópicos e conceitos, análise de várias publicações e ferramentas, e muita dificuldade enfrentada para publicar os dados em LOD (algumas delas serão explicitadas nas próximas seções), foi desenvolvida uma proposta de esquematização de alto nível focando em didática e fácil entendimento. Essa esquematização também tem uma perspectiva essencialmente prática cujo objetivo é orientar o desenvolvimento de um projeto de publicação de dados em LOD. Algumas dessas orientações foram coletadas da comunidade e reunidas junto com outras orientações propostas após as dificuldades enfrentadas no desenvolvimento deste trabalho.

O processo pressupõe um domínio já pré-estabelecido, considera que o projeto de publicação em LOD está ainda na fase de concepção e orienta seguir as recomendações definidas.

O Processo de Publicação de LOD divide-se em um fluxo de cinco fases e cada uma enfatiza uma característica essencial dos conceitos de LOD. O quadro a seguir explicita e resume as características de cada fase.

Quadro 2 – As Fases do Processo de Publicação de LOD

1 Análise do Domínio
Base e suporte fundamentado e orientado ao negócio para todas as fases seguintes.

2 Engenharia da Ontologia
Foca em reúso, expressividade semântica do modelo, alta captura do conhecimento do domínio, possibilidade de interoperar com outras bases de dados em LO D e realizar infererências. Aspecto tecnológico enfatizado: OWL

3 Projeto de Triplificação
Processos técnicos de transformar ou gerar dados em formatos padrões para LOD, ligando os dados a outras bases nesse formato.
Aspecto tecnológico enfatizado: RDF

4 Acesso Público aos Dados
Publicar e expor os dados à comunidade
Aspecto tecnológico enfatizado: Serviços de protocolo SPARQL

5 Aplicações
Consumir os dados com a possibilidade de explorar outras diversidades de natureza de dados interligados (*mashups*).
Aspecto tecnológico enfatizado: Consultas SPARQL

A Figura 5 a seguir esquematiza todo o processo incluindo palavras-chave de cada fase e as principais interações entre as fases. Cada uma das setas, as quais representam interações entre as fases, serão explicadas no decorrer deste capítulo.

Figura 5 – Processo de Publicação de LOD

Esse processo é dito de alto nível porque é uma esquematização simplificada que mostra o Processo de Publicação de LOD a partir de um ponto de vista situado em uma camada superior de abstração omitindo os detalhes mais específicos inerentes de cada fase do processo. Devido à simplicidade e ao alto nível do esquema proposto, é possível utilizá-lo para explicar todo fluxo do ciclo de vida do Processo de Publicação de LOD mesmo para quem não tem domínio do assunto, sem mencionar as especificações tecnológicas de cada fase. Por isso, esquematizar de uma forma simples (apenas 5 fases com características bem específicas e distintas), visual e metodológica o Processo de Publicação é importante porque são muitas tecnologias, nomes, siglas e etc. envolvidas e tudo parece complexo demais.

Apesar desse processo poder ser explicado sem muitos detalhes, apenas enfatizando a principal característica e benefício proporcionado por cada fase, este capítulo descreve cada uma das 5 fases, sugerindo orientações com o objetivo de publicar dados em LOD. No capítulo 4 aplicaremos todo o processo ao cenário do PingER.

Observe que o processo é flexível o suficiente para prever trânsito de ida e volta entre as fases porque veremos que é comum, após avançar para uma fase, ter de voltar a anteriores. Além disso, como mencionado, várias recomendações e orientações serão destacadas. Entretanto, se algumas delas não forem possíveis ou muito difíceis de serem implementadas, a publicação final não deveria ser interrompida. Em adição, apesar do processo ter o objetivo de sistematizar a publicação de dados genéricos em LOD, cada caso, domínio e realidade tem suas peculiaridades que, eventualmente, pode inviabilizar a implementação de alguma orientação ou recomendação. A ideia do "melhor esforço possível" é muito válida em todo o processo.

3.1 Análise do Domínio

Giancarlo Guizzzardi (2000, p.33) enfatiza a importância da análise de domínio devido à necessidade de redução do alto custo desproporcional da manutenção de software em consequência de mudanças arbitrárias no projeto. Ademais, o autor destaca o desenvolvimento *para* reúso e *com* reúso.

Para um melhor entendimento desta fase do Processo de Publicação de LOD, é fundamental observar o conceito de **domínio do problema**.

Guizzardi (2000) explica que o domínio do problema "representa um conjunto de itens de informação presentes em um certo contexto do mundo real, interrelacionados [sic] de forma bastante coesa, e que desperta o interesse de uma certa comunidade". O autor ainda exemplifica um domínio de transporte aéreo. O conjunto de itens de informação a ser considerado seria voos, assentos na aeronave, relacionamento entre tripulação e voos, etc.

Neighbors (apud GUIZZARD, 2000) define **Análise de Domínio** como "uma tentativa de identificar os objetos, operações e relações entre o que peritos em um determinado domínio percebem como importante".

Investir tempo para estudar o domínio facilita o desenvolvimento de aplicações bem mais próximas da realidade do problema, aumentando a chance de satisfazer os objetivos do projeto. Além disso, diminui a necessidade de manutenção e alteração no software já que ele foi desenvolvido focado e orientado ao domínio e ao problema.

Um processo para análise do domínio pode ser descrito e as três principais subfases são resumidas desta maneira (GUIZZARDI, 2000):

a) Planejamento

É necessário (i) entender o domínio, isto é, suas características, limitações, restrições e o que os dados do domínio informam; (ii) definir as prioridades e as restrições do projeto; (iii) caracterizar o problema, definir o escopo (as definições e relevâncias) e a abordagem do projeto e como medir se o produto se adequa ao objetivo; e (iv) levantar os requisitos do domínio.

b) Aquisição e seleção dos dados

Os seguintes questionamentos devem ser levantados e, se possível, respondidos: Quais dados devem ser considerados no projeto? Serão todos os dados que o domínio disponibiliza ou apenas um subconjunto deles? Como se dá o acesso aos dados? Aproximadamente, qual são a qualidade e a quantidade dos dados?

c) Análise dos dados

Nesta subfase, uma análise mais apurada da qualidade e quantidade dos dados do domínio deve ser considerada. Para isso, incluem-se as seguintes atividades: (i) investigar a consistência, correção, limpeza e completude dos dados; (ii) tentar extrair as entidades, relações, funções, axiomas, regras de negócio, etc. e desenvolver um modelo conceitual prévio do domínio, caso não esteja disponível; e (iii) desenvolver, se não existir, um glossário definindo os termos utilizados no domínio.

Idealmente, os produtos da fase Análise do Domínio são: um documento com os requisitos do domínio, uma análise no mínimo superficial quantitativa e qualitativa dos dados, um primeiro rascunho do modelo conceitual do domínio e um glossário com os termos utilizados.

3.2 Engenharia da Ontologia

Após uma captura relevante do conhecimento do domínio (realizada na fase anterior), pode-se seguir para a próxima fase: a construção da ontologia do domínio. Como já vimos na seção 2.2, uma ontologia é utilizada como modelo de dados de referência para anotação semântica na Web Semântica. Já introduzimos os fundamentos de ontologias naquela seção e já destacamos sua importância. A fase de Engenharia da Ontologia no contexto de Processo

de Publicação de LOD também é de extrema significância e deve ser investido tempo considerável nesta fase. Dentre os pontos mais importantes, podem-se destacar alguns.

Assim como na Análise do Domínio, a Engenharia da Ontologia pode economizar muito tempo no futuro já que reduziria a manutenção do esquema. Mesmo que os esquemas de LOD sejam bem mais flexíveis que os esquemas tradicionalmente estruturados de bancos relacionais (como visto na seção 2.2.1), alterar um esquema de dados é raramente uma tarefa simples, principalmente se já houver uma quantidade imensa de dados baseados nele. Ressalta-se, porém, que mesmo que sejam tomados todos os cuidados durante a modelagem, é ainda provável que seja necessário alterar alguma característica do modelo identificada futuramente nas fases posteriores do processo.

A ideia de reúso é bem forte na fase de Engenharia da Ontologia. Existem muitos conceitos e termos que são comuns em uma grande quantidade de domínios, logo é bem provável que alguém no mundo já os modelou e possivelmente a modelagem se adequa, pelo menos em parte, ao domínio em questão. Por exemplo, conceitos geográficos (lugares, países, cidades, etc) ou conceitos temporais (ano, dia, hora, etc) possivelmente farão parte do domínio em questão. Além desses que são bem comuns, é possível que existam outras modelagens mais específicas ao domínio e que convenientemente se adequam. Logo, deve-se gastar tempo pesquisando ontologias ou modelagens conceituais relacionadas ao domínio a ser modelado no projeto. Dessa forma, o ontologista poderá analisar e comparar as modelagens já existentes e assim utilizar a mais adequada, de modo que modele a própria ontologia após pesquisar as que já existem. Finalmente, recomenda-se deixar para modelar somente os conceitos e relações que são específicos ao domínio do projeto. É possível, assim, partir de ontologias existentes e adaptá-las ao domínio.

Adicionalmente, o reúso de ontologias reforça a ideia de comunidade. Ao utilizar modelagens já existentes, contribui-se com uma padronização, facilita a comunicação e auxilia a consistência entre sistemas de Linked Open Data. Destacadamente, apoia a ideia de "interligados" dos conceitos de LOD, aumentando a interoperabilidade dos bancos de dados de triplas na nuvem. Entretanto, uma das desvantagens do reúso é que para aproveitar soluções já elaboradas e utilizá-las na nova solução, é preciso investir tempo tentando entender o que já foi construído e isso nem sempre não é uma tarefa trivial.

O ontologista deve também se atentar à expressividade semântica dos metadados que descrevem o domínio. Intuitivamente, nomes de entidades, relações e atributos mais significativos em relação ao domínio devem ser priorizados, pois facilitam o entendimento de humanos na hora do reúso, novamente incentivando a ideia de comunidade em LOD. Quão

mais expressivo – tanto em relação à semântica quanto à completude do modelo – mais próximo da realidade do domínio ele será.

Além desses fatores de reúso, comunidade e semântica, apesar da Engenharia da Ontologia tender a focar na modelagem conceitual do domínio, o esquema dos dados deve ser modelado levando em conta também como os dados RDF serão utilizados, de modo a observar o desempenho de consultas. Infelizmente ainda não há muitas publicações relacionadas a essa questão e muita pesquisa necessita ser desenvolvida. No entanto, experimentalmente, percebe-se que, além de outros fatores, o esquema de dados também influencia no desempenho da recuperação de dados RDF, especialmente utilizando consultas SPARQL.

É importante atentar ao paradoxo desempenho contra expressividade semântica quando se modela uma arquitetura que pode vir a ser processada por um computador. Essas duas características podem ser vistas como opostas, no sentido que, quanto mais semântico é o modelo (isto é, mais próximo da realidade), mais complexo e difícil de processar ele é. Por isso, ao realizar qualquer modelagem, é fundamental descrever o modelo de forma mais expressiva e próxima da realidade possível, mas deve-se sempre levar em consideração quão difícil será para o computador processar aquele modelo.

Em adição a esse paradoxo entre desempenho e expressividade semântica, é preciso observar a característica de generalidade de uma modelagem. Fazer um modelo genérico dos dados é normalmente muito vantajoso porque ele poderá ser usado de diversas formas, por diversas pessoas, realidades, etc. Entretanto, ao se modelar muito genericamente, os conceitos começam a se distanciar da realidade do domínio, comprometendo a semântica dos dados. Além disso, é possível que a modelagem fique muito complexa ou simples demais (no sentido de seu entendimento e por esconder detalhes da realidade, respectivamente), afetando não só a semântica, mas também o desempenho de processamento do modelo por um computador.

A própria ontologia (geralmente um ou mais arquivos OWL) é o produto desta fase. Ademais, junto com a ontologia, é importante existir algum documento que seja de fácil compreensão por humanos e que a descreva em detalhe (as classes, as propriedades, a taxonomia, etc), explicite os outros modelos que foram incorporados no processo de reúso de ontologias existentes e também mencione quais outros BDs RDF da nuvem de LOD estão sendo ligados (através dos vocabulários ou ontologias reutilizados). Ao preocupar-se com a documentação da ontologia modelada, auxiliará outras pessoas que poderão ou não reutilizá-las, ou consumir os próprios dados do BD RDF, ou ligar os dados do projeto a dados de outros BDs na nuvem de LOD. Na prática, a grande maioria dos BDs RDF, apesar de

extremamente ricos em dados úteis, não proveem essa documentação, fazendo com que os desenvolvedores de projetos de LOD percam tempo utilizando técnicas de engenharia reversa para descobrir o modelo de dados utilizado por eles. Resumidamente, documentar a ontologia é essencial.

Quanto às interações com outras fases no Processo de Publicação de LOD, o fluxo ideal é vir da fase Análise do Domínio para a fase Engenharia da Ontologia e então ir para a fase seguinte (Projeto de Triplificação, seção 3.3). Entretanto, na prática, é muito comum durante a fase de Engenharia da Ontologia voltar para a fase de Análise do Domínio para coletar algum conhecimento extra (ou que estava faltando) ou de alguma das fases posteriores ter de voltar para essa para rever o modelo.

3.3 Projeto de Triplificação

Triplificação é o nome que se dá ao processo de transformar ou gerar dados no formato de triplas RDF (conforme descrito na seção 2.3) ou de instanciar os indivíduos utilizando a ontologia do domínio. Este processo costuma ser relativamente tão complexo e trabalhoso que é conveniente estabelecer um projeto somente para realizá-lo.

Desde o princípio do Projeto de Triplificação, é importante atentar-se às orientações de publicar dados como LOD, vistas na seção 2.5.3, especialmente tendo a precupação de viabilizar como os dados sendo triplificados possam ser ligados a outras bases já existentes. Recomenda-se utilizar o `sameAs` da OWL para ligar indivíduos a outras bases. Também se ressalta nesta fase a importância da definição de padrões de URIs (seção 2.2.2); foi visto que é recomendável que se use URIs HTTPs e um processo sistemático para sua geração.

Essencialmente, pode-se dividir esta fase em duas subfases:

a) Escolha do Sistema de Gerenciamento de Banco de Dados RDF

Inicialmente, é necessário escolher o SGBD RDF a ser utilizado. Assim como nas fases anteriores, é importante gastar tempo pesquisando as soluções existentes. Neste caso, realizar um levantamento dos repositórios disponíveis e eleger o mais adequado ao projeto.

Existem diversos SGBDs RDF: Jena[46], Sesame[47], Virtuoso[48], AllegroGraph[49], etc. Todos eles têm em comum a tarefa de implementar as funções CRUD sobre dados RDF.

[46] http://jena.apache.org/
[47] http://www.openrdf.org/about.jsp
[48] http://virtuoso.openlinksw.com/
[49] http://www.franz.com/agraph/

Porém, o que os diferencia, dentre outros fatores, é como eles indexam as triplas na camada física de dados e como realizam as otimizações de consultas e carga dos dados. Essa divergência entre as implementações causa diferença de desempenho e deve ser observada.

Embora não tenha ainda um padrão de métricas para qualificar um SGBD RDF, em geral, costuma-se, dentre outros aspectos, verificar a quantidade de dados que um repositório de triplas consegue lidar. Isso costuma ser especialmente crítico para projetos com grandes volumes de dados a pubicar.

Consequentemente, é preciso fazer uma análise quantitativa orientada ao número de triplas, utilizando o que foi capturado na fase de Análise do Domínio para chegar a uma estimativa do quanto será inserido no BD RDF. Idealmente, a estimativa deve ser precisa pelo menos na ordem de milhão. Os experimentos publicados realizaram testes sobre *datasets* de pelo menos 1 milhão de triplas então não há dados suficientes para inferir se a escolha do SGBD RDF interfere caso o número de triplas seja menor que 1M (BIZER; SCHULTZ, 2010; BIO ONTOLOGY, 2010). Entretanto, como visto na seção 2.3.1, por causa da propriedade granular das triplas, para descrever um domínio e instanciar seus registros utilizando esse padrão, é muito comum ter BDs que superam 1M de triplas. Os resultados dos experimentos mostram que existem repositórios que apresentam desempenho muito superior a outros, dependendo do número de triplas. Especialmente, Berlin SPARQL Benchmark (2013) faz experimentos de grandes quantidades de dados sobre SGBDs RDFs e recentemente foi publicada uma extensa e bem descrita comparação focada em desempenho entre os mais populares.

Adicionalmente, o W3C publicou um sumário[50] dos repositórios existentes que foram capazes de lidar com número de triplas realmente gigantes (alguns da ordem de bilhão e até trilhão).

Portanto, escolher bem o repositório de triplas é uma importante tarefa que também pode economizar tempo no futuro. Experimentalmente, depois de basear-se no SGBD escolhido e instanciar uma quantidade imensa de triplas no repositório, trocar de SGBD não costuma ser uma tarefa trivial.

b) Carregando o repositório

Uma vez escolhido o SGBD RDF, começa a subfase de triplificação propriamente dita.

[50] http://www.w3.org/wiki/LargeTripleStores

Se os dados não estiverem disponíveis em algum formato, eles devem ser gerados no padrão RDF. Normalmente, existe algum processo automatizado que vai gerar os dados; basta adaptá-lo para que o *output* do processo seja dados formatados no padrão necessário.

Senão, se os dados já existirem em alguma outra fonte e estiverem em algum outro formato, é necessário aplicar um processo fundamental bem difundido no mundo de banco de dados, especialmente em *data warehousing*: Extração, Transformação e Carga (ETC) (KIMBALL; CASERTA, 2004).

O processo, idealmente totalmente automático, se dá da seguinte maneira:

- Inicialmente, os dados precisam ser extraídos utilizando algum mecanismo. Por exemplo, se os dados estiverem em uma base de dados relacional, realizar consultas SQL sobre ela; se os dados estiverem em alguma base semi-estruturada (em um banco XML ou arquivos que seguem algum padrão conhecido, como CSV), utilizar mecanismos que recuperem esses dados; se os dados estiverem em texto corrido, não-estruturado, utilizar técnicas de mineração textual para recuperar os dados. Enfim, um processo de extração deve ser estabelecido dependendo do projeto e de como os dados estão disponíveis. Técnicas já existentes devem ser combinadas com técnicas desenvolvidas especificamente para a solução. A qualidade dos dados extraídos também deve ser observada; possivelmente um processo de limpeza dos dados ocorrerá.
- Uma vez extraídos, os dados devem ser transformados no padrão RDF seguindo a ontologia estabelecida.
- Finalmente, os dados são carregados no repositório de triplas.

Vale observar que reúso é importante mais uma vez também nesta fase porque já existem ferramentas estáveis que realizam ETC inclusive no mundo de Web Semântica, como a ETL4LOD[51], produzida pelo GRECO-UFRJ, onde a ferramenta Kettle da suite do Pentaho foi estendida com *plugins* específicos para tratamento de dados RDF e acesso via SPARQL. Logo, pesquisar soluções existentes aplicáveis e adaptáveis ao projeto deve ser considerado.

Em adição à carga inicial dos dados, é comum ter BDs que são frequentemente alimentados ao longo do tempo. Portanto, é importante desenvolver um projeto de triplificação sistemático e o mais automático possível para instanciar triplas de acordo com a necessidade.

O produto desta fase é o banco de dados RDF.

[51] http://greco.ppgi.ufrj.br/lodbr/

Quanto às interações com outras fases no Processo de Publicação de LOD, o fluxo ideal é vir da fase Engenharia da Ontologia e então ir para a fase seguinte (Acesso Público aos Dados, seção 3.4). Entretanto, na prática, é muito comum durante a fase de Projeto de Triplificação voltar para a fase de Engenharia da Ontologia e remodelar algum detalhe que não foi bem analisado ou esquecido. Enfatiza-se que alterar o modelo do domínio e redesenhar a ontologia não é tão custoso quanto em ambientes muito estruturados (como em BDs relacionais) uma vez que os esquemas são bem mais flexíveis. Além disso, outro fator que faz voltar para a fase anterior é o mau desempenho ocorrido por algum detalhe na modelagem que precisará ser revisto. Ademais, voltar direto para a Análise do Domínio também é possível para rever algum conceito do domínio ou para rever a quantidade, qualidade ou até seleção dos dados.

3.4 Acesso Público aos Dados

A quarta fase do Processo de Publicação de LOD é a que destaca o Acesso Público aos Dados, enfatizando a abertura (o "*open*" dos conceitos de LOD) e a facilidade de acesso, utilizando padrões conhecidos. Além dos dados terem sido gerados na fase anterior (de Triplificação) conforme as orientações de publicação de LOD vistas na seção 2.5.3, é boa prática estabelecer um SPARQL Endpoint (estudado na seção 2.4.2) público que permita um acesso fácil e padronizado à base de dados em RDF. Também foi visto (seção 2.5.3) que se deve oferecer um *Dump RDF* de toda a base e utilizar descritores da base de dados (vocabulário VOID – seção 2.4.3.1) e do SPARQL Endpoint (*Service Description* – seção 2.4.3.2).

Para facilitar o acesso aos dados bem como a compreensão do modelo por detrás, nesta fase destaca-se também a necessidade de publicar os produtos da fase Engenharia da Ontologia. Tornar a ontologia acessível e aberta (publicar os arquivos fonte em OWL) bem como sua documentação, incluindo um glossário dos termos utilizados nela, fortalece a comunidade de LOD e facilita o reúso dos dados. Vimos na seção 3.2 por que é muito útil documentar a ontologia. Lembrando que se os arquivos OWL estiverem publicamente disponíveis, alguém que for consumir os dados poderá, além de analisar o código-fonte, abri-los em algum visualizador de ontologias para entender o modelo do domínio. As versões mais recentes do Protégé, além de outras funções vistas na seção 2.2.5, conta com um bom visualizador de ontologias.

Um *website* que reúna o *link* para o SPARQL Endpoint público, *links* públicos para o código-fonte da ontologia e sua documentação e para um RDF *Dump* da base são idealmente produtos desta fase.

Quanto às interações com outras fases no Processo de Publicação de LOD, o fluxo ocorre vindo da fase de Projeto de Triplificação e, quando o BD RDF é estabelecido, ir para a fase de Acesso Público aos Dados. Após publicar os dados e informações, pode-se partir para a última fase, a de Aplicações.

3.5 Aplicações

Finalmente, a quinta e última fase do processo baseia-se em consumir os dados em LOD. Com a base de dados RDF e o SPARQL Endpoint, pode-se construir consultas SPARQL para recuperar os dados e aproveitar todas as vantagens proporcionadas pelo fato dos dados estarem em LOD (vistas na seção 2.5.1). Em especial, destaca-se a interoperabilidade com outros bancos em LOD de natureza completamente diferentes, através de *Mashups,* como visto na seção 2.5.2. Ademais, devido à propriedade granular das triplas (seção 2.3.1) e devido à estruturação dos esquemas providos pela ontologia, podemos gerar consultas SPARQL detalhadas para recuperar informação significativamente específica e precisa. Uma vez recuperados os dados em LOD, pode-se gerar relatórios, *dashboards* e visualizações inteligentes nunca antes imaginadas e que suportam tomadas de decisão. Algumas dessas aplicações serão exemplificadas na seção 4.5 utilizando o cenário desta monografia.

Os produtos desta fase são as aplicações desenvolvidas sobre os dados em LOD do projeto.

Quanto às interações com outras fases no Processo de Publicação de LOD, o fluxo ideal é vir da fase Acesso Público aos Dados e fechar o ciclo, produzindo as aplicações desejadas. Entretanto, na prática, somente durante a execução de consultas SPARQL complexas é quando realmente experimentamos e testamos o desempenho do projeto. Logo, se o desempenho não for satisfatório, será necessário voltar para uma das fases anteriores para investigar as causas:

a) Voltar à fase Projeto de Triplificação: se as consultas SPARQL demorarem muito, talvez seja necessário voltar ao Projeto de Triplificação para escolher um outro SGBD RDF. Ou, pode também perceber algum padrão de consulta que torna a execução muito mais lenta então talvez tenha que reinstanciar as triplas de modo a

não utilizar esse padrão. Entre outras situações que prejudicam o desempenho e que possivelmente é causado por como as triplas são instanciadas e armazenadas.

b) Voltar à fase da Engenharia da Ontologia: desempenho, inconsistências, falhas de classificação e modelagem, entre outros problemas podem ser identificados na fase de Aplicação e então é necessário voltar para a fase de Engenharia da Ontologia para remodelar o domínio e o esquema, o que, como mencionado na seção 3.2.1, não costuma ser uma tarefa simples.
c) Voltar à fase Análise do Domínio: isso pode ocorrer porque é possível que a aplicação não satisfaça o requisito do cliente. Neste caso, é necessário voltar para a primeira fase para tentar entender exatamente o que o cliente precisa. Como é de se imaginar, essa pode ser a transição mais custosa já que existe a possibilidade de ter que alterar a modelagem, o processo de triplificação, as instâncias dos dados, etc. Essa transição deve ser evitada ao máximo e a probabilidade dela ocorrer diminui se cada fase do processo foi desenvolvida adequadamente.
d) Voltar à fase de Acesso Público aos Dados: caso algum dos produtos anteriores seja alterado (ex.: o BD RDF ou a ontologia), será necessário republicá-los.

4 Processo de Publicação de LOD aplicado ao PingER

As fases do processo de Publicação de LOD foram esquematizadas após um árduo trabalho com diversas tentativas e erros que poderiam ser evitados caso as fases fossem seguidas conforme as orientações, desde o início.

A maioria dos conceitos, discussões e conclusões apresentadas neste capítulo foi obtida no final do projeto. Até chegar aos resultados finais, muitas propostas de soluções surgiram e frequentemente foi necessário voltar uma ou mais fases para rever e, às vezes, até refazer tudo. Como vimos no capítulo 3, é normal voltar algumas fases e transitar entre elas. Entretanto, neste capítulo, mostramos, de forma linear, apenas as conclusões e os produtos de cada fase. Isto é, o Processo de Publicação de LOD para o projeto PingER é apresentado, começando na fase inicial de concepção e avançando fase a fase até chegar à final, das aplicações dos dados em LOD. No final do capítulo, discutimos, ainda, algumas transições não ideais entre as fases do processo.

4.1 Análise do domínio do Projeto PingER

4.1.1 Planejamento do Projeto PingER Linked Open Data

Como parte do planejamento de um projeto, é necessário antes entender o domínio, definir o problema e tentar elaborar uma abordagem de solução.

Conforme já mencionado na seção 1.3.2, o Projeto PingER coleta dados sobre qualidade da Internet desde 1998 até os dias correntes, medindo informação sobre cerca de 80 nós monitores e mais de 800 nós monitorados, cerca de 8200 pares[52] (porque nem todo monitor monitora todos os nós monitorados) em mais de 160 países, podendo informar a quase totalidade da população mundial conectada à Internet sobre a qualidade da rede. Toda medição se dá com um *ping* (o glossário completo com esse e outros termos estão no Apêndice A) em relação a um par de nós, o nó monitor e o nó monitorado, em relação a um intervalo de tempo e em relação a um tipo de métrica (PingER utiliza mais de 10 métricas) (COTTRELL; MATTHEWS).

Disso, depreende-se que o projeto PingER mantém uma enorme quantidade de dados. Esses dados são armazenados em um servidor, em diversos arquivos em formato de texto corrido, extensão *.txt*.

[52] http://www-wanmon.slac.stanford.edu/cgi-wrap/dbprac.pl?monalias=all

Esses arquivos possuem um padrão de nomes que identifica o fato sendo medido. Por exemplo, o arquivo `throughput-100-by-node-allyears.txt` contém dados sobre a métrica de rede *throughput*, para envio de pacotes de dados de tamanho 100 bytes, agrupados por ano, para todos os anos e por nós de rede. Para um outro fato, digamos, "recuperar todas as medidas da métrica *pacotes perdidos* que aconteceram no ano de 2008", o nome do arquivo seria condizente com os parâmetros especificados. Vide Anexo 1 para um exemplo de um desses arquivos .txt.

Antes deste trabalho, a maneira mais fácil de recuperar esses dados era através de uma aplicação chamada Pingtable[53]. Ela provê uma interface web amigável para recuperar os dados brutos do PingER, a saber, os arquivos .txt mencionados acima. Um interessado nos dados especifica os parâmetros a serem utilizados na recuperação dos dados desejados (Figura 6), a aplicação busca o arquivo condizente com os parâmetros especificados (seguindo o padrão mencionado), e processa os valores separados por espaço (CSV com delimitador espaço em branco), mostrando-os em uma tabela comum das páginas web clássicas, no formato HTML (Figura 7).

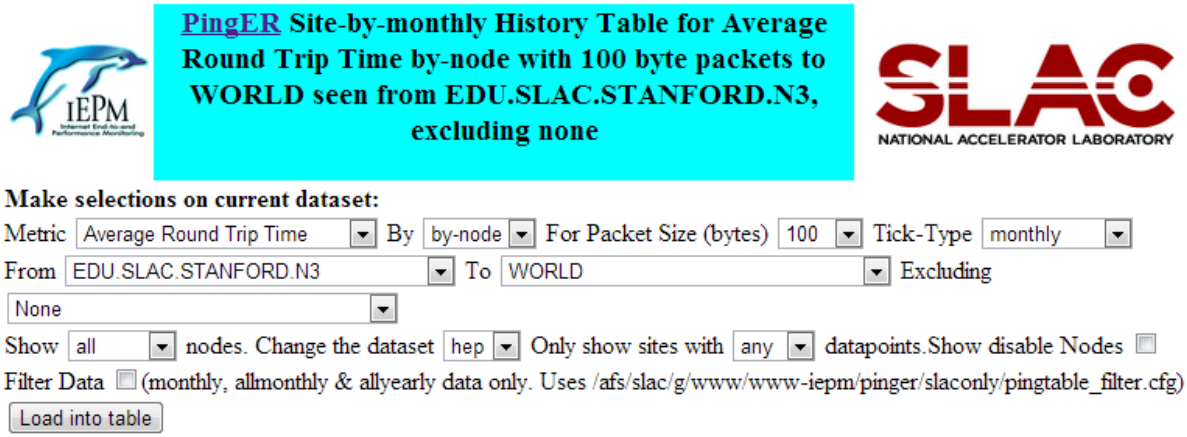
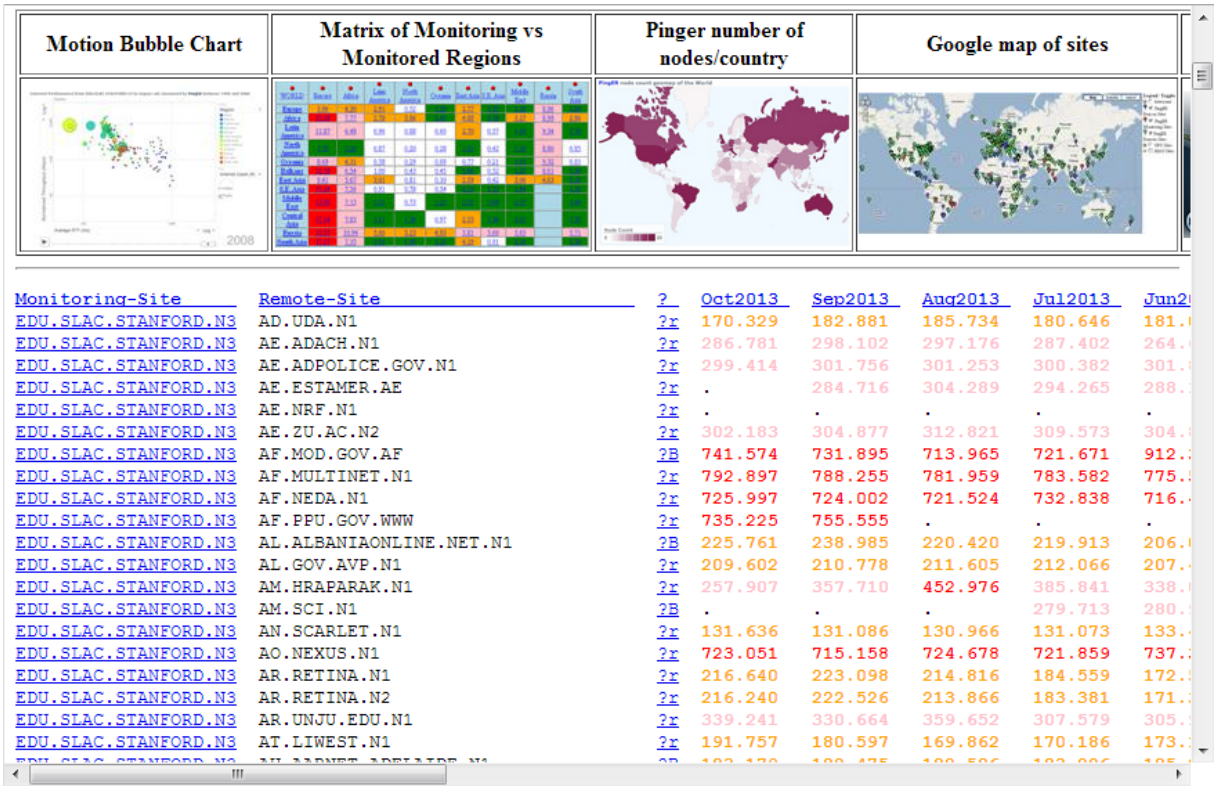


Figura 6 – Interface HTML da especificação dos parâmetros para a Pingtable

[53] http://www-wanmon.slac.stanford.edu/cgi-wrap/pingtable.pl



Monitoring-Site	Remote-Site	?	Oct2013	Sep2013	Aug2013	Jul2013	Jun2
EDU.SLAC.STANFORD.N3	AD.UDA.N1	?r	170.329	182.881	185.734	180.646	181.
EDU.SLAC.STANFORD.N3	AE.ADACH.N1	?r	286.781	298.102	297.176	287.402	264.
EDU.SLAC.STANFORD.N3	AE.ADPOLICE.GOV.N1	?r	299.414	301.756	301.253	300.382	301.
EDU.SLAC.STANFORD.N3	AE.ESTAMER.AE	?r	.	284.716	304.289	294.265	288.
EDU.SLAC.STANFORD.N3	AE.NRF.N1	?r	.	.	.	.	.
EDU.SLAC.STANFORD.N3	AE.ZU.AC.N2	?r	302.183	304.877	312.821	309.573	304.
EDU.SLAC.STANFORD.N3	AF.MOD.GOV.AF	?B	741.574	731.895	713.965	721.671	912.
EDU.SLAC.STANFORD.N3	AF.MULTINET.N1	?r	792.897	788.255	781.959	783.582	775.
EDU.SLAC.STANFORD.N3	AF.NEDA.N1	?r	725.997	724.002	721.524	732.838	716.
EDU.SLAC.STANFORD.N3	AF.PPU.GOV.WWW	?r	735.225	755.555	.	.	.
EDU.SLAC.STANFORD.N3	AL.ALBANIAONLINE.NET.N1	?B	225.761	238.985	220.420	219.913	206.
EDU.SLAC.STANFORD.N3	AL.GOV.AVP.N1	?r	209.602	210.778	211.605	212.066	207.
EDU.SLAC.STANFORD.N3	AM.HRAPARAK.N1	?r	257.907	357.710	452.976	385.841	338.
EDU.SLAC.STANFORD.N3	AM.SCI.N1	?B	.	.	.	279.713	280.
EDU.SLAC.STANFORD.N3	AN.SCARLET.N1	?r	131.636	131.086	130.966	131.073	133.
EDU.SLAC.STANFORD.N3	AO.NEXUS.N1	?r	723.051	715.158	724.678	721.859	737.
EDU.SLAC.STANFORD.N3	AR.RETINA.N1	?r	216.640	223.098	214.816	184.559	172.
EDU.SLAC.STANFORD.N3	AR.RETINA.N2	?r	216.240	222.526	213.866	183.381	171.
EDU.SLAC.STANFORD.N3	AR.UNJU.EDU.N1	?r	339.241	330.664	359.652	307.579	305.
EDU.SLAC.STANFORD.N3	AT.LIWEST.N1	?r	191.757	180.597	169.862	170.186	173.

Figura 7 – Resultados de medidas mostrados na Pingtable

Entretanto, utilizar esse sistema de armazenamento de dados, mesmo já apresentando uma considerável padronização, está muito aquém do uso de um sistema de banco de dados, cujas funcionalidades e benefícios são já amplamente reconhecidos (NAVATHE; ELMASRI, 2010). Devido à falta de muitas dessas facilidades, tanto a recuperação de dados daqueles arquivos quanto sua gerência não é muito efetiva, limitando ou dificultando por demais a manutenção e a realização de análises mais elaboradas sobre os dados.

Adicionalmente, existe a grande dificuldade de interoperar e intercambiar os dados do PingER com dados de outras fontes. Como visto anteriormente no capítulo de introdução (seção 1.3.2), esses dados podem ser extremamente úteis e aplicáveis a diversas áreas e situações. Se eles seguissem algum padrão de mais amplo uso, seria mais viável sua exploração conjunta com outros dados, diversificando ainda mais as possibilidades de análises. Além desses aspectos, ao utilizar um padrão comum e aberto para a comunidade, possivelmente muito mais pessoas iriam consumi-los, possibilitando, ainda, usos não antecipados desses dados.

Portanto, o problema alvo deste projeto pode ser assim enunciado: os dados do PingER não são armazenados de uma forma facilmente acessível, não sendo também uma forma padrão. Por isso, gerar relatórios e visualizações inteligentes dos dados é uma tarefa

difícil, principalmente se essas tarefas estiverem associadas a dados de naturezas diferentes e se esses dados forem abertos para mais ampla utilização.

Frente ao problema assim definido, o trabalho tem como objetivo disponibilizar os dados do PingER de uma maneira eficiente, fácil, padronizada e que seja útil a muitas pessoas, além dos próprios integrantes do PingER. A abordagem de solução contempla a utilização das estratégias e técnicas de Web Semântica e LOD para publicar os dados, seguindo as orientações já apontadas. Permitindo, dessa forma, uma utilização mais ampla e efetiva, tornando mais fácil o cruzamento com dados de outras fontes.

4.1.2 Seleção dos dados

Para o desenvolvimento da abordagem, foi necessário selecionar os dados a serem tratados e definir como acessá-los.

Primeiro, analisamos como os dados do PingER podem ser recuperados para serem consumidos pela aplicação que os converterá em formato LOD. Foi verificado que a Pingtable executa HTTP GETs no servidor para recuperar o arquivo .txt referente ao determinado cruzamento de parâmetros para então mostrar os dados do arquivo .txt em um documento HTML. Portanto, é possível desenvolver uma aplicação que tenha o mesmo mecanismo da Pingtable, isto é, que execute HTTP GETs para recuperar os arquivos .txt (em formato CSV) e processá-los. Ficou, então, definida a maneira de se recuperar os dados do PingER.

Segundo, foi preciso selecionar os dados a serem considerados no projeto. Como salientado na subseção anterior, o projeto PingER envolve uma quantidade enorme de dados, tornando impraticável, pelo prazo de desenvolvimento estabelecido, processar e converter todos para o formato desejado. Daí, junto com o líder do PingER, ponderamos qual a prioridade dos dados, isto é, que tipo de dados seriam mais importantes e relevantes. Com isso, foi estabelecido o seguinte escopo:

- Em relação ao escopo de nós da rede: medições de qualidade de rede entre todos os pares de nós de rede (i.e., *ping* de todos os nós monitores para todos os nós monitorados);
- Em relação ao nível de detalhe geográfico: apenas dados de nó para nó, não contemplando dados agregados de região (cidade, país, continente, etc.) para região. Se o cliente quiser dados em relação a um determinado país, por exemplo, basta executar uma consulta que agrega (soma, calcula média, etc.) todos os dados de medida de nós de rede localizados naquele país;

- Em relação ao nível de detalhe temporal: o menor grão (dados em maior detalhe) temporal seria dia**,** embora o PingER ofereça menor grão em hora. Foi analisado que processar dados de todas as 24 horas, de todos os dias, de todos os meses, de todos os anos seria impraticável;
- Em relação ao tipo de métrica: o PingER utiliza 16 métricas associadas à qualidade da Internet. Porém, apenas as 11 mais relevantes para o cliente foram selecionadas para serem consideradas: *Mean Opinion Score, Directivity, Average Round Trip Time, Conditional Loss Probability, Duplicate Packets, Inter Packet Delay Variation, Minimum Round Trip Delay, Packet Loss, TCP Throughput, Unreachability, Zero Packet Loss Frequency.* [54] A definição desses termos encontra-se no glossário do PingER LOD no Apêndice A;
- Em relação ao tamanho do pacote: o PingER considera *pings* de pacotes de tamanho 100 e 1000 bytes. Foi verificado, junto com o líder do PingER, que escolher somente um desses tamanhos já era o suficiente para resolver o problema. Escolhemos considerar apenas pacotes de tamanho 100 bytes.

4.1.3 Análise dos dados do Projeto PingER

Foi necessário elaborar um primeiro rascunho do modelo conceitual dos dados do domínio. Pelas características do PingER, percebeu-se uma natureza de dados passíveis de manipulação em umérica e estatística os quais são gerados a partir de cruzamento de parâmetros (período de tempo, região, tipo de métrica de rede, etc). Foi visto que os dados são acumulados ao longo dos anos e são utilizados para realizar estudos que se relacionam à qualidade de rede e ainda dão suporte à tomada de decisão. Dados com essas características são amplamente estudados e já existem modelos clássicos para descrevê-los e representá-los. Um dos modelos conceituais mais usados em dados dessa natureza é resultante dos estudos de *Data Warehousing* (DW), conhecido como esquema estrela (*star schema*).

Resumidamente, um esquema estrela baseia-se em uma visão multidimensional dos dados na qual o valor sendo medido é comumente visualizado em uma posição central (em DW, chamado de **tabela fato**) e os parâmetros (chamados de **dimensões**), que o definem ficam em posições satélites formando um esquema análogo a uma estrela (LEARN DATA MODELING, 2012).

[54] Este capítulo utiliza nomes em Inglês nos glossários, diagramas, figuras e telas para manter compatibilidade com os dados do sistema do PingER.

Para o PingER, uma medida de rede é calculada a partir de um *ping* e refere-se ao cruzamento dos parâmetros: quando a medida ocorreu, onde ela ocorreu (de qual nó para qual nó) e o que a medida mediu (qual métrica de rede – *throughput*, quantidade de perda de pacotes, etc). A partir dessas ideias, foi elaborada uma modelagem conceitual simples (Figura 8) para o domínio do PingER, baseando-se no esquema estrela.

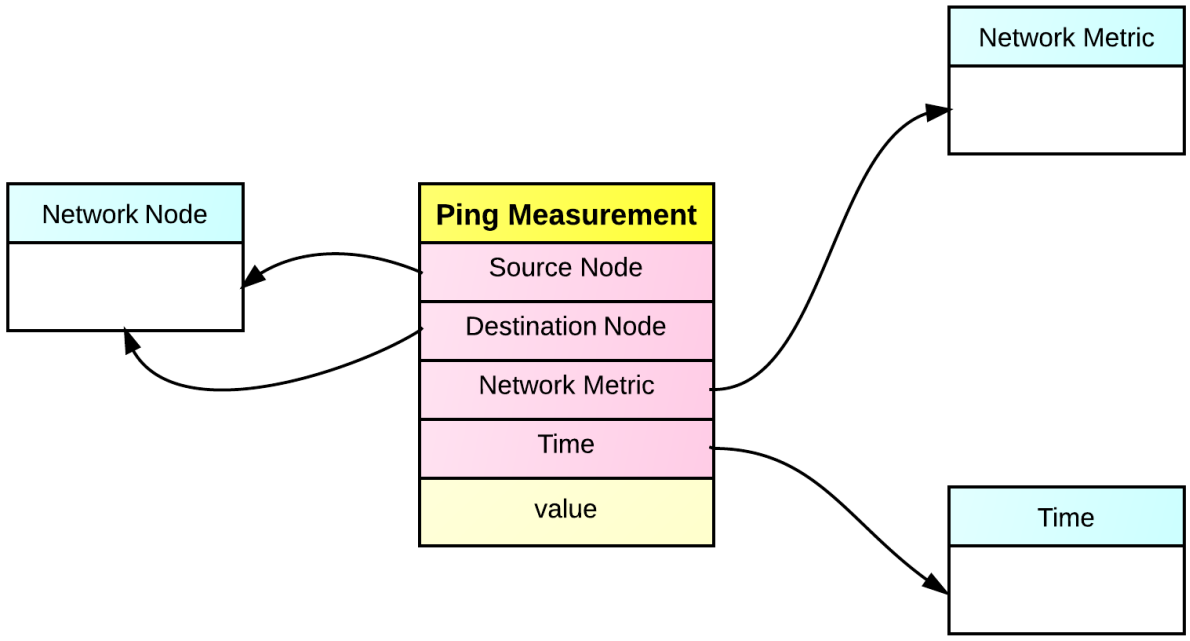


Figura 8 – Primeiro rascunho do modelo conceitual do PingER baseado em um esquema estrela

Em adição ao primeiro modelo conceitual do domínio, tentou-se extrair relações, conceitos e entidades e elaborar um glossário com a definição dos termos mais recorrentes do PingER. O Apêndice A contém o glossário construído a partir da coleta de conhecimento do domínio, realizada juntamente com o dono do PingER.

Quanto à qualidade dos dados, felizmente não tivemos maiores problemas. O PingER tem um conjunto de dados bem completo e controlado. Quando faltavam dados era porque realmente nunca houve medições para determinado cruzamento de parâmetros. Os arquivos .txt recuperados via HTTP GET já eram tratados e, por estarem em formato CSV, não seria necessária nenhuma técnica complexa de mineração textual para processá-los.

4.2 Engenharia da ontologia do projeto PingER Linked Open Data

A partir do conhecimento adquirido sobre o domínio do PingER na fase de Análise do Domínio, prossegue-se para a fase onde uma modelagem mais completa deve ser realizada, na forma de uma ontologia, levando atentamente em consideração todas as questões levantadas

na seção 3.2. Resumidamente, essas questões são: reúso, interoperabilidade, expressividade semântica, completude (quão completo é o modelo em relação à realidade) e desempenho das consultas.

Considerando o requisito de reúso e todas suas vantagens, buscamos ontologias existentes relativas ao domínio de medidas de rede de computadores.

4.2.1 Ontologia MOMENT

O projeto "Monitoring and Measurement in the Next generation Technologies" (MOMENT), parte do Seventh Framework Programme for Research (FP7), constitui um grande investimento da União Europeia em desenvolvimento tecnológico (EUROPEAN COMMISSION, 2012). O projeto tem por objetivo integrar infraestruturas de medida e monitoramento (RAO, 2010), promovendo a definição de formato de dados e desenvolvimento de interfaces unificadas para aumentar flexibilidade do design de aplicações de Internet do futuro. Adicionalmente, o projeto permite a representação semântica integrada e recuperação de medidas e monitoramento de rede de computadores (HIGH PERFORMANCE COMPUTING AND NETWORKING, 2010). Um dos produtos deste projeto é a própria ontologia em arquivos OWL[55].

A ontologia foi descrita em uma Especificação de Grupo (*Group Specification*) e publicada pela European Telecommunications Standards Institute (2012). Além disso, Alfredo Salvador *et al.* (2010) publicaram um artigo no qual a ontologia é explicitada. Ademais, apresentações[56,57] também auxiliam a descrever o modelo de dados de uma abordagem semântica para medida de tráfego de rede. Nesta seção, após explicar a ontologia MOMENT aplicada ao domínio do PingER, sua utilização (ou não) será avaliada.

A ontologia MOMENT é realmente complexa, talvez por pretender ser genérica de modo a contemplar as principais características que se referem a medidas de rede. Analisando o material disponível sobre a ontologia e os arquivos OWL, conclui-se que o modelo baseia-se em uma estrutura que abstrai detalhes específicos do domínio e utiliza essencialmente somente duas superclasses para descrever qualquer tipo de medidas e monitoramento de tráfego de rede, que é o caso do PingER. A propósito, devido à característica generalista da ontologia, é conveniente introduzir a ideia de subdomínio do domínio de medidas e monitoramento rede. Especialmente o caso do PingER, o subdomínio se restringe a medir

[55] https://svn.fp7-moment.eu/svn/moment/public/Ontology/
[56] http://goo.gl/eXuiM0
[57] http://goo.gl/cNOENG

pings entre nós fonte e nós destino. A Figura 9 mostra o esquema no qual toda a ontologia se baseia.

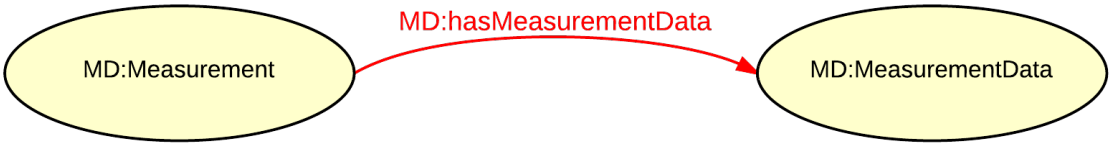


Figura 9 – Estrutura fundamental da ontologia MOMENT (EUROPEAN TELECOMMUNICATIONS SI, 2012)

Todos os prefixos (*namespaces*) mencionados neste capítulo, principalmente nas figuras das modelagens, encontram-se no Apêndice B. Os prefixos que possuem valor "`PingER-ont`" são referentes a modificações ou criações de conceitos da ontologia que foram necessários para atender às necessidades do domínio do projeto. A Figura 10 mostra a legenda de todos os conceitos de ontologia (classe, object property, datatype property, equivalências, etc.) utilizados nos diagramas desta monografia.

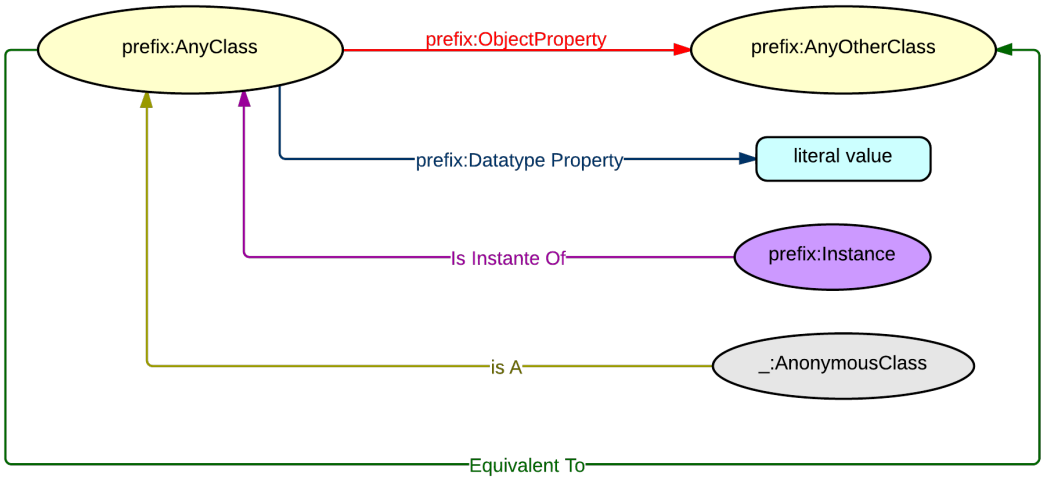


Figura 10 – Legenda de todos os conceitos utilizados nos diagramas das ontologias nesta monografia

Apesar da complexidade da ontologia MOMENT, ela se baseia essencialmente em duas superclasses somente: `Measurement` e `MeasurementData`, com a object property `hasMeasurementData` que as relaciona. Apenas com essas duas classes e essa relação, é possível modelar quase que o domínio inteiro com a ideia "uma medida tem dados", isto é, uma instância da classe `Measurement` tem dados (`hasMeasurementData`) instâncias da classe `MeasurementData`. Os dados da medida se relacionam principalmente a subclasses de `MeasurementData`.

Entretanto, se por um lado é vantajoso ser genérica para poder se adaptar a qualquer subdomínio, podem existir desvantagens em relação a outros fatores e isso deve ser analisado e considerado.

Da perspectiva de expressividade semântica, “uma medida ter dados” não significa muito para o subdomínio. Para resolver essa questão, a ontologia MOMENT previu diversos casos de subdomínios relacionados a medidas e monitoramento de rede e utilizou amplamente taxonomias para descrever subclasses mais semanticamente adequadas de um determinado subdomínio, inclusive o domínio do PingER. A Figura 11 apresenta uma proposta de como o subdomínio do PingER poderia ser modelado utilizando a ontologia MOMENT original, i.e., sem realizar nenhuma alteração ou adaptação.

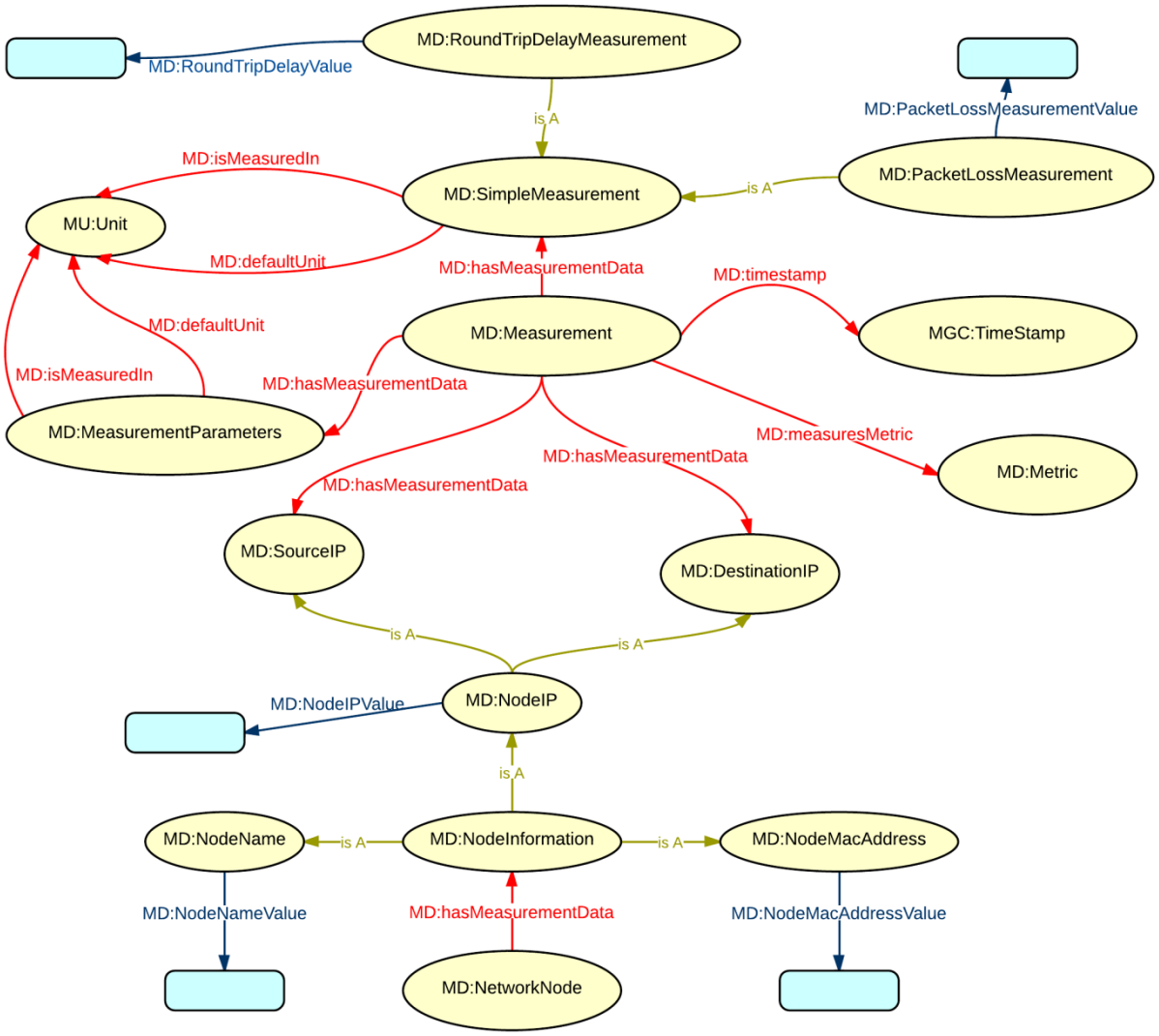


Figura 11 – Proposta da ontologia MOMENT (EUROPEAN TELECOMMUNICATIONS SI, 2012) para o domínio do PingER

Na representação do modelo da Figura 11, no centro fica a superclasse principal, que é a medida `Measurement`. Uma instância de `Measurement` tem dados da medida (`hasMeasurementData`) instâncias de subclasses de `MeasurementData`. Uma medida do PingER é em relação a nós fontes e nós destinos. A ontologia provê as subclasses `SourceIP` e `DestinationIP`, que são subclasses de `NodeIP`, que é subclasse da classe mais genérica que descreve outras informações (instâncias de subclasses de `NodeInformation`, como `NodeName` e `NodeMacAddress`) sobre um nó de rede (instâncias de `NetworkNode`), o qual se relaciona com `NodeInformation` e suas subclasses através novamente da propriedade `hasMeasurementData`.

Além das informações sobre os nós envolvidos, na medida se considera também quando ocorreu e o que foi medido. Na ontologia, `Measurement` se relaciona a `TimeStamp` (através da propriedade `timestamp`) para prover informação temporal e se relaciona a `Metric` (através da relação `measuresMetric`) para especificar qual métrica de rede está sendo medida. Uma métrica pode ainda ter uma unidade padrão na qual as medidas foram feitas.

Prosseguindo, uma instância da classe `Measurement` se relaciona (através, mais uma vez, da propriedade `hasMeasurementData`) a uma instância de uma das subclasses de `SimpleMeasurement`. Essas subclasses são referentes a uma métrica de medida de rede, as quais são amplamente utilizadas no domínio do PingER.

Em relação às unidades, a métrica utilizada na medida pode ter uma unidade padrão (uma instância de `SimpleMeasurement` se relacionando a uma instância de `Unit`) ou uma medida pode também ser feita em uma unidade diferente da padrão. A ontologia MOMENT é poderosa o suficiente para apoiar automaticamente conversão entre unidades.

Além das unidades, uma medida pode também ter outros parâmetros extras ainda não mencionados. Por exemplo, foi visto na seção 4.1.2 que PingER especifica tamanhos de pacote 100 e 1000 bytes. Apesar deste projeto considerar apenas tamanhos de 100 bytes, é necessário permitir flexibilizar a escolha de parâmetros diferentes. Por isso, uma instância do tipo `Measurement` pode se relacionar a `MeasurementParameters`, de novo por `hasMeasurementData`.

Finalmente, o valor observado pelo cruzamento dos parâmetros necessários é um atributo (data property) de uma das subclasses de `SimpleMeasurement`. Exemplificando, uma instância da classe `Measurement` tem como dados (`hasMeasurementData`) instância de uma subclasse de `SimpleMeasurement`, como a `PacketLossMeasurement`, que contém o valor observado da medida; no exemplo, um `PacketLossMeasurementValue`, seguindo a estrutura

fundamental da ontologia diagramada na Figura 9. Note também que é possível que um valor observado pode ser eventualmente medido por uma unidade diferente da padrão.

Seria o ideal reutilizar a ontologia MOMENT aplicada ao domínio do PingER já que ela é capaz de descrever o domínio e desfrutaríamos e ressaltaríamos todas as vantagens de reúso mencionadas na seção 3.2.

Entretanto, é necessário julgar se ela deve ser realmente utilizada no projeto e os critérios de avaliação foram sob perspectivas mencionadas anteriormente: expressividade semântica, completude e desempenho das consultas.

a) Expressividade semântica

Pelas diversas subclasses das superclasses principais (`Measurement` e `MeasurementData`), percebe-se que MOMENT considerou uma quantidade realmente significativa de subdomínios relacionados a medidas e monitoramento de rede. As subclasses de `MeasurementData`, em sua grande maioria, possuem um data property especializado para especificamente descrever o valor do dado sendo medido. Entretanto, apesar das especializações das classes e data properties, a ontologia falhou em minuciar as object properties. Não existem relações mais semanticamente especializadas entre classes. Como discutido, a object property `hasMeasurementData` é genérica demais, não expressa muito e é exaustivamente utilizada diversas vezes na ontologia. Uma sugestão de melhoria seria, por exemplo, se uma instância de `NetworkNode` se relacionasse com uma instância de `NodeIP` através de uma possível propriedade mais específica e próxima do domínio nomeada `hasNodeIP`. É intuitivo perceber que a possível relação `hasNodeIP` quer dizer muito mais em relação ao domínio do que `hasMeasurementData`.

Ademais, `NetworkNode` possui a propriedade `hasMeasurementData` que, como visto no diagrama da estrutura fundamental da ontologia MOMENT (Figura 9), é uma propriedade que liga `Measurement` a `MeasurementData`. De fato, na taxonomia da ontologia, `NetworkNode` é uma subclasse de `Measurement`. Do senso comum e mesmo do dicionário[58], uma "medida" é um ato ou processo de medir ou, ainda, uma quantidade obtida através desse processo. No domínio do nosso interesse (o PingER), isso é ainda mais claro: uma medida é um valor observado obtido através do cruzamento de parâmetros, inclusive nós de rede. Conclui-se então que, mesmo genericamente, declarar que um nó de rede (um dispositivo no qual *links* de rede se interceptam, tipicamente um roteador, switch ou end point – vide Apêndice A) é uma medida traz um desvio de semântica.

[58] http://www.merriam-webster.com/dictionary/measurement

Adicionalmente, levou-se em consideração utilizar a subclasse de `Measurement` nomeada de `Ping` porque parecia ser semanticamente mais adequada. Porém, ela é uma classe anônima definida por uma lista de restrições (reveja seção 2.2.4) que uma medida do tipo `Ping` deve satisfazer (ter um IP fonte, ter um IP destino, ter um tamanho de pacote, etc) e algumas restrições não eram contempladas no domínio do PingER. Por isso, uma medida do PingER não poderia ser classificada como uma instância da classe `Ping` e logo optou-se utilizar mesmo a superclasse mais abstrata `Measurement`.

Finalmente, é notável que a modelagem utilizando a ontologia MOMENT original ficou muito mais complexa do que ela talvez precisaria ser e menos expressiva do que poderia ser. Isso aconteceu devido à característica generalista da ontologia.

b) Completude

Comparando à modelagem apresentada na Figura 11 com o primeiro rascunho do modelo conceitual apresentado na Figura 8, percebe-se que de fato a ontologia MOMENT é poderosa o suficiente para contemplar mais um subdomínio de medidas e monitoramento de rede; no caso, o domínio do PingER. Os conceitos fundamentais específicos do PingER podem, de fato, ser representados na ontologia. Todavia, ao observar a análise do domínio feita na seção 4.1, percebe-se uma necessidade de representar dados geográficos. Afinal, um nó de rede está localizado em algum lugar físico, isto é, que minimamente tem uma latitude e longitude no planeta. A ontologia MOMENT original previu isso e definiu a classe `PhysicalLocation` que possui atributos específicos relacionados à localização do nó no mundo. Contudo, os arquivos OWL[59] não definem uma relação (object property) que permita ligar um `Measurement` a uma `PhysicalLocation`, nem mesmo indiretamente, caracterizando uma considerável incompletude para ser utilizada no projeto do PingER. Ou seja, mesmo se a ontologia MOMENT fosse utilizada, ela necessariamente precisaria ser adaptada no mínimo nesse ponto. Embora, nesse caso, isso não é um grande problema já que é possível resolver facilmente. Tirando esse contratempo pontual, a ontologia foi capaz de modelar todo o subdomínio do PingER.

c) Desempenho nas consultas

Como visto na seção 3.2, além da semântica do modelo, é essencial observar a otimização do desempenho e como o computador vai processar o modelo. Na seção 3.3, foi mencionado que a quantidade de triplas é um fator relevante para medir desempenho nas

[59] https://svn.fp7-moment.eu/svn/moment/public/Ontology/

tecnologias de Web Semântica, isto é, deve-se tentar minimizar, quando possível, a quantidade de triplas a serem utilizadas. Já pelas conclusões do item (a), percebe-se que instanciar dados em RDF utilizando a ontologia MOMENT possivelmente levaria a um grande número de triplas, possivelmente bem mais do que o desejado.

No item (a) foi discutido que instâncias de subclasses de `MeasurementData` possuem um atributo que definem seu valor. Instâncias de `Measurement` não possuem o valor medido (o dado resultado do cruzamento dos parâmetros) diretamente relacionado à instância, mas sim relacionado à instância de uma subclasse de `MeasurementData`. Em triplas:

```
(measurement1, hasMeasurementData, packetlossmeasurement1) .
(packetlossmeasurement1, PacketLossMeasurementValue, valor numerico)
```

Onde `measurement1` e `packetlossmeasurement1` são instâncias. Em números, para cada medida no PingER, serão necessárias 2 triplas somente para descrever o valor. Relembrando, da seção 4.1, que cada medida do PingER considerada no projeto se dá minimamente em relação a um par de nó monitor e monitorado (cerca de 8200 pares), que ocorreram nos últimos 60 dias e que mediram uma determinada métrica (11 são consideradas). Fazendo uma conta rápida, chegamos facilmente a um número superior a 5 milhões de medidas. Se forem necessárias 2 triplas para descrever cada valor observado na medida, teremos no mínimo 10 milhões de triplas. Uma sugestão de melhoria seria ligar a data property do valor observado diretamente à instância da medida. Todavia, isso rompe toda a estrutura fundamental da ontologia MOMENT vista na Figura 9 porque o valor observado na medida deve ficar em `MeasurementData` e não em `Measurement`. Isso é um problema considerado grave para o projeto.

Outro caso análogo são as subclasses de `NodeInformation`. A representação da ontologia MOMENT (Figura 11) mostra que uma instância de `NetworkNode` se liga a instâncias de subclasses de `NodeInformation`, as quais se ligam ao seu data property para descrever seu valor. Em triplas:

```
(networknode1, hasMeasurementData, nodeip1)
(nodeip1, hasNodeIPValue, a string que contém a informação do IP do
nó)
```

Seria intuitivamente melhor (no mínimo em termos de menor quantidade de triplas) se uma instância de `NetworkNode` se ligasse diretamente a uma informação específica através de uma relação (data property) específica, por exemplo:

```
(networknode1, hasNodeIP, a string que contém a informação do IP do
nó)
```

Ainda nesse caso do `NodeInformation`, existe um outro problema ainda pior, não diretamente relacionado ao número de triplas. A consulta SPARQL para recuperar a informação do nome do nó (`NodeName`) de uma instância de `NetworkNode` que seja fonte de uma medida (operação bem comum para o domínio) fica muito mais complexa – não só para entender e escrevê-la, mas também para o processador de consultas executá-la – devido às restrições impostas pela modelagem original. Visualmente, na Figura 11, vindo a partir do `Measurement` e descendo no grafo, é possível acessar o IP do nó fonte. Daí, para recuperar outra informação do nó (ex., `NodeName`) a partir do IP dele, é preciso ainda navegar no grafo para depois acessar o `NetworkNode` que contém o `NodeIP` igual ao `SourceIP` relacionado à medida. Esse trecho da consulta SPARQL ficaria (abstraindo os prefixos) desta forma:

```
?Measurement :hasMeasurementData ?SourceIP .
?SourceIP :SourceIPValue ?SourceIPValue . #string do IP do nó fonte
?NetworkNode :hasMeasurementData ?NodeIP .
?NodeIP :NodeIPValue ?SourceIPValue . #filtra pelo IP do nó fonte
?NodeIP :hasMeasurementData ?NodeName .
?NodeName :NodeNameValue ?NodeName .
```

Por esse motivo também, seria muito mais simples se um `NetworkNode` se ligasse diretamente às suas informações. Contudo, isso também infringiria a estrutura fundamental da ontologia já que, como visto anteriormente, `NetworkNode` é definido como uma subclasse de `Measurement` e logo não pode ter dados da medida (`NodeInformation` é subclasse de `MeasurementData`) diretamente associados.

Novamente, o problema do desempenho ocorreu devido à característica generalista da ontologia e, especialmente nesse caso, devido à restrição fundamental de que um `Measurement` não pode ter dados de medida diretamente associados. Como a principal operação do projeto será consultas SPARQL sobre a base, é essencial tentar otimizar o desempenho das consultas e isso deve ser levado em consideração ao construir a ontologia para o domínio do PingER.

Portanto, apesar da discussão sobre o aspecto de desempenho versus expressividade semântica na seção 3.2, conclui-se que a ontologia MOMENT tem desvios e até perda de semântica, além de apresentar problemas relacionados a desempenho de consultas. Nota-se que excessiva generalidade interferiu negativamente na semântica e desempenho. Entretanto, ela é extremamente poderosa (no sentido de ser genérica) para descrever subdomínios do domínio de medidas de rede, definindo conceitos fundamentais para o domínio, e provendo boa base para uma proposta de melhoria de modelagem. Por essas razões optou-se por não utilizar ontologia original, mas basear-se em alguns de seus conceitos para propor outra

ontologia aplicada e especializada ao domínio do PingER e que observa atentamente os problemas levantados.

4.2.2 Ontologias de conceitos gerais (tempo e espaço)

Conceitos gerais de tempo (quando a medida foi feita) e espaço (onde os nós de rede se localizam) são essenciais na modelagem do PingER.

Para descrever tempo na ontologia MOMENT, Salvador *et al.* (2010), afirmam utilizar a ontologia Time do W3C (WORLD WIDE WEB CONSORTIUM , 2006) a qual define a classe `DateTimeDescription` para minuciar intervalos de tempo. Para os dados selecionados para o projeto (seção 4.1.2), apenas as propriedades `day`, `month`, `year`, `dayOfYear`, `dayOfWeek` foram necessárias para cobrir todos os intervalos de tempo. Para especificar a unidade temporal sendo medida no intervalo, utilizou-se a propriedade `unitType`. Por exemplo, o intervalo de tempo definido pelos instantes iniciais e finais do dia "15 de novembro de 2013" possui a unidade temporal `day`.

Para utilizar as propriedades definidas pela classe `DateTimeDescription` da ontologia Time do W3C, criamos uma nova classe nomeada `DateTime` e a definimos como subclasse da `DateTimeDescription`. Para tornar as consultas SPARQL mais eficientes para filtrar os intervalos, utilizamos adicionalmente as data properties `startTime` e `endTime` que definem os intervalos de tempo. Por conveniência das apresentações, definimos também a propriedade `displayValue` para armazenar *aliases* com nomes significativos para humanos e fáceis de serem recuperados programaticamente. Os padrões `DDMmmAAAA`, `MmmAAAA` e `AAAA` para dias, meses e anos respectivamente (ex: `12Jan2013`, `Jan2013`, `2013`) foram adotados por serem utilizados pelo PingER.

Em relação à descrição espacial, a ontologia MOMENT provia a classe `PhysicalLocation` que define data properties (cujo alcance são *strings*) que caracterizam onde o nó da rede se localiza fisicamente no planeta. Essa abordagem é boa para recuperar rapidamente e eficientemente o nome do país ou da cidade ou até mesmo as coordenadas geográficas do nó de rede. E isso de fato é o suficiente para resolver a maioria das consultas que ocorrem no domínio do PingER. Entretanto, ela está longe do ideal do ponto de vista da interoperabilidade de dados abertos ligados entre outras bases RDF que contenham dados geográficos.

Imagine uma situação em que se deseja saber alguma informação extra sobre o país onde o nó se localiza e essa informação não é contemplada no domínio. Se os dados

estivessem adequadamente ligados de alguma forma, seria possível navegar na nuvem de LOD e recuperar a informação desejada. Para resolver isso, sugerimos utilizar object properties, ao invés de data properties, para descrever a localização física de um nó da rede. Se cada elemento da descrição do nó da rede for um recurso, ele poderá ter *links* para outras bases de dados em RDF e assim utilizar técnicas de *mashup* e recuperar outras informações que poderão ricamente acrescentar à base de conhecimento do PingER. Além disso, se outras pessoas quiserem ligar seus dados com os dados em LOD do PingER, uma forma comum de entrada poderia ser através dos *links* entre dados geográficos. Ou seja, se a ontologia do PingER não previr esses *links*, estaríamos fechando uma porta intuitiva, comum e muita usada para interoperabilidade entre bases RDF.

Um exemplo em triplas que ligaria o recurso Brasil definido pelo PingER ao recurso Brasil da DBPedia poderia ser declarado como:

```
(PingER:networknode1, isInPhysicalLocation, PingER:Brazil)
(PingER:Brazil, owl:sameAs, DBPedia:Brazil)
```

Como base para a parte geográfica da ontologia do PingER, utilizamos a ontologia do Geonames (2013) de modo a tirar vantagens do reúso através dos dados e conceitos proporcionados por um dos ícones mais importantes em LOD.

Devido às discussões anteriores, criamos uma nova `PhysicalLocation`. Ela tem as mesmas ideias daquela proposta pela MOMENT, mas é relacionada ao Geonames. Em seguida, declaramos as classes `Town`, `State`, `Country`, `Continent` e `School` que sãs as entidades geográficas utilizadas no domínio do PingER. Na ontologia do Geonames, todas essas entidades geográficas e muitas outras (a saber, montanhas, vales, lagos, praias, hotéis, aeroportos e etc.) são instâncias da classe `Feature`[60]. Por isso, para ligar a essa ontologia, definimos que a nossa `PhysicalLocation` é subclasse da `Feature` do Geonames. Nota-se também que a `Feature` utiliza os vocabulários da `SpatialThing`, definidos pela WGS84_POS[61] do W3C, para descrever dados de latitude e longitude de uma "coisa espacial".

A ontologia do Geonames define códigos (instâncias da classe `Code`) para especializar instâncias da `Feature` em subentidades geográficas específicas tais como as contempladas no domínio do PingER (cidade, estado, país, continente e universidades). O Quadro 3 contém os códigos do Geonames que serão considerados no PingER LOD:

[60] http://www.geonames.org/export/codes.html
[61] http://www.w3.org/2003/01/geo/

Quadro 3 – Códigos de *features* (entidades) do Geonames[62]

Feature	Code	Descrição no Geonames
Cidade	P.PPL	Lugar povoado (*Populated place*) – cidade ou outra aglomeração de construções onde pessoas moram e trabalham.
Estado	A.ADM1	Divisão administrativa de primeira ordem, como um estado no Brasil ou nos Estados Unidos.
País	A.PCLI	Entidade política independente.
Continente	L.CONT	África, América do Norte, América do Sul, Antártica, Ásia, Europa, Oceania.
Universidade	S.SCH	Construção(ões) onde uma ou mais áreas do conhecimento são ensinadas.

Pela ontologia, para um indivíduo qualquer ser, por exemplo, um País, é necessário que ele seja membro da classe `Feature` e equivalente a uma classe anônima (reveja seção 2.2.4) definida pela restrição de ter object property que aponte para uma instância que tenha o código (`featureCode`) A.PCLI.

A Figura 12 mostra a organização das classes de localização física do domínio do PingER conforme a ontologia do Geonames e como subclasses da `Feature` .

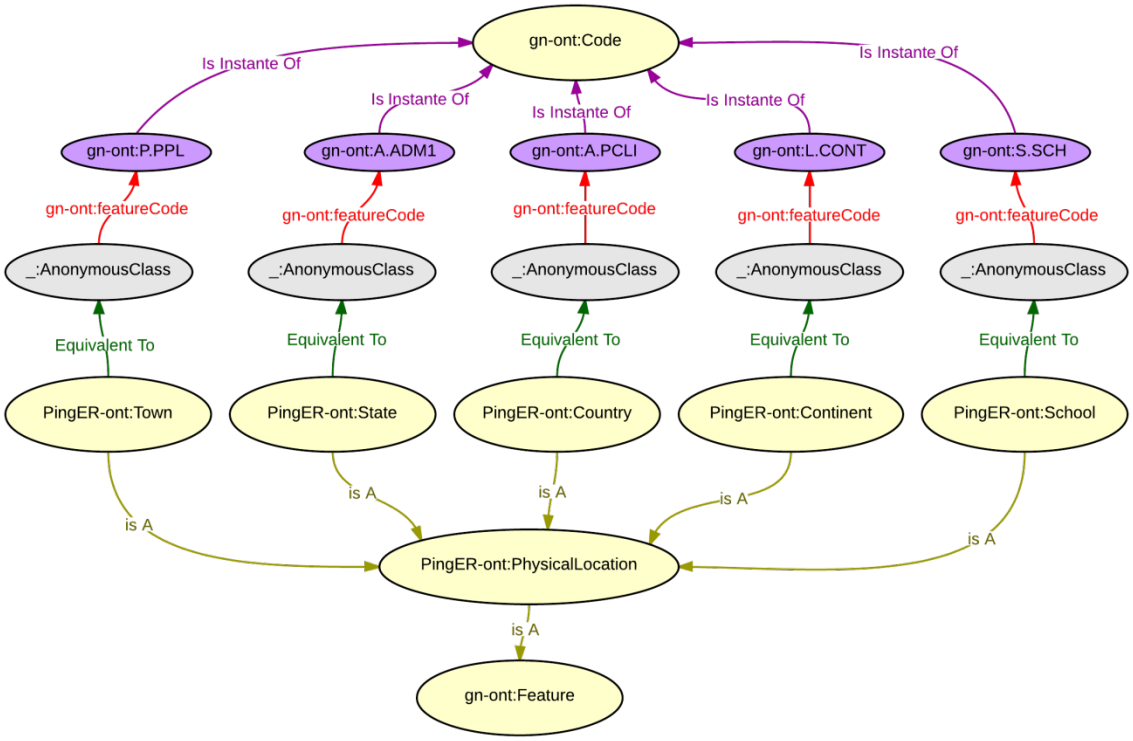


Figura 12 – Organização das classes de localização física do domínio do PingER na ontologia do Geonames (2013)

[62] http://www.geonames.org/export/codes.html

Adicionalmente, devido à natureza dos dados do PingER, é preciso existir uma maneira de organizar hierarquicamente onde um nó se localiza fisicamente de modo a dar suporte à agregação nas consultas, como introduzido na subseção 4.1.2. A classe `Feature` da ontologia do Geonames possui a object property `parentFeature` que pode ser usada para indicar a entidade geográfica (instância de `Feature`) onde uma outra entidade está localizada e isso pode ser usado para a hierarquia geográfica. Ademais, a object property `parentFeature` se especializa em `parentADM1` e `parentCountry` (significam "está no estado" e "está no país", respectivamente) que trazem semântica e estrutura mais específicas que auxiliarão inclusive no desempenho das consultas. Prosseguindo nessa linha, para cobrirmos todos os casos do domínio do PingER, basta analogamente adicionarmos as object properties `parentTown` e `parentContinent` à ontologia.

Como cada localização física, concordando com a estrutura ontológica do Geonames (herdada da classe `Feature`), possui as object properties `parentTown`, `parentADM1`, `parentCountry` e `parentContinent`, a ontologia proposta contempla algumas redundâncias. Por exemplo, as seguintes triplas são explicitamente declaradas:

```
(estado do Rio de Janeiro, parentCountry, Brasil) .
(cidade do Rio de Janeiro, parentADM1, estado do Rio de Janeiro) .
(cidade do Rio de Janeiro, parentCountry, Brasil) .
```

A última tripla poderia ser omitida uma vez que, no grafo de triplas, é possível construir o caminho `(cidade do Rio de Janeiro, parentADM1, estado do Rio de Janeiro, parentCountry, Brasil)`, com o qual teríamos indiretamente a informação de que a cidade do Rio de Janeiro está no país Brasil. Somente nessa análise superficial, se não existisse a tripla extra, já se observa a redução de redundância, logo a diminuição do número de triplas (pelo menos 1 tripla a menos para cada cidade do mundo). Como são muitas cidades no mundo, representaria um ganho de espaço razoável.

Entretanto, manter a redundância traz vantagens no desempenho de consultas. Já que os dados estão diretamente ligados ao recurso, não é necessário caminhar no grafo de triplas para obter alguma informação extra. As consultas sobre o modelo ficam menores (mais fáceis de entender e do processador de consultas executar), uma vez que é possível verificar diretamente, por exemplo, se uma cidade pertence ou não a um país, ao invés de antes passar pelo estado.

Portanto, apesar de parecer um problema, a proposta da nossa ontologia, assim como o Geonames, se vale das vantagens proporcionadas pela redundância dos dados. O ônus trazido por não minimizar o número de triplas é muito inferior ao ganho com desempenho. Além

disso, as redundâncias são controladas, baseando-se nos dados já limpos do PingER e nos dados do Geonames, que apresenta um razoável grau de confiança de acordo com a comunidade de dados ligados (3KBO, 2013). Em relação a uma possível perda de desempenho devido a essa redundância, em especial no caso do PingER, o número de medições (isto é, o cruzamento entre parâmetros para se obter um valor sobre a qualidade da rede) é gigantesco, da ordem de dezenas de milhões, logo a quantidade de triplas relacionadas a uma medição devem ser necessariamente minimizada. Todavia, as cidades monitoradas no PingER são da ordem de milhares (menos de duas mil cidades, mais precisamente). Em outras palavras, essa redundância não é crítica. Experimentalmente, pelo menos no caso das consultas para o projeto PingER LOD, para o processador de consultas, de um lado, processar mil triplas a mais ou 10 mil a mais não representa diferença significativa. Por outro lado, processar 10 milhões de triplas a mais ou 20 milhões a mais, afeta consideravelmente o desempenho das consultas. Ainda, não só as consultas, mas o tempo de carga dos dados é muito maior.

Além da relação hierárquica entre as entidades geográficas, é preciso relacionar o nó de rede a sua localização física. Isso será realizado através da object property `isInPhysicalLocation` que liga uma instância de `NetworkNode` a uma instância de `PhysicalLocation` ou uma de suas subclasses. Para convenientemente facilitar as consultas e trazer mais semântica, a relação `isInPhysicalLocation` pode ser ainda especializada em `isInTown`, `isInState`, `isInCountry`, `isInContinent` e `isInSchool` – caso o nó esteja em uma universidade – cujo domínio são instâncias de `NetworkNode` e o alcance são instâncias de `PhysicalLocation` ou suas subclasses.

Essa modelagem também permite sobrecarregar redundâncias de localizações físicas. Não é necessário declarar que um nó de rede está em uma cidade, em um estado, em um país e em um continente. Bastaria afirmar que o nó está na cidade e, caminhando no grafo, as outras informações de nível hierárquico superior poderiam ser obtidas indiretamente. Entretanto, como foi discutido nos parágrafos anteriores, declarar explicitamente em todos os nós de rede toda a descrição taxonômica da sua localidade física significou ganho em desempenho, permitindo agregações geográficas (isto é, agregar as medições por cidade, estado, país, etc.) mais eficientes. Além disso, o número de nós monitorados pelo PingER é menos de mil, o que, como discutido, não é considerado grande comparado ao número de medições. Ou seja, essa redundância também não é crítica.

Adicionalmente, a Figura 13 mostra que uma universidade (ou qualquer instituição acadêmica monitorada pelo PingER) também tem as relações `parentTown`, `parentADM1`,

`parentCountry` e `parentContinent` para manter a organização hierárquica e possibilitar agregações futuras.

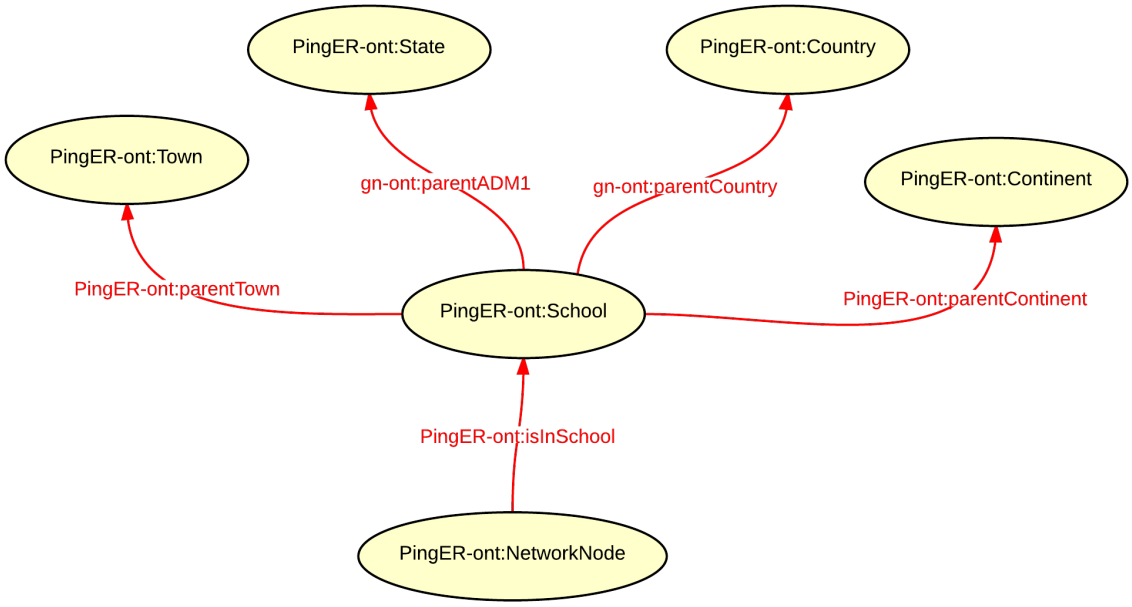


Figura 13 – Recorte da parte da ontologia que modela as relações hierárquicas da geografia de universidades.

Todas essas partes cobrem o subdomínio geográfico do domínio do PingER. A Figura 14 apresenta a parte da ontologia do PingER que esquematiza as relações de localidade dos nós de rede, baseada na ontologia do Geonames.

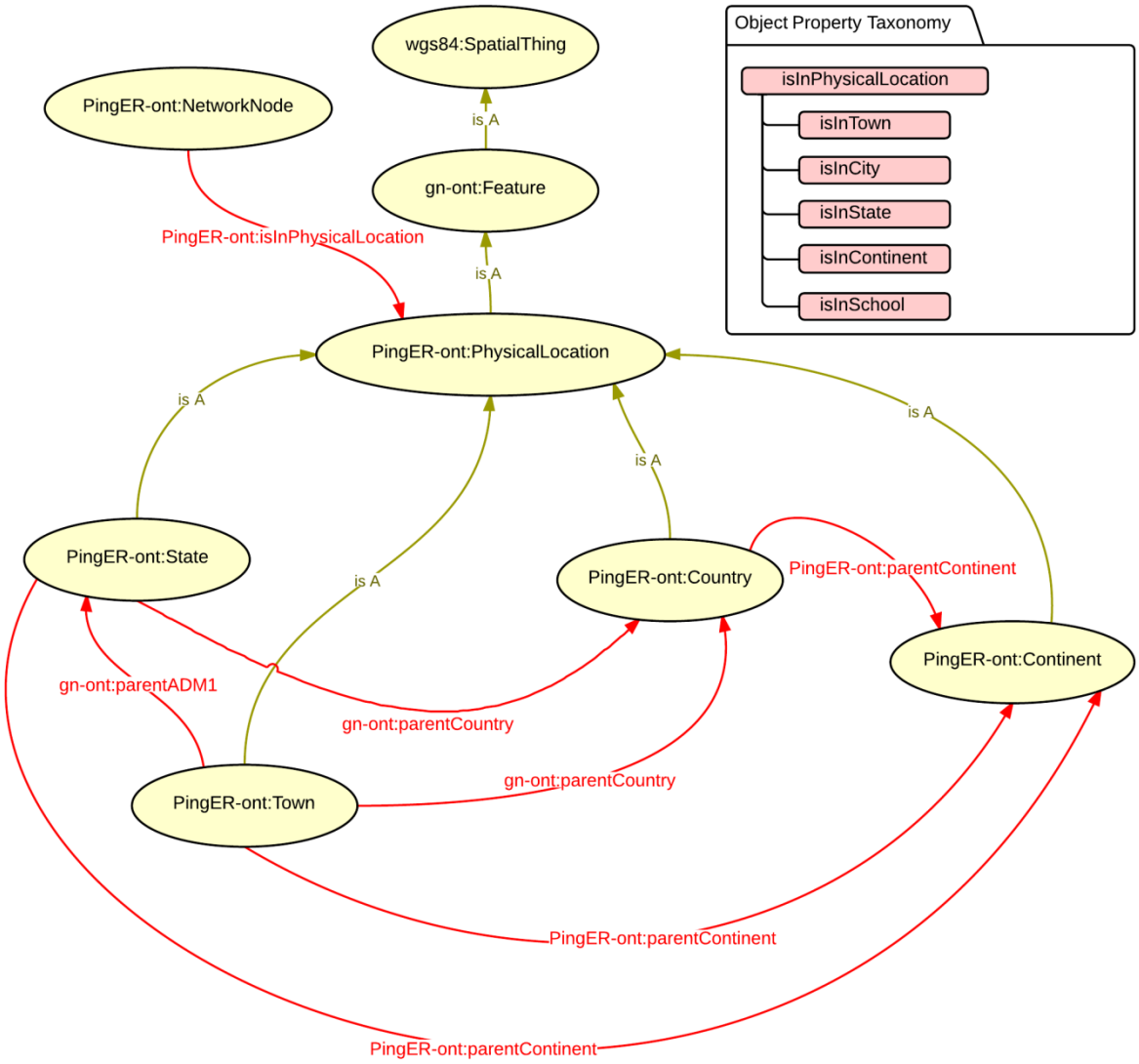


Figura 14 – Parte geográfica da ontologia do PingER

4.2.3 Ontologia proposta para o PingER

Finalmente, unindo todas as discussões levantadas, sugerimos uma ontologia, representada na Figura 15, baseando-se nos conceitos de OWL aplicada para o domínio do PingER que propõe resolver o modelo conceitual apresentado na seção 4.1.3.

Esta ontologia reutiliza e se baseia nos conceitos introduzidos pela ontologia MOMENT, adaptando-os às necessidades do domínio e do projeto. Também utiliza as ontologias do Geonames e a Time, do W3C.

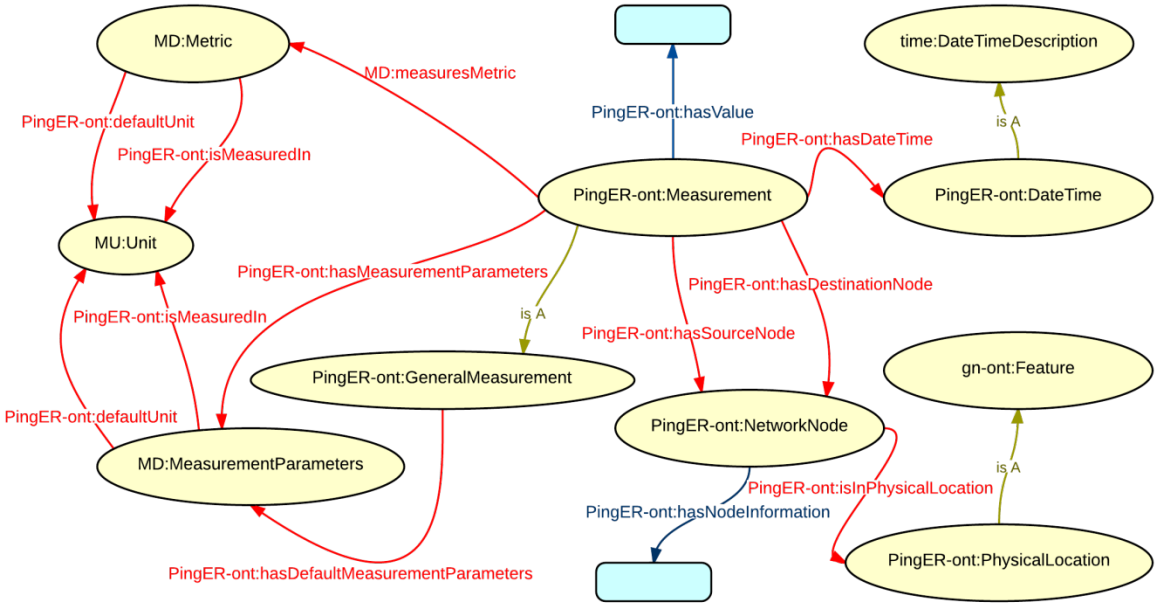


Figura 15 – A ontologia do domínio do PingER

Ao comparar com a modelagem proposta utilizando a ontologia MOMENT (Figura 11) original, sem adaptações, observa-se que ela é mais específica logo mais semântica para o domínio do PingER, tendendo a otimizar o desempenho das consultas e minimizar o número de triplas. Principalmente, ela dá suporte para resolver de forma eficiente todos os requisitos do negócio para o projeto, inclusive utilizando todos os conceitos de publicação de LOD, discutidos em seções anteriores.

Note que as únicas classes da MOMENT totalmente reutilizadas, isto é, sem nenhuma adaptação, foram as `Metric` e `Unit`. Elas já são boas o suficiente para resolver as necessidades do domínio. Entretanto, as object properties `defaultUnit` e `isMeasuredIn` precisaram ser modificadas para que seu domínio inclua a classe `Metric`. Dessa forma, as instâncias de `Metric` teriam uma unidade padrão (que é o caso do PingER) ou, em uma abordagem mais flexível, poderiam ter uma outra unidade diferente. A conversão das unidades pode ser dada por inferência de acordo com a ontologia MOMENT. Por exemplo, uma instância da `Metric` poderia ser *throughput* a qual possui uma unidade padrão (kbit/s para o PingER) que é uma instância da classe `Unit`.

Além de `Metric`, as propriedades `defaultUnit` e `isMeasuredIn` têm domínio também a `MeasurementData`, da ontologia MOMENT, bem como suas subclasses, como a `MeasurementParameters`, que pode ser ainda especializada em `PacketSize`. Tamanho de

pacote é um parâmetro comum no domínio do PingER e é preciso permitir que as medidas tenham parâmetros diferentes, além do padrão.

Entretanto, vimos da seção 4.1.2 que PingER LOD considera apenas tamanho de pacote 100 bytes e, por isso, é preciso modelar especialmente uma maneira de representar valores padrões sem interferir muito no desempenho. Se formos obrigados a especificar em cada medida que ela tem, por exemplo, o tamanho de pacote 100 bytes, estaremos incluindo informação desnecessária já que é sabido que todas as medidas consideradas têm esse valor (a não ser que seja especialmente declarado algo diferente). Em outras palavras, a ontologia obrigará especificar 1 tripla a mais para cada medida e já discutimos (4.2.1.c) que isso não é eficiente. A maneira proposta para resolver isso é definir, na ontologia, que toda medida (instância de `Measurement`) tem parâmetros padrões (`hasDefaultMeasurementParameters`), inclusive, por exemplo, tamanho de pacote padrão. Para resolver isso utilizando os mecanismos disponíveis em OWL, declaramos que a classe `Measurement` é uma especialização da classe mais genérica `GeneralMeasurement` a qual tem parâmetro padrão (instância da classe `MeasurementParameters`). Isso significa que qualquer medição – apesar de poder sobrescrever informação sobre o valor do parâmetro padrão de medida com valores diferentes do padrão – possui valor padrão de parâmetros de medida bem definidos já previstos no modelo ontológico. Dessa maneira, agregamos valor semântico à classe `Measurement` e não a cada instância de `Measurement` que, como já discutimos, são da ordem de milhões. Logo, não será obrigatório repetir essa informação a cada instância, reduzindo drasticamente o número de triplas necessário.

Observe, também, que para tornar as relações `hasMeasurementParameters` e `hasDefaultMeasurementParameters` mais eficientes em semântica e desempenho, podemos especializá-las em `hasPacketSize` e `hasDefaultPacketSize`. Por enquanto somente essa subpropriedade é necessária para o domínio do PingER, mas isso é facilmente extensível a outros parâmetros não previstos. E ainda, `hasNodeInformation` poderá ser especializado em data properties mais específicas em relação a informações de um nó de rede para o PingER. A Figura 16 mostra a organização taxonômica desses conceitos.

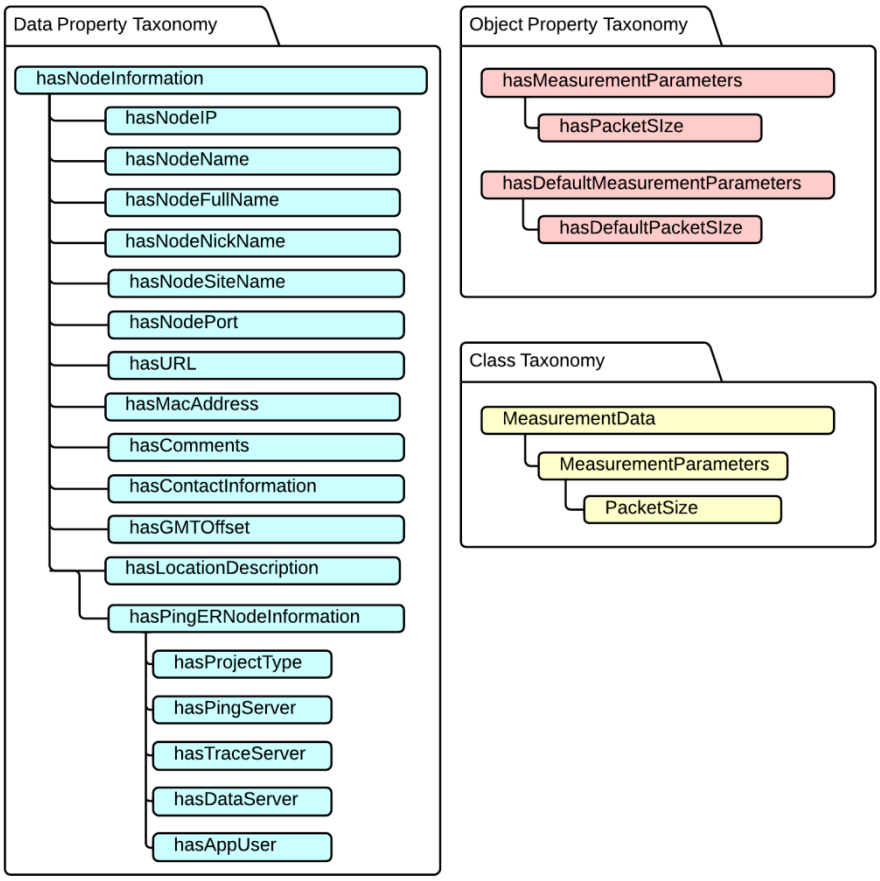


Figura 16 – Taxonomia auxiliar da ontologia do PingER

4.3 O Projeto de Triplificação do PingER Linked Open Data

4.3.1 A escolha do SGBD RDF

Vimos que a escolha do repositório de triplas é muito importante no projeto, pois é um dos fatores que mais influencia no desempenho do tempo de execução das consultas. Foi mencionado na seção 3.3.1 que planejar, estimar o número de triplas a ser carregado no banco e, principalmente, pesquisar as alternativas de SGBD RDF existentes é muito importante e poupa tempo no futuro. Essas recomendações surgiram porque isso não foi observado atentamente durante a fase de escolha do SGBD e, por isso, muitas alterações foram necessárias, gastando-se muito tempo. O processo de escolha do SGBD do projeto PingER LOD deu-se desta maneira:

a) Jena SDB

Inicialmente, foi escolhido o Jena SDB, com MySQL. Esse repositório provê mecanismos de armazenamento e consulta utilizando SGBDs relacionais tradicionais (como MySQL e PostgreSQL).

Após estabelecer o repositório de triplas Jena SDB 1.3.6 com uma biblioteca nativa de Java para armazenar e consultar uma base relacional em MySQL, foi possível carregar uma quantidade razoavelmente grande de triplas para testar.

Especificamente, dados de todos os nós e de medições anuais foram carregados. Entretanto, o desempenho das consultas dos dados não correspondeu às expectativas. Algumas consultas simples como "listar o valor de uma dada métrica em um dado ano para um dado par nó monitor e nó monitorado" demorou cerca de 7 minutos para executar. Existiam apenas pouco mais de 100K triplas na base.

Foi feita, então, uma primeira estimativa, ainda não tão cuidadosa, de quantas triplas seriam carregadas no banco e observou-se que o número de triplas seria maior que 1M.

Por isso, decidiu-se gastar mais tempo pesquisando sobre comparações entre SGBDs RDF e dois artigos foram utilizados para auxiliar e embasar a escolha do repositório.

b) Sesame Native

Utilizando os resultados de Bizer e Schultz (2010) e de Bio Ontology (2010), decidiu-se tentar utilizar o Open RDF Sesame Native 2.7.2. Esta solução requer um Java Webserver para hospedar o repositório. Foi utilizada a última versão do Tomcat[63], a versão 7. A implementação da camada física do repositório disponibiliza a possibilidade de configuração de índices, os quais são ditos decisivos para otimizar o desempenho de execução de consultas (OPEN RDF, 2013).

Foram escolhidos os índices `spoc,sopc,psoc,posc,opsc,ospc`. A ordem dos índices para (`s`)ujeito, (`p`)redicado e (`o`)bjeto define o padrão de uma busca. Por exemplo, o índice `spoc` otimiza a busca na qual o sujeito é o primeiro campo da tripla. Mais especificamente, uma consulta que levaria vantagem sobre este índice seria: liste todas as instâncias das `métricas` (i.e., o sujeito a ser consultado) que têm o `tipo TCP Throughput`. `Tipo` (o predicado) e `TCP Throughput` (o objeto) são dados e a consulta está procurando a `métrica` (o sujeito).

O (`c`)ontexto é geralmente utilizado como um quarto campo da "tripla" (mais precisamente uma "quad" agora) para especificar grafos nomeados (*named graphs*) no Sesame. A solução deste projeto não utiliza diferentes *named graphs,* não considerando o quarto campo das quads.

Entretanto, muitos índices aumentam o espaço em disco utilizado (para armazená-los) e também aumentam o tempo para sua carga e manutenção. Mesmo assim, foi concordado

[63] http://tomcat.apache.org/

que menor tempo de execução de consultas teria mais prioridade que tempo de carga dos dados no repositório e espaço em disco utilizado.

Ainda enquanto o repositório estava sendo testo, já havia quase 13M de triplas carregadas e Bio Ontology (2010) apontou que o maior banco já testado no Sesame Native tinha 50M de triplas. Foi feita uma nova estimativa e verificou-se que era possível ultrapassar esse número com os dados do PingER. Porém, essa alternativa continuou sendo utilizada porque o desempenho do Sesame Native era surpreendentemente superior ao do Jena SDB: exatamente a mesma consulta e na mesma máquina que demorava 7 minutos passou a demorar menos de 5 segundos. Nesta fase do projeto, acreditava-se que esse desempenho já era bom o suficiente. Logo, toda parte de carregar e consultar a base foi migrada para o Sesame Native.

c) Sesame OWLIM

Com o passar do tempo, o número de triplas aproximou-se de 50M de triplas. As consultas mais simples continuavam muito rápidas, mas algumas mais complexas (e estas geralmente eram as mais interessantes pois incluíam *mashups*) algumas vezes demoravam quase 2 minutos, o que era um pouco frustrante. Já era possível ter resultados bem interessantes e gerar relatórios antes não disponíveis; porém, o tempo prolongado das consultas era um fator negativo. De fato, o Sesame Native, como os artigos previam, funciona muito bem para um número de triplas razoavelmente grande (10M+), porém ao aproximar-se das 50M de triplas, o desempenho deixa muito a desejar. Por esses motivos, pesquisamos soluções para Big Data e alternativas que lidam com número realmente grande de triplas. Chegamos ao OWLIM.

O OWLIM-SE foi escolhido para substituir o Sesame Native. Além de, atualmente, ser considerado o repositório de triplas mais escalável do mundo (ONTOTEXT, 2013), podendo carregar mais de 10 bilhões de triplas. O OWLIM utiliza toda a implementação da interface do Sesame, que já estava sendo utilizada nas APIs no projeto Java de triplificação. Por este motivo, migrar o projeto do Sesame Native para o OWLIM não foi uma tarefa trabalhosa.

Adicionalmente, de todas as estimativas do número de triplas a serem carregadas no banco, nenhuma chegou próxima ao número de 1 bilhão de triplas. Logo, essa parece ser uma boa e estável solução.

Quanto ao desempenho, as consultas que antes levavam quase 2 minutos agora levam menos de 10 segundos.

Uma ressalva dessa solução é que o OLWIM-SE, para ser minimamente eficiente para o número de triplas que temos no projeto, requer uma máquina com pelo menos 4 GB de memória RAM exclusiva.

4.3.2 ETC de Dados Gerais

Uma vez escolhido o SGBD RDF, pode-se dar início à fase de Extração, Transformação e Carga (ETC) dos dados do PingER. Essa é ainda subdividida em duas subfases totalmente independentes (logo, facilmente paralelizáveis): subfase de ETC de Dados de Medida de Rede (será visto na próxima seção) e a subfase de ETC de Dados Gerais. Em relação às tecnologias utilizadas, a API de Java Open RDF Sesame 2.7 (OPEN RDF, 2013) atua na camada de transformação em RDF e carga no BD.

Além dos dados específicos de medida de qualidade de rede, PingER LOD utiliza dados sobre conceitos gerais, tais como dados geográficos (continentes, países, estados, cidades), temporais, de universidade e dos nós de rede (monitores e monitorados). Cada um desses tipos de conceitos tem um processo independente de ETC (extração dos dados disponíveis, transformação em RDF e carga no repositório).

Ademais, todos os processos de ETC, de ambas as subfases, são executados automaticamente. Isto é, existe um sistema de **tarefas cronológicas** nos computadores do SLAC que executam os processos de tempo em tempo automaticamente para gerar dados triplificados, de acordo com a frequência agendada no sistema. Será visto na próxima subseção que para dados de medida agregados na unidade temporal **dia**, são armazenados dados dos últimos 60 dias somente. Porém, ao longo dos meses, devido à triplificação automática dos dados, PingER LOD contará com dados na unidade temporal **dia** dos últimos meses e até dos últimos anos.

Além disso, se dois processos forem agendados para executarem em paralelo, o sistema de tarefas cronológicas escalona um dos computadores disponíveis para executar a tarefa, tornando o sistema de ETC consideravelmente mais rápido.

Mostraremos, também, que em ambas as subfases, definimos padrões de URI HTTP. Utilizamos o prefixo "`:`" para indicar recurso referente à instâncias do domínio. Veja o Apêndice B para a definição de todos os prefixos.

Os processos da subfase de ETC de Dados Gerais são os seguintes:

a) Processo de carga de continentes

Os continentes, por constituírem um conjunto estático e pequeno de dados, são instanciados na memória do programa, já em formato RDF, e então carregados no repositório.

Frequência do processo: uma única vez – os dados sobre continentes que são utilizados não se alteram com o tempo.

Padrão de URI escolhido para continentes: `:Continent<ID no Geonames>`; Ex. `:Continent625515` refere-se ao continente América do Sul.

b) Processo de carga de países

O processo utiliza HTTP GETs[64] para acessar a API do Geonames[65] para recuperar um JSON com dados de todos os países. Então, o processo transforma cada entrada do JSON em um recurso RDF e suas propriedades e carrega-o no repositório.

Frequência do processo: uma única vez – os dados sobre países que são utilizados não se altera com o tempo

Padrão de URI escolhido para países `:Country<ID no Geonames>`; Ex. `:Country3469034` refere-se ao país Brasil.

c) Geração do `NodeDetails`

O processo utiliza um HTTP GET nos dados do PingER para extrair todos os nós[66] e suas informações (IP, Nickname, Latitude, Longitude, etc) e gerar o JSON auxiliar `NodeDetails`. Este JSON contém todos os nós e suas informações e é utilizado em seguida em outros processos de ETC.

Frequência do processo: diariamente – os nós podem ser adicionados, removidos ou alterados. Logo, os dados sobre nós devem ser atualizados no repositório diariamente.

d) Processo de carga de cidades e estados

Para cada nó (isto é, cada entrada do JSON `NodeDetails`), o processo executa HTTP GETs na API do Geonames para encontrar a `NearestTown` (cidade mais próxima com pelo menos 1.000 habitantes) e a `NearestCity` (cidade mais próxima com pelo menos 15.000 habitantes) baseando-se na latitude e longitude do nó. Quando se aplica, o estado no qual a cidade se encontra também é recuperado. O processo finalmente transforma em RDF todos os dados recuperados seguindo a ontologia especificada na seção 3.2.2 e carrega as triplas no repositório.

[64] http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html
[65] http://www.geonames.org/export/web-services.html
[66] http://www-iepm.slac.stanford.edu/pinger/pingerworld/nodes.cf

Frequência do processo: também diariamente pelo mesmo motivo de (c).

Padrão de URI escolhido para cidades: `:Town<ID no Geonames>`; Ex. `:Town5391959` refere-se à cidade São Francisco.

Padrão de URI escolhido para estados: `:State<ID no Geonames>`; Ex. `:State5332921` refere-se ao estado Califórnia.

e) Processo de carga de universidades

Assim como o processo anterior, para cada nó (isto é, cada entrada do JSON `NodeDetails`), o processo executa consultas no SPARQL Endpoint da DBPedia[67] para tentar encontrar a universidade cujo nome é similar com o nome completo do nó. Se for encontrada, dados sobre a universidade (número de alunos, pública ou particular, fundos monetários, etc) são recuperados, transformados e carregados no repositório de triplas. Em seguida, cada nó de rede que está em uma universidade encontrada nesse processo é associado à devida universidade. Uma abordagem melhor para esse processo precisa ser desenvolvida para recuperar mais dados significativos sobre universidades.

Frequência do processo: Semanalmente – porque realizar consultas na DBPedia para todos os nós de rede é um processo bem demorado.

Padrão de URI escolhido para universidades: `:<ID da Universidade para a Wikipedia/DBPedia>`; Ex. `:Stanford_University` refere-se à Universidade Stanford, na Califórnia.

f) Processo de carga de nós

Depois de triplificar continentes, países, cidades, estados e universidades, finalmente ocorre o processo de ETC dos nós de rede. Novamente, para cada nó (isto é, cada entrada do JSON `NodeDetails`), o processo triplifica um Nó de Rede utilizando a ontologia, ligando aos seus respectivos conceitos triplificados anteriormente. Triplifica também as informações do nó geradas na fase (c).

Frequência do processo: também diariamente pelo mesmo motivo de (c).

Padrão de URI escolhido para nós de rede: `:Node-<Nome do Nó>`;

Ex. : `Node-pinger.slac.stanford.edu` refere-se à nó de rede do PingER, no SLAC.

g) Outros parâmetros

É preciso instanciar ainda outros recursos, em geral utilizados nas medidas:

[67] http://dbpedia.org/sparql

- Métricas padrão (isto é, 11 instâncias de métrica com unidade padrão instanciada e ligada). Para as métricas com unidade default, o padrão URI escolhido foi: `:<Nome da Métrica>`; Ex. `:Throughput`
- Tempo. Um intervalo de tempo é um recurso e precisa ser instanciado. Instanciamos todos os tempos considerados pelo PingER LOD (seção 4.1.2). Padrão URI escolhido: `:Time<Padrão a ser exibido no displayValue>`; (Reveja `displayValue` em 4.2.2) Ex. `:TimeJan2008`
- Vimos (seções 4.1.2 e 4.2.3) que PingER LOD só considera tamanho de pacotes de 100 bytes. Logo, é necessário instanciar um indivíduo padrão para satisfazer a ontologia. Instanciamos a classe `PacketSize` (uri: `:PacketSize100`), definimos seu valor ("100"), ligamos à instância referente a bytes na classe `Unit` e, finalmente, ligamos a instância de `MeasurementDefault` à recém criada instância de `PacketSize`, através da propriedade `hasDefaultPacketSize`. Assim, definimos que todo `Measurement` (por ser filho de `MeasurementDefault`) tem tamanho de pacote padrão 100 bytes. Enfatizamos que ainda é possível definir outro tamanho do pacote, diferente do padrão, caso seja necessário.

Os processos listados são executados automaticamente e alguns deles podem ocorrer em paralelo. As únicas restrições são: (c) precisa ser necessariamente executado antes de (d), (e), e (f) porque utilizam o JSON `NodeDetails`; e (f) precisa acontecer depois de (d) e (e). Qualquer outro fluxo é possível. Entretanto, se esses processos executarem em fluxo diferente do listado, existe a possibilidade de existirem *links* quebrados. Por exemplo, se instanciar cidades antes de países, pode ocorrer a tripla (Rio de Janeiro, está no país, Brasil) antes de se instanciar o indivíduo Brasil no banco; a tripla pode existir porém o objeto apontará para um recurso que não existe no repositório. Isso não é um grande problema porque não impede o carregamento do restante dos dados e o *link* é automaticamente reparado quando o dado que está faltando é carregado no repositório. Essa é uma das grandes vantagens de se utilizar esquemas flexíveis.

A Figura 17 mostra um diagrama da sequência dos passos descritos anteriormente.

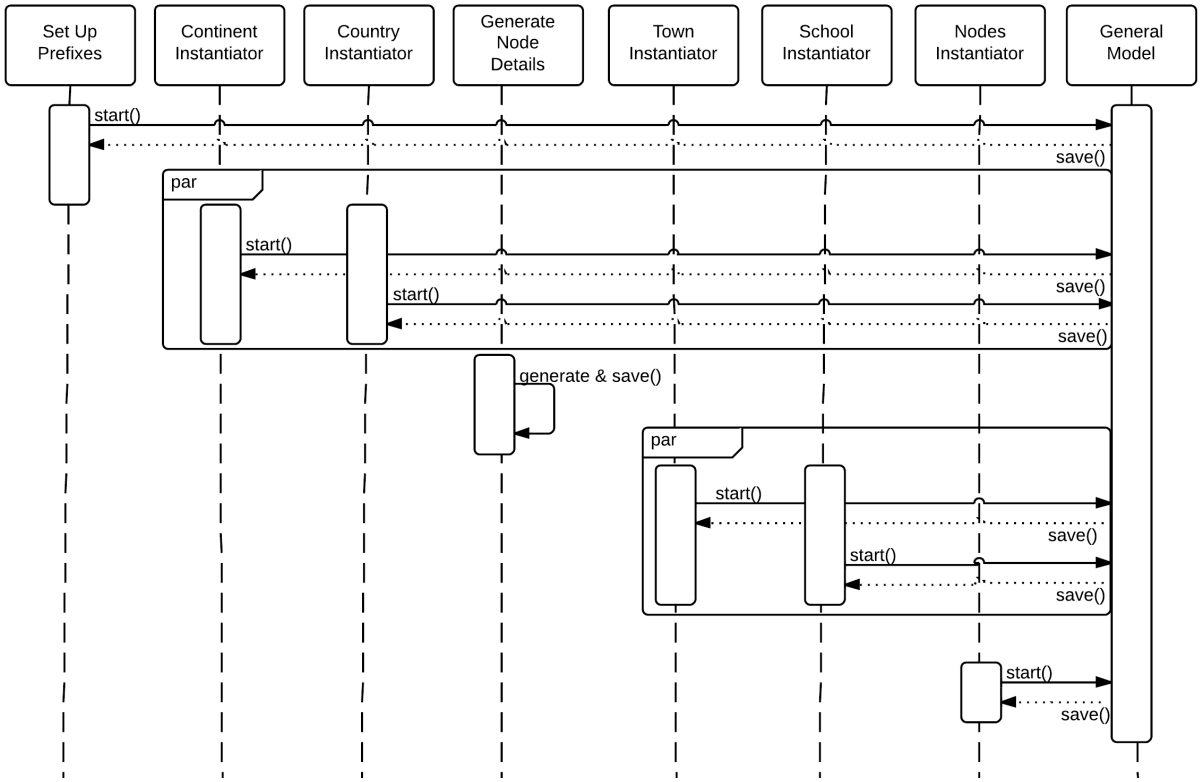


Figura 17 – Diagrama de sequência para dados gerais

4.3.3 ETC de Dados de Medida de Rede

A subfase de ETC de Dados de Medida de Rede é a mais importante porque triplifica dados de fato específicos do projeto PingER. É também a mais complexa por ter uma quantidade realmente massiva de dados, exigindo estratégias mais elaboradas para otimizar o processo de carga. Como já mencionado, esta subfase ocorre automática, paralela e independente da subfase anterior. Resumidamente, esta subfase extrai dados de vários CSV disponíveis cruzando parâmetros na Pingtable, depois os transforma em RDF e os carrega no repositório.

Para recuperar os arquivos CSV, a abordagem utilizada é a seguinte:

Para cada nó monitor (para cada entrada do JSON `MonitoringMonitored`, visto em seguida), para cada métrica, para cada parâmetro de tempo, o programa executa um HTTP GET na Pingtable para recuperar o arquivo CSV disponível para o cruzamento dos parâmetros determinados. Os HTTP GETs são sobre URLs da forma:

```
http://www-wanmon.slac.stanford.edu/cgi-wrap/pingtable.pl?
from=EDU.SLAC.STANFORD.N3&
to=WORLD&
file=average_rtt&
tick=allyearly&
```

```
format=tsv&by=by-
node&size=100&ex=none&only=all&dataset=hep&percentage=any
```

Onde os parâmetros:

- `from` – é o nó monitor.
- `to` – é o nó monitorado. O padrão é `WORLD` porque retorna um CSV com dados do cruzamento dos parâmetros desejados do nó monitor para *todos* (mundo) os nós que ele monitora. O projeto PingER LOD armazena somente medidas entre nós monitores para nós monitorados (desagregados), apesar do projeto PingER armazenar dados agregados de nós por países, regiões ou continentes.
- `tick` – representa a agregação temporal. Como visto (seção 4.1.2), PingER tem dados desde 1998 até os dias atuais, agregados por hora, dia, mês, ano. O projeto PingER LOD armazena as seguintes agregações:
    - `allyearly` – agregação das medições por ano.
    - `allmonthly` – agregação das medições por meses, para todos os anos.
    - `last60days` – agregação das medições por dia, para os últimos 60 dias.
- `file` – representa o nome da métrica de rede. PingER LOD está considerando somente as seguintes métricas:
    - `Mean Opinion Scores`
    - `Directivity`
    - `Average Round Trip Time`
    - `Conditional Loss Probability`
    - `Duplicate Packets`
    - `Inter Packet Delay Variation`
    - `Minimum Round Trip Delay`
    - `Packet Loss`
    - `TCP Throughput`
    - `Unreachability`
    - `Zero Packet Loss Frequency`

Ou seja, cada arquivo CSV é relativo a **um nó monitor**, a **uma métrica** e a **uma agregação temporal**.

Uma vez entendido como os arquivos CSV são recuperados, pode-se seguir para a segunda e última subfase de ETC dos dados do PingER. Um processo de ETC dos Dados de Medida de Rede se dá em relação a **uma única métrica** e a **uma única agregação temporal**. Cada um desses processos são independentes e estão sendo executados paralelamente de

acordo com as configurações do sistema tarefas cronológicas do SLAC, vistos na seção anterior. Para popular completamente o repositório, é preciso realizar todo o fluxo de tarefas a seguir para todas as 11 métricas consideradas no projeto e para cada um dos 3 tipos de agregações temporais consideradas. Cada um dos processos independentes executa o seguinte fluxo de tarefas:

a) Gerar o JSON auxiliar `MonitoringMonitored`

Este JSON contém a informação de todos os nós monitores e seus nós monitorados. Ele é gerado executando HTTP GET para recuperar os nós monitores[68] e, para cada nó monitor, executar um HTTP GET para listar todos nós monitorados[69] por aquele nó monitor.

b) Gerar o JSON `MonitoringNodesGroupedForTSV`

Executa-se um HTTP GET na lista dos nós monitores do PingER. Dessa lista, os nós monitores são agrupados em `M/T` grupos e armazenados no JSON `MonitoringNodesGroupedForTSV`. Onde `M` é o número de nós monitores e `T` é o número de threads que farão download dos CSV necessários. Durante o desenvolvimento do projeto, `M = 80` e `T = 20`.

c) Download dos arquivos CSV necessários (`GetPingTableTSVThreadsStarter`)

Divide-se o número de nós monitores (`M = 80`) em `T = 20` threads, de modo que cada thread seja responsável por 4 `(=M/T)` nós monitores. Cada thread executa 1 HTTP GET para cada nó monitor para baixar o arquivo CSV necessário, de modo que tenham 20 threads fazendo download paralelamente. Isso é feito 4 vezes até baixarem os 80 arquivos CSV referentes aos 80 nós monitores, àquela métrica e àquela agregação temporal.

d) Paralelizando o processamento dos CSV (`MonitoringNodesThreadsStarter`)

Após baixar os `M = 80` arquivos CSV, 1 para cada nó monitor, é necessário processá-los. A estrutura `MonitoringNodesGrouped`, que agrupa os nós monitores em `K` grupos, permite paralelizar esta parte também: cada thread é responsável por processar `M/K` arquivos CSV. Atualmente, o programa dedica 1 thread para cada arquivo CSV (isto é, `K = M`), de modo que 80 arquivos CSV são processados paralelamente.

[68] http://www-wanmon.slac.stanford.edu/cgi-wrap/dbprac.pl?monalias=all)

[69] http://www-wanmon.slac.stanford.edu/cgi-wrap/dbprac.pl?monalias=MONITORING_NODE&find=1

e) Gerando NTriples a partir de CSVs (`MeasurementInstantiatorThreadsStarter`)

As threads mencionadas anteriormente processam os arquivo CSV separadamente. Entretanto, foi visto que cada arquivo CSV referente a um nó monitor contém dados sobre todos nós monitorados daquele nó monitor.

Cada thread referente a um arquivo CSV lança `N` mais threads (onde `N` é o número de nós monitorados pelo nó monitor referente ao arquivo CSV sendo processado).

Finalmente, essa thread triplifica os dados referentes a uma medida bem específica: de um determinado nó fonte, para um determinado nó destino, utilizando a determinada métrica para qualidade de rede, para cada período de tempo. Essa medida é salva no arquivo de NTriples.

f) Carga do arquivo NTriples no repositório

Finalmente, depois de ter todos os dados dos arquivos CSV triplificados em um arquivo separado, este arquivo é carregado no repositório de triplas.

Todas essas paralelizações foram decisivas na redução do tempo gasto para extrair, processar, transformar e carregar todos os dados dos arquivos CSV no repositório. Além disso, foi experimentalmente observado que carregar as triplas no banco assim que elas são geradas (isto é, milhares de conexões com o banco), tornava o processo consideravelmente mais lento. Por isso, foi escolhido salvar todas as triplas em um arquivo NTriples separado e depois carregá-lo inteiro de uma só vez no repositório. Isso também ajudou a reduzir o tempo de carga.

O padrão de URI escolhido para as medidas é: `:M-<Apelido do Nó fonte>_<Apelido do Nó destino>_<Nome da Instância da métrica de rede>_<Parte identificadora da URI de instâncias de tempo>`;

Ex. : `M-EDU.SLAC.STANFORD.N3_BR.UFRJ.N1_Throughput_26Mar2008`

A Figura 18 mostra um diagrama da sequência dos passos descritos anteriormente.

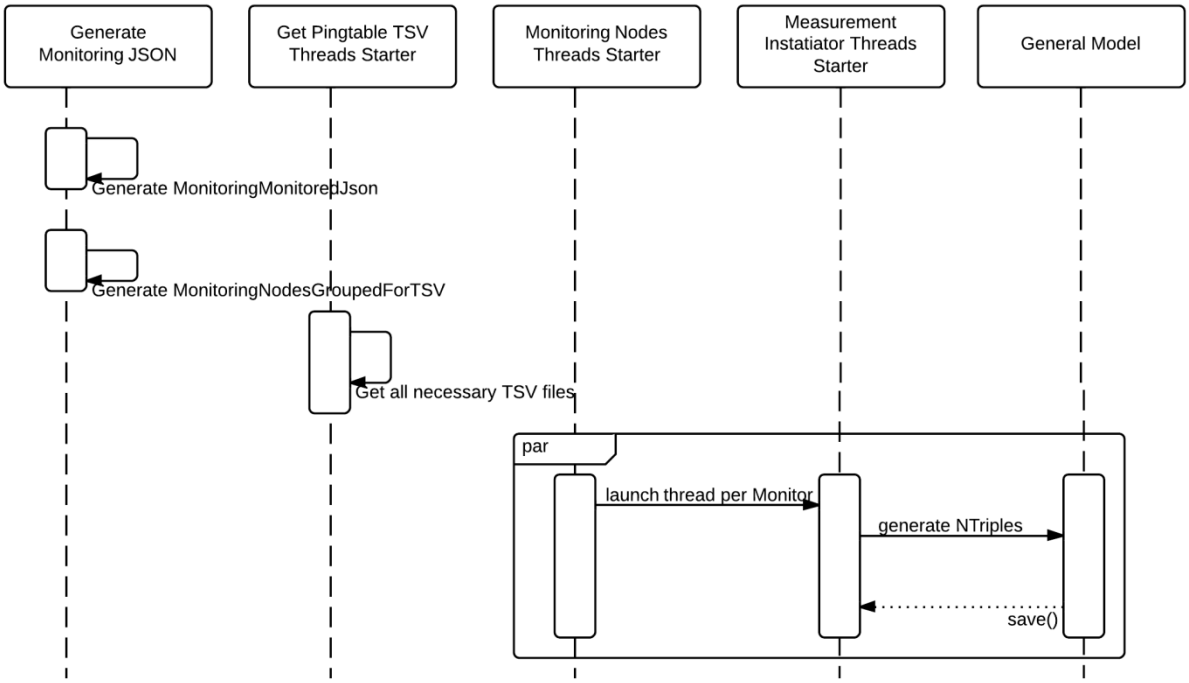


Figura 18 – Diagrama de sequência para instâncias de medidas de rede

O projeto PingER LOD hoje já contém mais de 50M de triplas, dentre as quais muitas são ligadas com recursos já existentes na nuvem de LOD. Este número tende a crescer ao longo do tempo com a execução automática dos processos disparados pelo sistema de tarefas cronológicas dos servidores do SLAC.

4.4 Acesso Público aos dados do PingER LOD

Foi construído um *website* que contém as informações sobre o projeto PingER Linked Open Data, bem como o *link* para o SPARQL Endpoint e documentação da ontologia.

Para o SPARQL Endpoint, foi construída uma aplicação-servidor em Java para processar consultas SPARQL vindas de requisições (GET ou POST), inseridas através de uma interface HTML, utilizando a API do Sesame 2.7 (OPEN RDF, 2013) para consultar a base e gerar resultados em JSON, XML ou CSV.

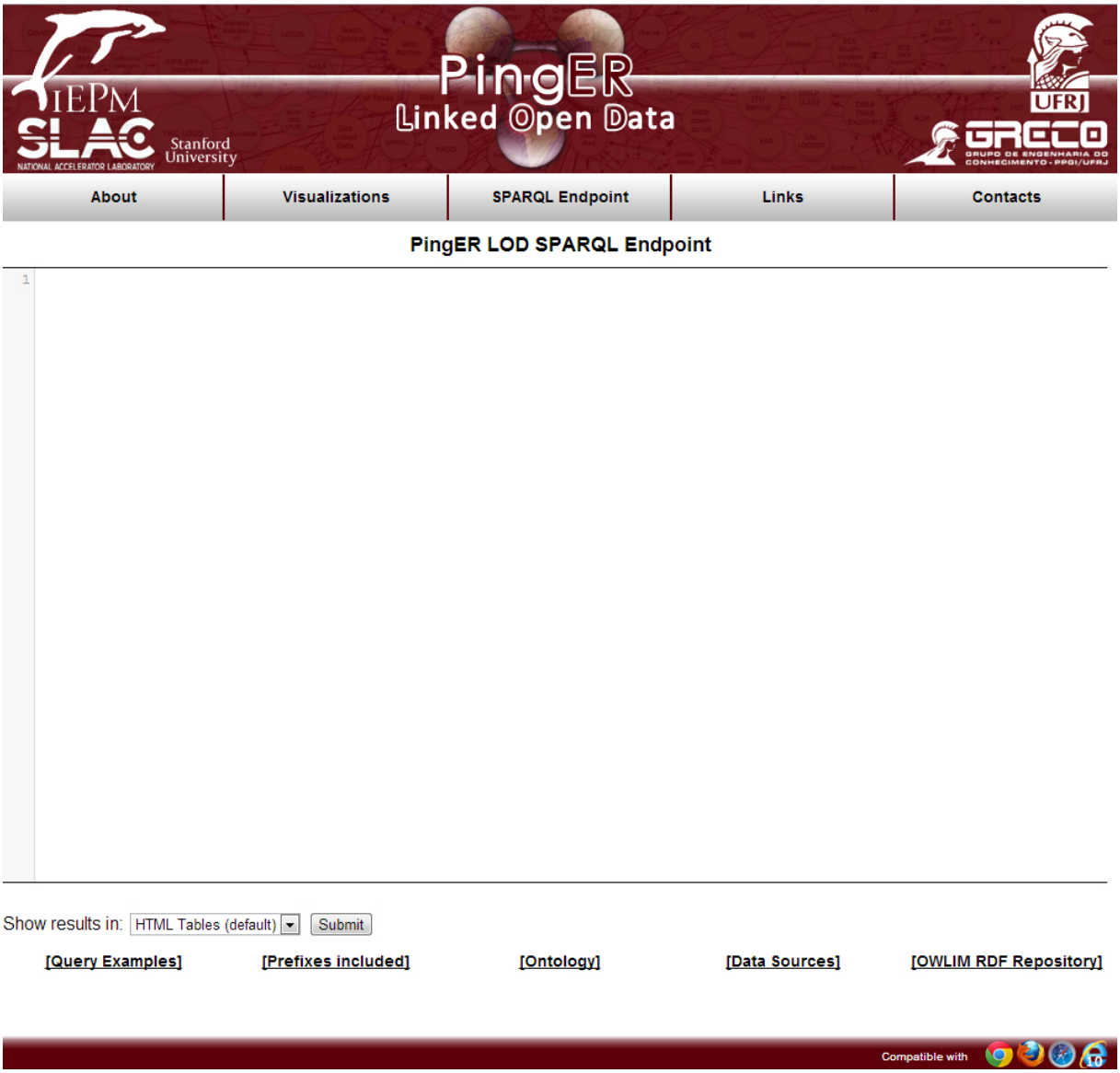


Figura 19 – Interface para o SPARQL Endpoint do PingeR LOD

Em adição à documentação da ontologia, foi desenvolvida uma aplicação web utilizando JavaScript, JQuery[70] e a biblioteca gráfica Raphaël[71] para mostrar uma visualização da ontologia em forma de um grafo rotulado direcionado, além da descrição de cada classe e propriedades.

[70] http://jquery.com/
[71] http://raphaeljs.com/

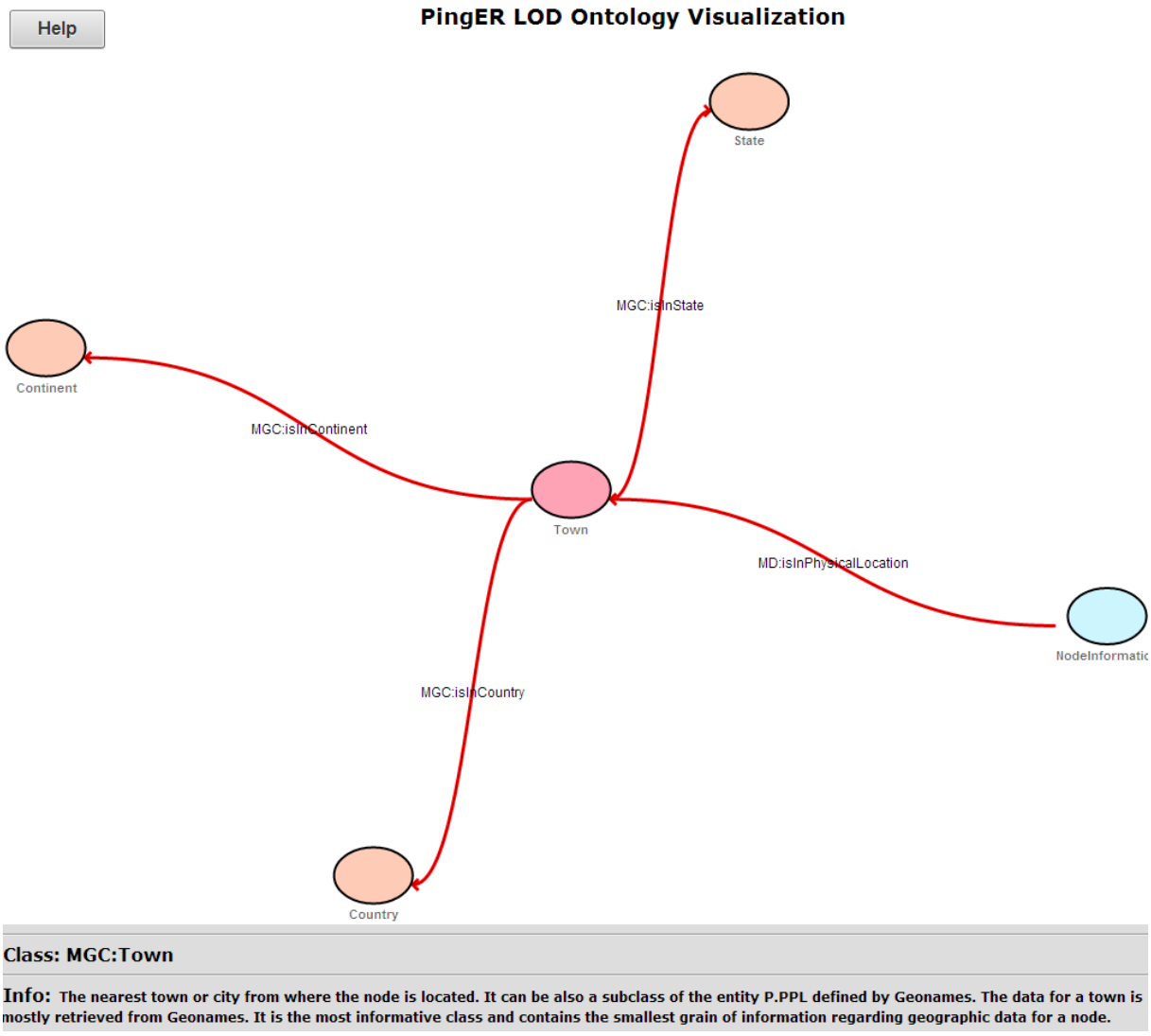


Figura 20 – Aplicação de visualização da ontologia do PingER

Para o futuro, será verificada uma maneira de disponibilizar um *Dump RDF* de toda base e a possibilidade de adicionar a base do projeto na LOD Cloud, no SPARQLES e no DataHub (ferramentas vistas na seção 2.5.2).

4.5 Aplicações para o projeto PingER Linked Open Data

Uma vez estabelecida uma base de dados RDF para o PingER LOD e um SPARQL Endpoint capaz de receber consultas SPARQL e enviar seus resultados, é possível desenvolver diversas aplicações que consumam os dados e explorem todas as vantagens de LOD (vistas na seção 2.5). Dentre as vantagens citadas, as aplicações desenvolvidas sobre este projeto destacam as seguintes: facilidade de acesso através do SPARQL Endpoint público; expressividade semântica dos dados devido à propriedade granular das triplas (seção 2.3.1);

possibilitando verbosamente descrever o domínio; especificidade dos resultados de uma consulta graças ao uso de um esquema (proporcionado pela ontologia) com um grau significativo de estruturação e manipulação de formatos de arquivo de fácil processamento por máquinas (a saber, JSON, XML e CSV); e interoperabilidade entre banco de dados interligados e de naturezas diversificadas. Esta seção exemplifica aplicações que exploram e evidenciam essas vantagens.

4.5.1 Análise simultânea de múltiplas métricas de rede

Este caso utiliza somente dados do domínio do PingER, isto é, medida de qualidade de rede de nós em vários países ao longo de um período de tempo. O caso serviu para exemplificar como os dados em LOD auxiliam mesmo se não utilizar *mashups* com outras bases de domínios diversos. Destaca a vantagem dos dados estarem bem estruturados por um esquema e por estarem em um formato bem expressivo: as triplas. Explora também o uso de consultas SPARQL complexas que conseguem capturar precisamente o que se deseja. Antes do projeto, era bem difícil e trabalhoso mostrar em um único gráfico várias métricas de rede simultaneamente. Era necessário utilizar a Pingtable (seção 3.1.2.1) para cada métrica, exportar os dados em algum *buffer* (por exemplo, um arquivo Excel) e então gerar o gráfico a partir da mescla entre os dados desse buffer.

Uma consulta SPARQL sobre o banco de dados do PingER LOD consegue recuperar a medida em relação a qualquer combinação possível (isto é, para a qual existe algum dado) de parâmetros especificados, inclusive unindo várias métricas.

Como visto (seção 2.4.2), os resultados da consulta em um SPARQL Endpoint são retornados em XML. Porém, a maioria das APIs possibilita retornar o resultado em outros formatos como CSV e JSON. O formato JSON foi escolhido porque a aplicação que constrói o gráfico foi feita em JavaScript, utilizando a biblioteca de gráficos Highcharts[72].

Como interface para o usuário inserir os parâmetros da consulta, foi construído um formulário HTML. Nessa interface, o usuário define os seguintes parâmetros:

- `Pinging From`: Origem do *ping*. Valores possíveis são: nó monitor, isto é, o nó de rede que enviou o *ping;* país de origem, isto é, uma agregação de todos os nós monitores que estão em um determinado país; ou continente de origem, isto é, uma agregação de todos os nós monitores que estão em um determinado continente.

[72] http://www.highcharts.com/

- `Pinging To`: Destino do *ping*. Os valores possíveis são análogos à origem do *ping*, porém em relação aos nós de rede monitorados.
- `Time Parameters`: Definição do período de tempo para o qual se tem interesse em investigar as medidas de rede. É possível especificar a unidade temporal (dia, mês ou ano) a ser utilizada na consulta.
- `Metrics`: Selecionar as métricas a serem mostradas simultaneamente no gráfico.

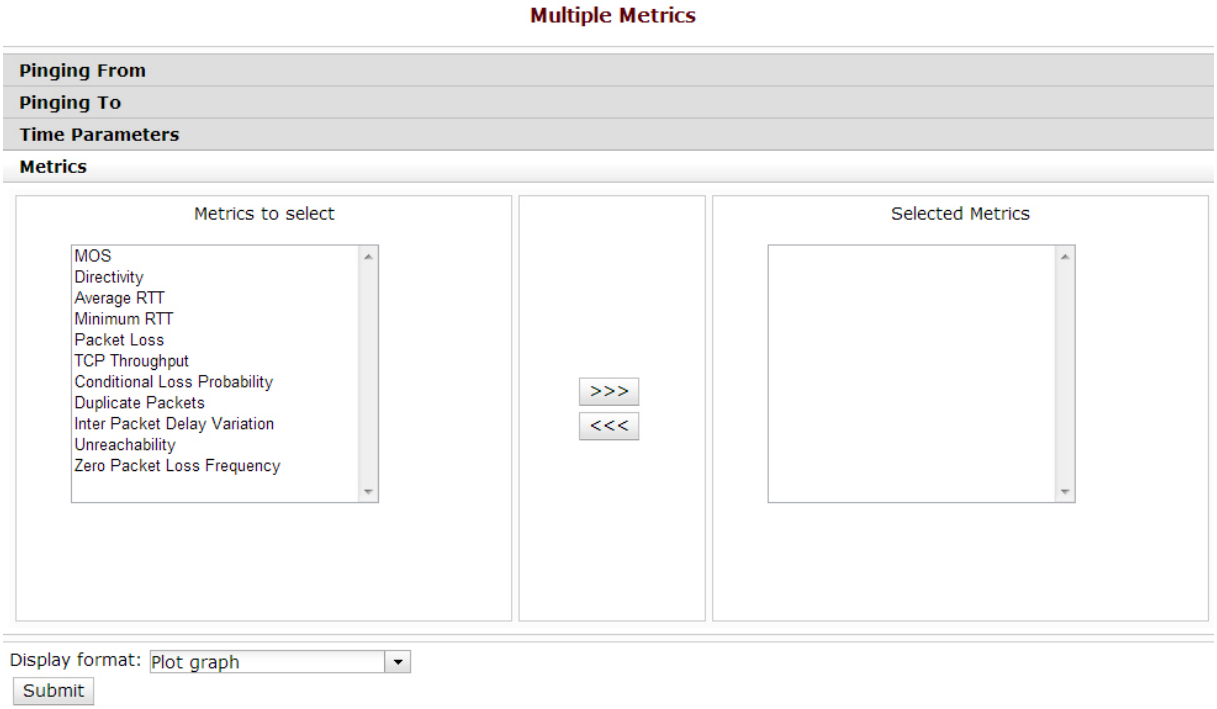


Figura 21 – Interface para a seleção dos parâmetros para gerar a consulta SPARQL

Após a especificação dos parâmetros, escolhe-se o formato de saída. É possível mostrar os resultados em CSV, JSON ou tabelas HTML. Pode-se também gerar a consulta SPARQL a ser utilizada no Endpoint do projeto. Finalmente, pode-se dinamicamente plotar o gráfico mostrando os resultados das medidas de várias métricas simultaneamente, que é o objetivo.

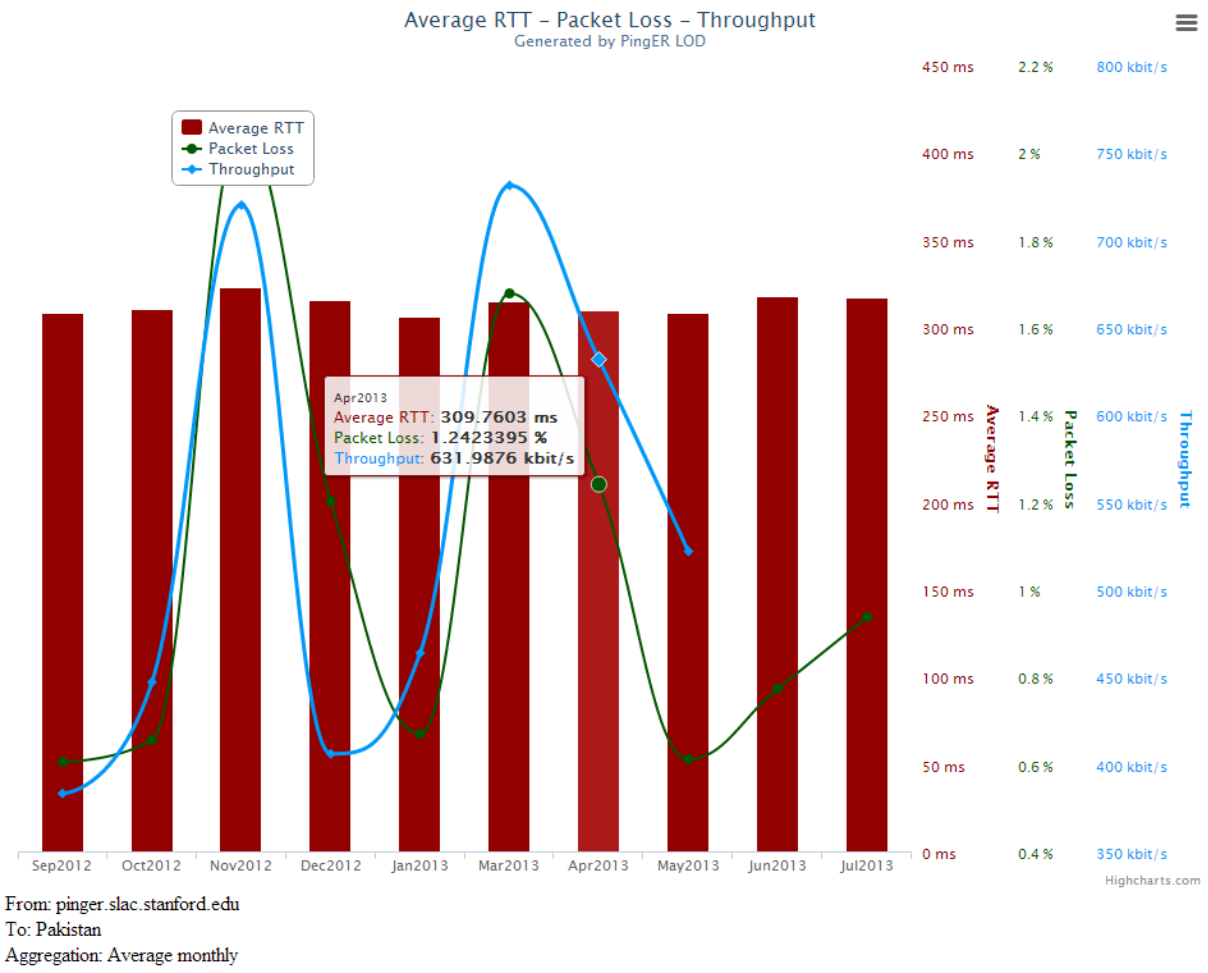


Figura 22 – Múltiplas métricas de rede mostradas simultaneamente.

A Figura 22 mostra o gráfico plotado utilizando os parâmetros: *pinging from* PingER SLAC, *to* Paquistão, período de tempo de agosto de 2012 até agosto de 2013 (unidade temporal **mês**), selecionado simultaneamente as métricas `Packet Loss`, `TCP Throughput` e `Average RTT`. A consulta SPARQL gerada por esse exemplo pode ser vista no Apêndice C.

O gráfico revela, simultaneamente, o quanto a perda de pacotes e throughput variaram ao longo do intervalo de tempo especificado mas a média de RTT permaneceu quase que invariável.

4.5.2 Métricas de Rede x Métricas de Universidades

Foi necessário voltar à fase da análise do domínio para verificar mais aplicações de como os dados em LOD do PingER poderiam ser úteis. Constatamos que a grande maioria dos nós (ambos monitores e monitorados) ao redor do mundo considerados pelo projeto eram de origem de universidades. Disso, estudamos como seria possível mesclar os dados do

PingER, de natureza de medidas de rede, com dados de natureza a princípio completamente diferentes, os de universidades. Este caso exemplifica o poder da interoperabilidade dos dados em LOD através de *mashups,* já que o estudo desse caso só foi possível porque os dados estão nesse formato.

Foi levantada uma questão a ser investigada: qual será a relação entre a qualidade da rede das universidades e a qualidade da universidade?

Para medir a qualidade da universidade, foi decidido utilizar a DBPedia para recuperar as métricas de universidade e o processo de triplificação precisou ser revisto para incluir os novos dados, como visto na seção (3.2.2.2.e). A consulta retorna dados de medida de rede medidos a partir de uma fonte (ou nó monitor ou um agregação regional – país ou continente) para todos os nós monitorados que se situam em uma universidade e foram capturadas no processo de triplificação. Também se especifica o período de tempo a ser investigado e as métricas de rede e de universidade a serem comparadas.

Assim como o caso anterior de múltiplas métricas simultâneas, também foi construída uma interface HTML para que o usuário especifique os parâmetros da consulta e escolha o formato de saída. Ao escolher Plotar gráfico, será construído um mapa utilizando a API de JavaScript do Google Maps[73] consumindo um JSON com o resultado da consulta SPARQL.

[73] https://developers.google.com/maps/documentation/javascript/

Figura 23 – Mapa comparando métricas de universidades (quanto maior o círculo, maior o número de alunos) e métricas de rede (quanto mais branco, maior o valor do *throughput*).

A Figura 23 mostra o gráfico plotado utilizando os parâmetros: *pinging from* PingER@SLAC, período de tempo de fevereiro de 2010 até março de 2011 (unidade temporal **mês**), selecionado simultaneamente as métricas de rede `TCP Throughput` e de universidade número de alunos.

O mapa mostra círculos mapeados utilizando a latitude e longitude do nó de rede e que variam em diâmetro, cor de preenchimento e cor da circunferência. O significado de cada variação se dá como o seguinte:

- O diâmetro do círculo representa a métrica de universidade (número de alunos, fundos arrecadados, etc). Quanto maior o diâmetro, maior o valor da métrica de universidade escolhida.

- A cor de preenchimento do círculo representa a métrica de rede (ex: *Throughput, Packet Loss, RTT,* etc). Quanto mais branco, maior é o valor da métrica de rede escolhida.
- A cor da circunferência indica o tipo de universidade (pública, privada, etc).

Ao clicar no círculo no mapa, são mostradas também informações sobre o nó de rede monitorado pelo PingER bem como *links* para bancos de dados RDF externos aos quais os dados do PingER LOD estão ligados.

No exemplo do gráfico da figura anterior, observamos que as universidades dos Estados Unidos apresentaram melhores resultados. As universidades da América do Sul e da África aparecem com *throughput* bem mais escuros (menores).

A consulta SPARQL gerada por esse exemplo pode ser vista no Apêndice D.

4.5.3 Métricas de Rede x PIB dos países Desenvolvimento e Pesquisa

Em adição aos estudos já realizados, resolveu-se explorar ainda mais as vantagens dos *mashups* e, então, investigar a relação entre qualidade de rede e investimento dos países (% do PIB) em pesquisa e desenvolvimento tecnológico, ao longo dos anos.

O BD RDF do World Bank (2.3.2.d) contém as informações necessárias sobre os investimentos do PIB dos países. Precisamos de alguma maneira mesclar os dados do PingER com os dados do World Bank para realizar o *mashup*.

As possibilidades são:

a) Recuperar os dados do World Bank e carregá-los na base local do PingER LOD.

Em teoria, essa abordagem possibilitaria resolver a consulta desejada com uma única consulta SPARQL bem rápida, já que os dados das duas bases diferentes estariam no mesmo local, como foi no caso das universidades da seção anterior. Entretanto, isso é ruim porque vai precisar adicionar à base do PingER LOD dados de natureza bem diferentes da proposta (desempenho de rede) e, além disso, quebra a ideia do LOD de ter vários bancos separados distribuídos mas que enxergados como um só.

b) Executar uma consulta SPARQL Federada.

Essa seria a melhor das opções porque é a maneira certa, padrão e recomendada de fazer *mashups* e realmente resolveria o problema em apenas 1 única consulta, acessando os 2 banco de dados RDF diferentes. Entretanto, vimos em 2.4.4, que consultas SPARQL Federadas ainda possuem muitas limitações. A consulta criada para tentar responder a essa

consulta demorou mais de 1 hora até que foi cancelada porque mostrou ser ineficiente demais. Ela pode ser vista no Apêndice E.

c) Executar duas consultas separadas em cada um dos Endpoints.

Um programa Java envia duas consultas SPARQL, uma para cada Endpoint (PingER LOD e World Bank), recebe os resultados, mescla eles (fazendo uma espécie de "*join*" controlado programaticamente) e gera um JSON o qual é usado para construir o mapa utilizando a API do Google Maps, semelhante à seção anterior, porém incluindo a informação temporal na visualização. Essa foi a solução escolhida porque não incluía mais dados de natureza muito diferentes no PingER LOD e não demorava muito para retornar os resultados desejados.

As duas consultas SPARQL para resolver essa consulta podem ser encontradas nos Apêndices F e G. O resultado pode ser visto na Figura 24.

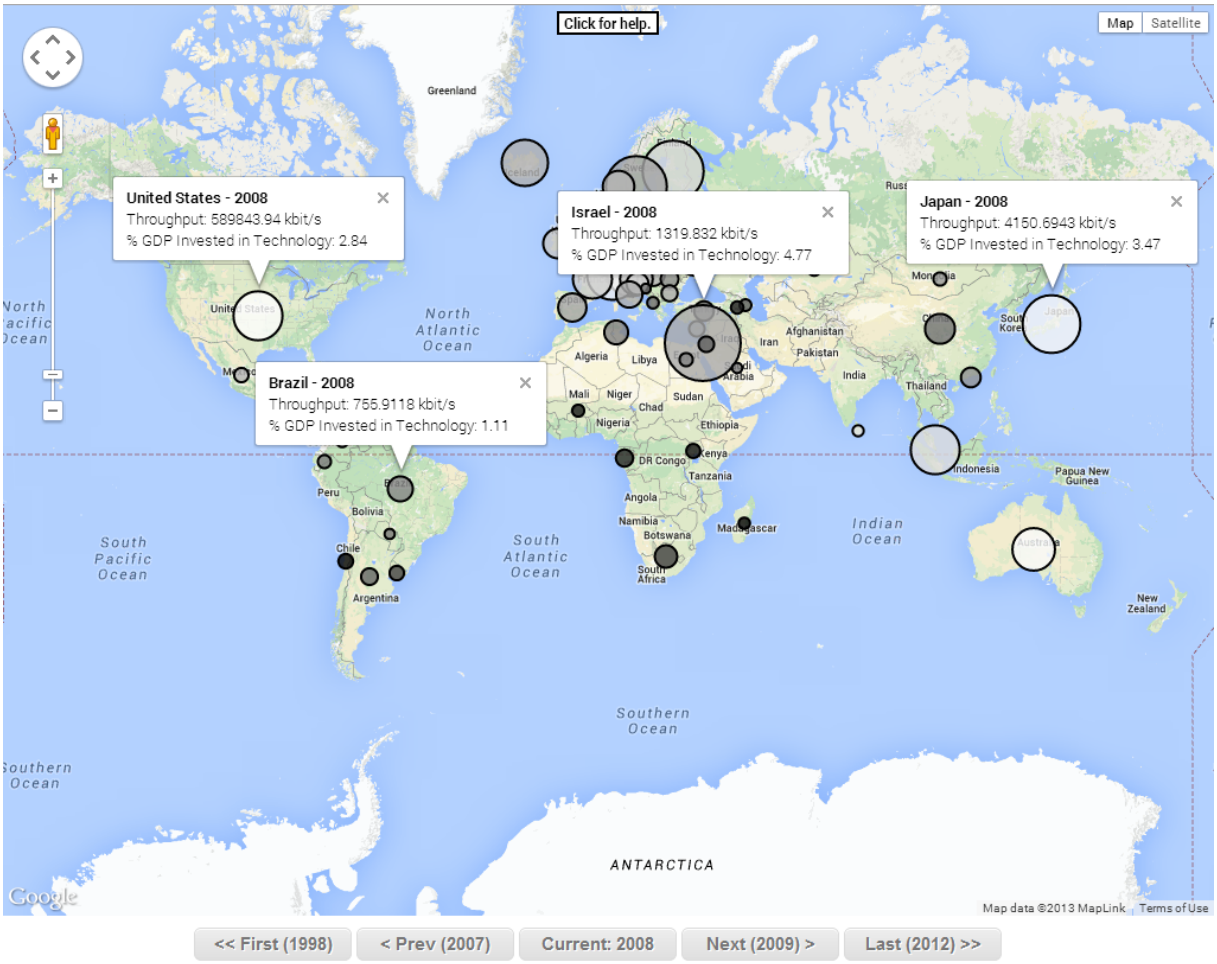


Figura 24 – Percentual do PIB em Tecnologia x Medida de desempenho de rede

O tamanho do círculo significa o quanto é investido (quanto maior o círculo, mais o país investe) e a cor mede o valor da métrica de rede (quanto mais branco, maior o valor). A

relação entre % PIB e a métrica de rede pode ser visualmente acompanhada ao longo dos anos (navegando pelo gráfico através dos botões `Prev` e `Next`), desde quando o PingER começou registrar seus dados (1998) até o ano anterior ao corrente.

Pelo gráfico, observamos que a maioria dos círculos mais claros são os maiores (maior diâmetro). Ou seja, como era de se esperar, em sua maioria, países que investem mais em tecnologia possuem melhor qualidade de rede.

4.5.4 Dados do PingER LOD como entrada para o CubeViz

Para exemplificar mais aplicações dos dados do PingER LOD, Ferman (2013) utilizou os dados triplificados pelo projeto como entrada da aplicação de visualização de dados em RDF utilizada em seu trabalho de conclusão do curso. A aplicação opera sobre o projeto OntoWiki (TRAMP *et al.,* 2010; AUER *et al.*, 2012b) junto com a extensão CubeViz que facilita, através de variados tipos de gráficos, a visualização de dados estatísticos em RDF, que cruzam 2 parâmetros (por exemplo, região geográfica e tempo) e quantificam uma métrica (por exemplo, *throughput* da rede). A aplicação desenvolvida por Ferman (2013) acrescenta novas funcionalidades ao CubeViz possibilitando mostrar múltiplos parâmetros e múltiplas métricas simultaneamente, além de favorecer a agregação pela hierarquia natural dos dados (exemplo, agregar todas as medidas de todas as cidades de um determinado país).

Para um exemplo aplicado aos dados do PingER LOD, selecionamos o seguinte cenário: Pings do nó monitor no SLAC para todas as cidades monitoradas no Brasil (i.e., cidades brasileiras para as quais tem-se nós monitorados), agregando por ano, para os anos 2010, 2011, 2012, quantificando as métrica de rede **perda de pacotes** e ***throughput*** .

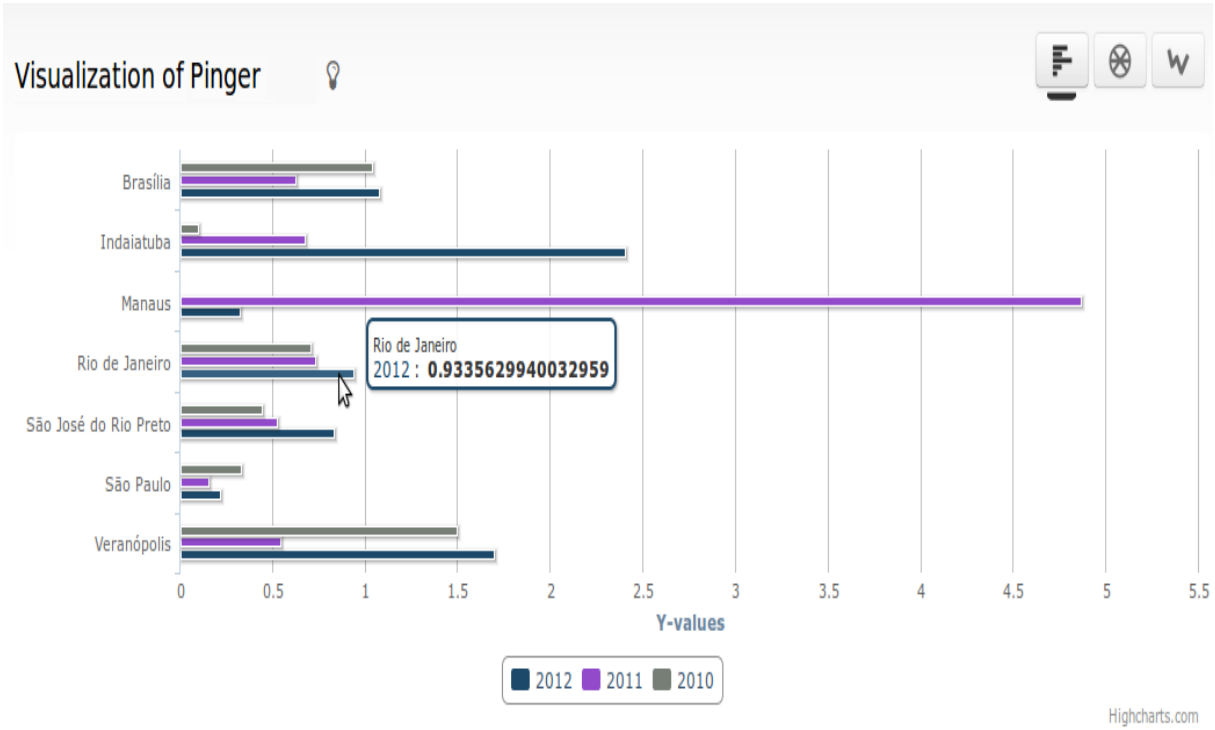


Figura 25 – Gráfico em barras de perda de pacotes gerado pela extensão do CubeViz, desenvolvido por Ferman (2013).

A Figura 25 mostra a combinação dos parâmetros para a métrica perda de pacotes. Pelo gráfico, observa-se, por exemplo, que em 2011, Manaus teve perdas de pacotes consideravelmente elevadas.

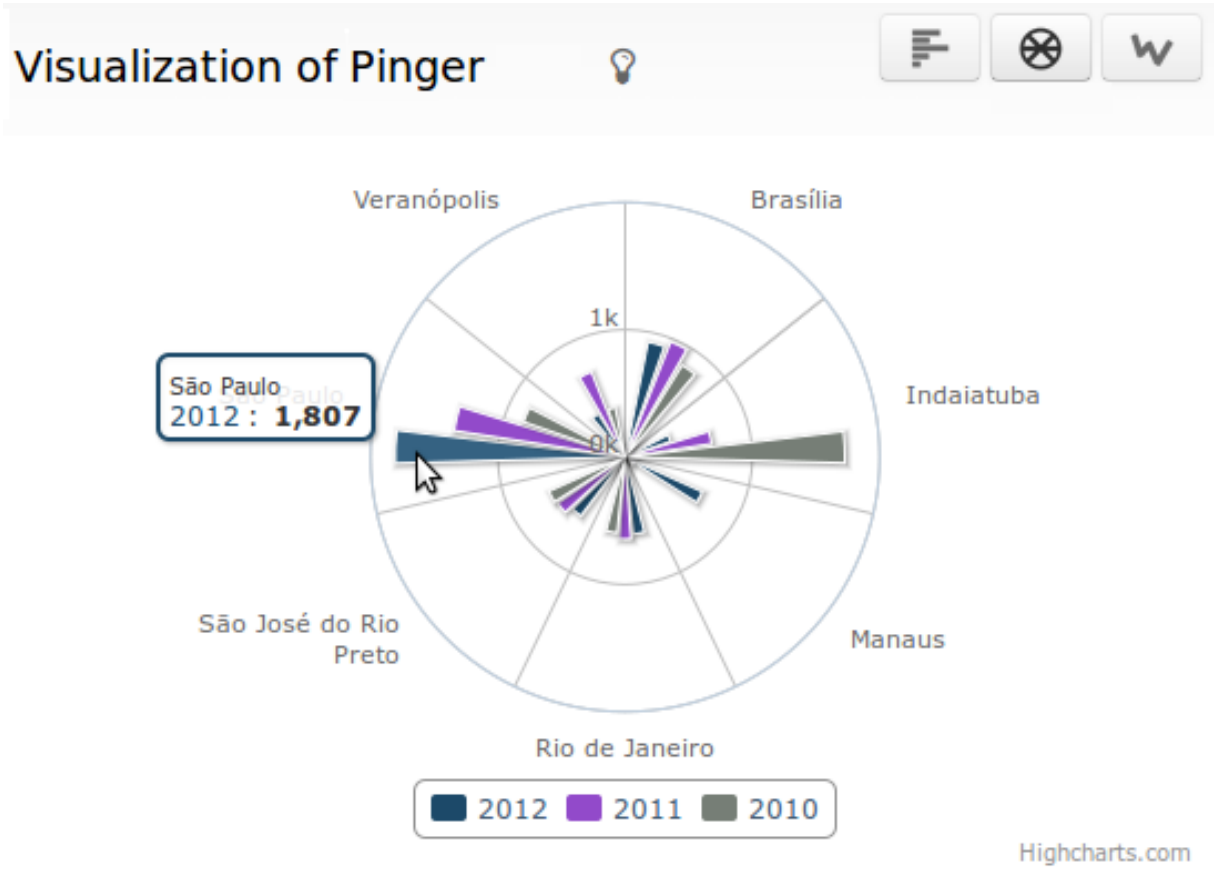


Figura 26 - Gráfico radial de *throughput* gerado pela extensão do CubeViz, desenvolvido por Ferman (2013).

A Figura 26 mostra a combinação dos parâmetros para a métrica *throughput*. Pelo gráfico, observa-se, que a cidade de São Paulo apresentou o melhor *throughput* em 2012 e Indaiatuba venceu em 2010.

A grande vantagem de se utilizar esse tipo de ferramenta é que não foi necessário codificar nenhuma consulta SPARQL para gerar a visualização. Entretanto, a ferramenta espera que os dados estejam necessariamente expressos segundo o *Data Cube Vocabulary*[74], que é um vocabulário recomendado para publicar dados modelados multidimensionalmente. Caso os dados não estejam expressos nesse vocabulário, é necessário transformá-los através de um processo de ETC.

4.6 Transições entre as fases

Como vimos na introdução deste capítulo, mostramos apenas as conclusões de cada fase, isto é, como cada uma delas está atualmente no final do projeto. Entretanto, para chegar

[74] http://www.w3.org/TR/vocab-data-cube/

até o final, ocorreram muitas interações entre as fases e várias vezes foi necessário mudar a estratégia, rever a solução e implementá-la. É importante ressaltar, novamente, que, apesar de ser normal voltar uma ou mais fases do processo, se uma análise mais cautelosa de cada uma delas fosse realizada, muitas dessas interações seriam poupadas logo menos tempo seria gasto no processo até chegar à fase das Aplicações. Tempo esse que poderia ser investido em novas aplicações ou outras otimizações ao invés de voltar para fases anteriores. Nesta seção vamos mencionar superficialmente algumas das interações específicas que ocorreram no projeto PingER LOD.

Na fase da Análise do Domínio, na seleção dos dados, foi considerado publicar em LOD os dados de medida com a dimensão temporal em seu menor grão (maior detalhe) disponível, isto é, em hora. Todavia, o líder do PingER alertou que seriam dados demais e por isso selecionamos (ingenuamente) publicar todos os dados diários (grão dia), desde 1998.

Já na fase de triplificação dos dados, ao tentar carregar todos os dados diários, verificamos que o tempo para carregar todas as triplas seria impraticável. Então, voltamos novamente à fase de seleção dos dados e tentamos triplificar apenas dados diários dos últimos 365 dias. O processo de ETC desses dados também ficou extremamente longo, impraticável. Finalmente, resolvemos selecionar apenas dados dos últimos 60 dias e então sim a quantidade de dados ficou palpável e o tempo de ETC ficou razoável. Além disso, ao longo dos meses com as tarefas sendo automaticamente executadas, teremos dados dos últimos 365 dias. E isso foi bom o suficiente para o líder do PingER.

Além dessas interações, até chegar naquele modelo da ontologia apresentado na seção 4.2, fizemos várias sugestões e alternativas. Tentamos utilizar a ontologia MOMENT respeitando a estrutura fundamental dela e fizemos todo o processo de ETC utilizando-a como esquema dos dados. Ao rodar as consultas (fase das aplicações e consumo dos dados), verificamos que ela atrapalhava o desempenho e algumas das relações possuíam desvio de semântica. Tivemos que voltar a essa fase frequentemente para ajustar ou remodelar completamente a ontologia para o projeto PingER LOD.

Outras interações ocorreram que nos fizeram transitar várias vezes entre as fases do processo, mas elas não foram tão expressivas quanto às mencionadas.

5 CONCLUSÃO

5.1 Considerações finais

Esse trabalho de conclusão de curso apresentou uma visão essencialmente prática de um Processo de Publicação de Linked Open Data e mostrou a aplicação dele em um cenário de dados de medida de desempenho de rede, o projeto PingER, operado pelo SLAC Stanford Linear Accelerator Laboratory.

No capítulo 2, foram reunidos os principais aspectos conceituais e tecnológicos da Web Semântica, utilizando uma abordagem prática objetivando publicar dados em padrões LOD. Como resultado, o material produzido naquele capítulo pode ser aproveitado para mostrar resumidamente os tópicos fundamentais envolvidos na área.

No capítulo 3, foi introduzido o Processo de Publicação de Linked Open Data, o qual foi proposto após várias dificuldades enfrentadas durante o desenvolvimento do projeto. O processo proposto orienta como desenvolver, desde a concepção, um projeto de publicação de LOD e, se for seguido, as dificuldades para publicar seriam aliviadas. Como contribuição à comunidade, esse processo poderia ser publicado porque serviria de material de referência fácil e resumido, ideal para novatos na área a fim de publicar seus dados em LOD.

No capítulo 4, mostramos a aplicação do processo sobre os dados de medida de desempenho de rede, os dados do PingER. Toda a evolução do projeto PingER LOD foi apresentada, desde a fase inicial de concepção até o final, com as aplicações e consumo dos dados. As dificuldades enfrentadas e as vantagens proporcionadas ao PingER e à comunidade foram ressaltadas em cada fase do processo, de acordo com suas características específicas. Destacadamente, a tentativa de reúso da ontologia MOMENT gerou uma análise da ontologia que pode ser utilizada para sugerir melhorias. Além disso, a ontologia proposta pelo projeto pode ser reutilizada (estendida ou adaptada) para outros projetos de domínio de medidas de rede.

Portanto, o desenvolvimento desse trabalho permitiu extenso consumo e produção de conhecimento e aplicações *de* e *para* uma proeminente área da Ciência da Computação e Informação: Web Semântica. Adicionalmente, foi de expressiva utilidade ao SLAC porque, por ter sido o primeiro trabalho essencialmente nessa área naquele Laboratório, gerou interesse de seus membros no assunto. Além disso, ainda produziu diversas aplicações aos dados do PingER, algumas delas nunca antes imaginadas e, principalmente, os abriu para a comunidade através de uma forma padronizada (padrões LOD) provendo fácil acesso público

e interoperabilidade. Em adição aos aspectos tecnológicos, este trabalho despertou maior interesse dos membros do PingER em monitorar ainda mais nós de rede brasileiros. Ademais, como efeito colateral, devido ao intenso consumo por esse projeto dos dados gerados pelo PingER, algumas anomalias foram detectadas e reparadas.

Finalmente, durante o desenvolvimento deste trabalho, foi necessário intenso contato com artigos científicos, o que estimulou o interesse em pesquisar aprofundadamente os detalhes por detrás de toda essa área que tem cada vez mais ganhado força. Web semântica ainda tem características de tecnologia em desenvolvimento, porém ela tem claro potencial em evoluir muito mais, se tornar popular e ir até além das proporções sonhadas por Berners-Lee (BERNERS-LEE; SHADBOLT; HALL, 2006). Para isso, é necessário desenvolver aplicações que consumam as tecnologias e conceitos da Web Semântica e produzir significativamente mais conhecimento científico e avanços tecnológicos na área, de modo que se entenda ainda mais todos os conceitos envolvidos, se identifique as demandas específicas da sociedade e estimule a população a utilizar as tecnologias interessantes e poderosas.

5.2 Trabalhos futuros

Os trabalhos futuros podem ser divididos em trabalhos específicos para o PingER LOD e trabalhos relacionados à Web Semântica em geral.

a) PingER LOD

- Muitas universidades monitoradas pelo PingER não estão sendo capturadas utilizando a abordagem mencionada na seção 4.3.2. Poderia investigar como aumentar a quantidade de universidades de modo a mostrar um gráfico mais rico (4.5.2). Talvez utilizar ferramentas especializadas em identificar *links* com a DBPedia (exemplo, DBPedia Spotlight[75]) ou ainda utilizar diferentes base de dados RDF sobre universidades.
- O sistema de atualização de dados da base está ineficiente. Se, por exemplo, um nó de rede mudar de nome, o sistema armazenará 2 nomes: o antigo e o novo, sem distinção de semântica. É preciso consertar isso para deixar os dados mais consistentes.
- A base de dados RDF gerada no projeto ainda não está no DataHub, nem no LOD Cloud e SPARQLES (seções 2.5.2 e 4.4). É preciso adicionar para aumentar a

[75] https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki

possibilidade dos dados do PingER LOD serem descobertos e utilizados pela comunidade. Publicar o *Dump RDF* (2.5.2) dos dados da base e adicionar descritores mais informativos do BD RDF, utilizando vocabulário VOID (2.4.3.1) ajudará nisso.

b) Web Semântica

Apesar de já existirem pesquisas nestas áreas, os seguintes tópicos poderiam ser mais estudados:

- Otimizações de consultas SPARQL – Como os processadores de consulta SPARQL funcionam? Durante a execução de algumas delas, foi observado que uma leve alteração na consulta (sendo seu resultado o mesmo) fazia o tempo de execução aumentar significativamente. Provavelmente isso depende do otimizador de consulta do SGBD RDF na qual a consulta foi executada. De qualquer maneira, é preciso investigar como as consultas poderiam ficar mais rápidas.
- Otimizações de consultas federadas – Como vimos (2.4.4), as consultas SPARQL Federadas são promissoras e parecem ser muito importantes para o conceito de LOD. Entretanto, ainda é preciso muita pesquisa e desenvolvimento tecnológico para deixá-las realmente eficientes.
- Indexação das triplas na camada física – Existe muita diferença entre os SGBDs RDF em relação a isso. Qual será a melhor abordagem? Qual será o melhor SGBD RDF para um determinado projeto? É preciso pesquisar para ter resultados mais claros.
- Ontologia interferindo no desempenho – Foi verificado que modificações sutis na ontologia do domínio alteram significativamente o tempo de execução das consultas. Mais investigações poderiam ser feitas nessa área.
- LOD e processamento de Big Data – Otimizações para *datasets* realmente gigantes se faz cada vez mais necessário, visto que banco de dados RDF com no mínimo 1 milhão de triplas é consideravelmente comum. Sabe-se que já existem SGBDs RDF que processam até 1 trilhão de triplas (reveja seção 3.3).
- Ferramentas web de visualização e modelagem de ontologias – Uma das maneiras de popularizar a Web Semântica será construindo cada vez mais tecnologias que utilizem e facilitem seus conceitos. Uma ferramenta web de visualização interativa da ontologia seria extremamente útil já que é a ontologia a base para entender o que os dados significam e como eles estão ligados.

REFERÊNCIAS

3KBO. **Strategies for Building Semantic Web Applications:** Geonames. 2013. Disponível em http://notes.3kbo.com/node/30. Acessado em nov. 2013.

ALPERT, Jesse.; HAJAJ, Nissan. **We knew the web was big**. Google Official Blog. 2008. Disponível em: http://googleblog.blogspot.com.br/2008/07/we-knew-web-was-big.html. Acessado em: nov. 2013.

ARAÚJO, Gomes Sabino de. **Consultas SPARQL federadas**. 2012. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Departamento de Ciência da Computação.

AUER, Sören *et al*. Managing the Life-Cycle of Linked Data with the LOD2 Stack. *In:* CUDRÉ-MAUROUX, E. *et al*. **The Semantic Web – ISWC 2012**: Lecture Notes in Computer Science. Alemanha, Springer Berlin Heidelberg, 2012a. p. 1-16.

AUER, Sören; RICHERT, Thomas; TRAMP, Sebastian; MARTIN, Michael; FRISCHMUTH, Philipp. **OntoWiki – An Authoring, Publication and Visualization Interface for the Data Web**. Alemanha, 2012b.

BERLIN SPARQL BENCHMARK. **Berlin SPARQL Benchmark V3.1 Results**. 2013. Centrum Wiskunde & Informatica, Berlin, Alemanha. Disponível em: http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/results/V7/. Acessado em: nov. 2013.

BERNERS-LEE, Tim. **An attempt to give a high-level plan of the architecture of the Semantic WWW**. 1998. Disponível em: http://www.w3.org/DesignIssues/Semantic.html. Acessado em: nov. 2013.

____. **Linked Data**. 2009. Disponível em: http://www.w3.org/DesignIssues/LinkedData.html. Acessado em: nov. 2013.

BERNERS-LEE, Tim.; CAILLIAU, Robert. **WorldWideWeb:** Proposal for a HyperText Project. 1990. Disponível em: http://www.w3.org/Proposal.html. Acessado em: nov. 2013.

BERNERS-LEE, Tim.; SHADBOLT, Nigel.; HALL, Wendy. **The Semantic Web revisited.** IEEE, 2006. Disponível em: http://eprints.soton.ac.uk/262614/1/Semantic_Web_Revisted.pdf. Acessado em: nov. 2013.

BIO ONTOLOGY. **Comparison of triple stores**. 2010. Disponível em: http://www.bioontology.org/wiki/images/6/6a/Triple_Stores.pdf. Acessado em: nov. 2013.

BIZER, Christian; SCHULTZ, Andreas. **The Berlin SPARQL Benchmark**. Web-based Systems Group, Freie Universität Berlin, 2010. Disponível em: http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/Bizer-Schultz-Berlin-SPARQL-Benchmark-IJSWIS.pdf. Acessado em: nov. 2013.

BLOOMBERG BUSINESSWEEK. **Q&A with Tim Berners-Lee. Bloomberg Technology**, 2007. Disponível em: http://www.businessweek.com/stories/2007-04-09/q-and-a-with-tim-berners-leebusinessweek-business-news-stock-market-and-financial-advice. Acessado em: nov. 2013.

BUIL-ARANDA, Carlos; HOGAN, Aidan; UMBRICH, Jürgen; VANDENBUSSCHE, Pierre-Yves. **SPARQL web-querying infrastructure: ready for action?** PUC-Chile, National University of Ireland, Fujitsu Limited, 2013. Disponível em: http://vmwebsrv01.deri.ie/sites/default/files/publications/paperiswc.pdf. Acessado em: nov. 2013.

CAMPOS, Maria Luiza Machado. **Notas e apresentações de aula.** 2013. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Departamento de Ciência da Computação.

COTTRELL, Les. **Internet End-to-end Performance Monitoring**. 2001. Disponível em: http://www-iepm.slac.stanford.edu/. Acessado em: nov. 2013.

____. **PingER and the digital divide**. 2011. Disponível em: https://confluence.slac.stanford.edu/download/attachments/123309267/brochure.docx. Acessado em: nov. 2013.

____. **PingER Case Studies**. 2013. Disponível em: https://confluence.slac.stanford.edu/display/IEPM/PingER+Case+Studies. Acessado em: nov. 2013.

COTTRELL, Les; MATTHEWS, Warren. **The PingER Project: Active Internet Performance Monitoring for the HENP Community**. v. 38, p. 130-136. 2000. Disponível em: http://www.ece.ucdavis.edu/~chuah/classes/eec274/eec274-s06/refs/00PingER.pdf. Acessado em: nov. 2013.

CYGANIAK, Richard; HAUSENBLAS, Michael; ALEXANDER, Keith; ZHAO, Jun. **Describing Linked Datasets**: on the Design and Usage of voiD, the "Vocabulary of Interlinked Datasets". 2009. Disponível em: http://richard.cyganiak.de/2008/papers/void-ldow2009.pdf. Acessado em: nov. 2013

DJURIC, Dragan; GAŠEVIC, Dragan; DEVEDŽIC, Vladan. **The Tao of Modeling Spaces**. Journal Of Object Technology: ETH Zurich, 2006. Disponível em: http://www.jot.fm/issues/issue_2006_11/article4/article4.pdf. Acessado em: nov. 2013.

EUROPEAN COMMISSION. **FP7: the future of European Union research policy**. Research & Innovation, 2012. Disponível em: http://ec.europa.eu/research/fp7/index_en.cfm. Acessado em: nov. 2013.

EUROPEAN TELECOMMUNICATIONS STANDARDS INSTITUTE. **Measurement Ontology for IP traffic (MOI); Requirements for IP traffic measurement ontologies development**. V1.1.1. ETSI Industry Specification Group (IGS). 2012. Disponível em:

http://www.etsi.org/deliver/etsi_gs/MOI/001_099/002/01.01.01_60/gs_moi002v010101p.pdf. Acessado em: nov. 2013.

FERMAN, Fabio. **Operadores Analíticos para Dados Estatísticos na Web de Dados**. 2013. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Departamento de Ciência da Computação.

FRANCONI, Enrico. **Description Logics: Tutorial Cource Information**. 2002. Disponível em: http://www.inf.unibz.it/~franconi/dl/course/. Acessado em: nov. 2013.

GEONAMES. **GeoNames Ontology**. 2013. Disponível em: http://www.geonames.org/ontology/documentation.html. Acessado em: nov. 2013.

GRUBER, Thomas R. **A translation approach to portable ontology specifications**. QRZOHGJH $FTXLVLWLRQ, [S.l.], v. 5, p. 199-220, 1993.

GUARINO, N. **Some ontological principles for designing upper level lexical resources**. In: INTERNATIONAL CONFERENCE ON LANGUAGE RESOURCES AND EVOLUTION, 1., 1998, Granada. 3URFHHGLQJV. 1998. Disponível em: http://www.loa-cnr.it/Papers/LREC98.pdf. Acessado em: nov. 2013.

GUARINO, N.; CARRARA, M.; GIARETTA, P. **An ontology of meta-level categories**. LADSEB-CNR Int. Rep. 6/93, Preliminary version, nov. 1993

GUINARD, Dominique; TRIFA, Vlad. **Towards the Web of Things**: Web Mahups for Embedded Devices. Institute for Pervasive Computing, Suíça, 2009.

GUIZZARDI, Giancarlo. **Uma abordagem metodológica de desenvolvimento para e com reúso, baseada em ontologias formais de domínio.** 2000. Universidade Federal do Espírito Santo. Disponível em: http://www.loa.istc.cnr.it/Guizzardi/MSc.htm. Acessado em: nov. 2013.

HARTH, Andreas; DECKER, Stefan. **Optimized index structures for querying RDF from the web**. Digital Enterprise Research Institute (DERI), National University of Galway, Ireland, Freiburg University, n.d. Disponível em: http://dip.semanticweb.org/documents/Harth-Decker-yars.pdf. Acessad o em: nov. 2013.

HIGH PERFORMANCE COMPUTING AND NETWORKING. **EU FP7 – Monitoring and measurement in the next generation technologies (MOMENT)**. 2010. Disponível em: http://www.hpcn.es/projects/monitoring-and-measurement-in-the-next-generation-technologies-moment/. Acessado em: nov. 2013.

HORRIDGE, Matthew. **A practical guide to building owl ontologies using protégé 4 and co-ode tools**. 1.3.ed. University of Manchester, 2011. Disponível em: http://130.88.198.11/tutorials/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf. Acessado em: nov. 2013.

JOHNSON, Bobbie. **Where does Wolfram Alpha get its information?** Guardian, 2009. Disponível em: http://www.theguardian.com/technology/2009/may/21/1. Acessado em: nov. 2013.

KIMBALL, Ralph; CASERTA, Joe. **The Data Warehouse ETL Toolkit**. Wiley India Pvt. Limited, 2004.

KOBIE, Nicole. **Q&A: Conrad Wolfram on communicating with apps in Web 3.0**. IT PRO, 2010. Disponível em: http://www.tweakandtrick.com/2012/05/web-30.html. Acessado em: nov. 2013.

LEARN DATA MODELING. **Star Schema: General information**. 2012. Disponível em: http://www.learndatamodeling.com/star.php#.UogBUsTUDIF. Acessado em: nov. 2013.

NAVATHE, Shamkant; ELMASRI, Ramez. **Fundamentals of Database Systems**. 6.ed. Addison-Wesley, 2010.

KIMBALL, Ralph; CASERTA, Joe. **The Data Warehouse ETL Toolkit**. Wiley India Pvt. Limited, 2004.

NOBEL PRIZE. **The Nobel Prize in Physics 2013**. Nobel Media, 2013. Disponível em: http://www.nobelprize.org/nobel_prizes/physics/laureates/2013/. Acessado em: nov. 2013.

ONTOTEXT. **OWLIM**. 2013. Disponível em: http://www.ontotext.com/owlim. Acessado em: nov. 2013.

____. **Semantic Repository**. 2013. Disponível em: http://www.ontotext.com/semantic-repository. Acessado em: nov. 2013.

OPEN RDF. **Sesame 2.7 Developer documentation**. 2013. Disponível em: http://www.openrdf.org/documentation.jsp. Acessado em: nov. 2013.

PIDD, M. **Tools for Thinking - Modeling in Management Science**. Wiley, New York, 2000.

RAO, SATHYA. **Monitoring and measurement in the next generation technologies**. 2010. Disponível em: ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/future-networks/projects-moment-factsheet_en.pdf. Acessado em: nov. 2013.

ROBERTS, Ric. **Understanding RDF serialisation formats**. Swirrl, 2012. Disponível em: http://blog.swirrl.com/articles/rdf-serialisation-formats/. Acessado em: nov. 2013.

SALVADOR, Alfredo; VERGARA, Jorge E. López de; TROPEA, Giuseppe; FERREIRO, Nicola Blefari-Melazzi Ángel; KATSU, Álvaro. **A semantically distributed approach to map ip traffic measurements to a standardized ontology**. International Journal Of Computer Networks & Communications, v. 2, no.1, 2010. Disponível em: http://airccse.org/journal/cnc/0110s02.pdf. Acessado em: nov. 2013.

SCHMIDT, Michael. **Foundations of SPARQL query optimization**. Freiburg University, 2010. Disponível em: http://www.informatik.uni-freiburg.de/~mschmidt/docs/diss_final01122010.pdf. Acessado em: nov. 2013.

SEMANTIC WEB. **Semantic Web Wiki**. 2013. Disponível em: http://semanticweb.org/wiki/Main_Page. Acessado em: nov. 2013.

SHNEIDERMAN, Ben. **Web science: a provocative invitation to computer science**. ACM, 2007. Disponível em: http://doi.acm.org/10.1145/1247001.1247022. Acessado em: nov. 2013.

SIEGEL, David. **Pull: O futuro da Internet e o impacto da Web Semântica em seus negócios**. 1.ed. São Paulo: Campus, 2010. p. 137-141.

SPARQLES. **Discoverability**. 2013. Disponível em: https://github.com/pyvandenbussche/sparqles/wiki/Discoverability. Acessado em: nov. 2013.

TRAMP, Sebastian; FRISCHMUTH, Philipp; HEINO, Norman. **OntoWiki**, **a Semantic Data Wiki Enabling the Collaborative Creation and (Linked Data) Publication of RDF Knowledge Bases**. 2010.

ULLMAN, Jeffrey. **First Course in database systems**. 1997. Prentice-Hall Inc., Simon & Schuster.

WEB NEXT. **GGG, WWW, 123**. 2007. Disponível em: http://web3next.blogspot.com.br/2007/11/ggg-www-123.html. Acessado em: nov. 2013.

WHEELER, A. **Nobel Prize in Physics 1990.** SLAC Library, 2006. Disponível em: http://www.slac.stanford.edu/library/nobel/nobel1990.html. Acessado em: nov. 2013.

WILLIAMS, James. **Introducing the concept of Web 3.0**. Tweak and Trick, 2011. Disponível em: http://www.tweakandtrick.com/2012/05/web-30.html. Acessado em: nov. 2013.

WORLD WIDE WEB CONSORTIUM. **Internet Media Type registration, consistency of use**. 2002. Disponível em: http://www.w3.org/2001/tag/2002/0129-mime. Acessado em: nov. 2013.

____. **Media types issues for text RDF formats**. 2008a. Disponível em: http://www.w3.org/2008/01/rdf-media-types. Acessado em: nov. 2013.

____. **OWL Web Ontology Language guide**. 2004a. Disponível em: http://www.w3.org/TR/owl-guide/. Acessado em: nov. 2013.

____. **RDF/XML syntax specification**. 2004b. Disponível em: http://www.w3.org/TR/REC-rdf-syntax/. Acessado em: nov. 2013.

____. **Semantic Web**. 2013a. Disponível em: http://www.w3.org/standards/semanticweb/. Acessado em: nov. 2013.

____. **SPARQL 1.1 Federated Query**. 2013b. Disponível em: http://www.w3.org/TR/sparql11-federated-query/. Acessado em: nov. 2013.

____. **SPARQL 1.1 Overview**. 2013c. Disponível em: http://www.w3.org/TR/sparql11-overview/. Acessado em: nov. 2013.

____. **SPARQL Implementations**. 2013d. Disponível em: http://www.w3.org/wiki/SparqlImplementations. Acessado em: nov. 2013.

____. **SPARQL protocol for RDF**. 2008b. Disponível em: http://www.w3.org/TR/rdf-sparql-protocol/. Acessado em: nov. 2013.

____. **SPARQL query language for RDF**. 2008c. Disponível em: http://www.w3.org/TR/rdf-sparql-query/. Acessado em: nov. 2013.

____. **Technical report development process: Maturity level for work in progress**. 2005. Disponível em: http://www.w3.org/2005/10/Process-20051014/tr.html#q73. Acessado em: nov. 2013.

____. **Time ontology in OWL**. 2006. Disponível em: http://www.w3.org/TR/owl-time/. Acessado em: nov. 2013.

APÊNDICES

APÊNDICE A – GLOSSÁRIO DOS DADOS DO PROJETO PINGER LINKED OPEN DATA

App User – It is the node information used by PingER that refers to the Windows user name of the last user to edit the node's record through the User Interface. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Average Round Trip Delay – It is the arithmetic mean of a sample of RTT values measured in a given time period, say 1 hour.

Beacon Node – A node that is meant to be monitored by all PingER monitoring hosts.

Conditional Loss Probability (CLP) – It is the probability that if one packet is lost, the following packet is also lost. More formally, *Conditional_loss_probability = Probability(loss(packet n+1)=true | loss(packet n) = true) (*http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#loss).

Contact Information – It is the node information used by PingER that keeps the name and email address(es) of the node's maintainer(s). https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Continent – The continent where the node is located. Europe, Africa, Asia, North America, South America, Oceania, Antarctica. It can be also a subclass of the entity L.CONT defined by Geonames http://www.geonames.org/export/codes.html.

Country – The country where the node is located.

County – A second-order administrative division. A subdivision of a first-order administrative division. It is commonly applicable for nodes in the United States.

Data Server – It is the node information used by PingER that refers to the URL for retrieving PingER data from the described node. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Directivity - This is a metric to identify the directness of the connection between 2 nodes at known locations. Directness is represented by Alpha. Alpha close to 1 means that the path between the hosts follows a roughly great circle route. Values much smaller than 1 mean that the path is very indirect. The derivation of directness coefficient is given here. http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#directness

Duplicate Packets – Measures the amount of duplicate ping response. See [ref http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#duplicates] for details about how this event may occur.

Endowment - The total value of an institution's investments is often referred to as the institution's endowment and is typically organized as a public charity, private foundation, or trust. In PingER LOD, the values of schools' endowment are currently being retrieved from DBPedia. http://en.wikipedia.org/wiki/Financial_endowment

Faculty Size – Usually, it refers to the number of teaching staff of a school. In PingER LOD, the values of schools` faculty size are currently being retrieved from DBPedia.

IEPM-BW - It was the Internet End-to-end Performance Monitoring (IEPM) project's Bandwidth Monitoring. It was the predecessor of the perfSONAR monitoring and was terminated in 2007.

Inter Packet Delay Variation (IPDV) – Also known as short term variability or "jitter" of the response time. Jitter is a symptom that there is congestion, or not enough bandwidth to handle the traffic. The jitter specifies the length of the VoIP codec de-jitter buffer to prevent over- or under-flow. It is very important for real-time applications such as telephony. Web browsing and mail are fairly resistant to jitter, but any kind of streaming media (voice, video, music) is quite susceptible to jitter. An objective could be to specify that say 95% of packet delay variations should be within the interval [-30msec, +30msec] (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#variable).

Mean Opinion Score (MOS) – It is a voice quality metric used by the telecommunications industry. The values of the MOS are: 1= bad; 2=poor; 3=fair; 4=good; 5=excellent. See also http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#mos

Measurement – A system used to determine factors of an object such as height, length, width, etc. In PingER, the main factors measured are Round Trip Time (RTT), and loss. The system used to make the measurements is ping.

Measurement Data – Any data concerning a measurement. It can be the value of a measurement, the definition of the metric being measured, or the information about the nodes being measured.

Metric – For the network measurement, a metric is the concept that defines which measurement is being made and how it is made. TCP Throughput and Packet Loss are examples of metrics

Maximum Round Trip Delay – It is the maximum value of a sample of RTT values measured in a given time period, say 1 hour.

Minimum Round Trip Delay – It is the minimum value of a sample of RTT values measured in a given time period, say 1 hour.

Monitor Node – Also known as PingER site and Monitoring Node. A PingER monitoring node is one that has installed the PingER monitoring software and is actively using PingER to monitor other nodes. In the metric (pair source and destination nodes), it is the source node.

Monitored Node – Also known as remote node. In the metric (pair source and destination nodes), it is the destination node. In other words, it is any node that is monitored by a monitor node.

Nearest City – The nearest town or city from where the node is located with population greater than 15000 habitants, according to Geonames data.

Node – A point at which network links intersect or branch a connecting point. This typically refers to a router, switch or end point. It can also be referred as "site". Note: an end-point is usually a host (i.e. a server, desktop, laptop, smartphone, notepad etc.).

Node Comments – Comments and notes on when and how the node's record was last updated. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Node GMT – PingER information about the node's time offset from GMT.

Node Information – Also known as Node Details. It is the generic term used to any information that describes a node. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Node Name – It is the generic term that refers to any type of name of a node. In addition to the DNS host name, a node can have a full name, a nick name, and a site name. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Node URL – It is the URL for the home page for the institution running the node. For the PingER Node example, "http://www.slac.stanford.edu" is the Node URL.

Out of Order Packets - For each sample of 10 packets, it looks to see if the sequence numbers of the responses are received in the same order as the requests were sent. If not, then that sample is marked as having one or more out of order responses. For a given interval (say a month) the value reported for out of order is the fraction of samples that were marked with out of order ping responses. Since the ping packets are sent at one second intervals it is expected that the fraction of out of order samples will be very small, and may be worth investigating whenever it is not.

Packet Loss - The loss is a good measure of the *quality* of the link (in terms of its packet loss rates) for many TCP based applications. Loss is typically caused by congestion which in turn causes queues (e.g. in routers) to fill and packets to be dropped. Loss may also be caused by the network delivering an imperfect copy of the packet. This is usually caused by bit errors in

the links or in network devices. (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#loss-measure*)*.

Physical Location – It contains information about where the node is located. It records location information provided by PingER. Note: It is commonly used just to keep track of PingER information about a node.

Ping – The name of a standard software utility (tool) used to test network connections. Ping tools send request messages to a target network address at periodic intervals and measure the time it takes for a response message to arrive.

Ping Server – It is the node information used by PingER that refers to the URL for requesting a ping from the described node to another. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Ping Size – It is the node information used by PingER that refers to the size of pings to be sent to the described node - only controls SLAC's PingER install. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

PingER Node Information - It is the generic term that refers to additional information that PingER needs to use about a node.

Project Type – It is the node information used by PingER that refers to flags that describe how the node is used. Flag "NOT-SET" is the default, meaning that it is a regular monitored node; Flag "B" means a beacon node; Flag "D" refers to a disabled node, i.e., no longer monitored, monitors or checked; Flag "I" means IEPM-BW; Flag "M" refers to a monitor node. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Round Trip Delay - also known as response time or Round Trip Time (RTT). The RTT is related to the distance between the sites plus the delay at each hop along the path between the sites. The distance effect can be roughly characterized by the speed of light in fiber, and is roughly given by *distance / (0.6 * c)* where *c* is the velocity of light (the ITU in document G.144, table A.1 recommends a multiplier of 0.005 ms/km, or 0.66c). Putting this together with the hop delays, the RTT *R* is roughly given by: *R = 2 * (distance / (0.6 * c) + hops * delay)* where the factor of 2 is since we are measuring the out and back times for the round-trip (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#rtt-measure)

Sample – A set of n pings.

School – Any educational establishment that is a node monitored by PingER or that is a PingER monitor e.g. Stanford University.

Simple Measurement – All measurements made by PingER that include a metric (such as Packet Loss or TCP Throughput). Information about a simple measurement typically include the name of the metric being measured, the default unit, and the default packet size.

Site Name – It is the domain name of a node. For the PingER node example, "slac.stanford.edu" is the site name. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Source Full Name – It is a human-friendly description of a node. For the PingER node example, "Stanford Linear Accelerator Center" is the full node name. See also **Source Name**. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Source IP – It is the IPv4 address of the node. **See also Source Name.**

Source Name – It is the DNS node name. It is commonly used to identify a node. For the PingER node example, "pinger.slac.stanford.edu" is the name. Note: The term "source" in a node name means that the name refers to the node being specified. In a metric (pair source and destination nodes), a source name is used for the description of the DNS of both the source node and the destination nodes. In other words, a destination node also has a source name. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Source Nick Name – It is an abstraction of the node name with the TLD first. For the PingER node example, "EDU.SLAC.STANFORD.N3" is the nick name. See also **Source Name**. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

State – The state where the node is located when the node is located in a country that can be divided into states or a first order division equivalent. It can be also a subclass of the entity A.ADM1 defined by Geonames. http://www.geonames.org/export/codes.html

Statistical Analysis – It is the definition of a specific measurement. It combines all parameters needed for a simple measurement and has the value of the measurement. It is defined by a combination of the name of the network metric (e.g. Packet Loss), the metric (pair source and destination nodes involved), and information about when the measurement was made.

TCP Throughput – Measures the transfer rate or throughput using TCP as transport protocol. The standard unit used by PingER is kbit/s. See also http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#derive and http://www.slac.stanford.edu/comp/net/wan-mon/thru-vs-loss.html for more information about how TCP Throughput is measured.

Throughput – Refers to how much data can be transferred from one location to another in a given amount of time (http://www.techterms.com/definition/throughput).

Time Stamp – It contains information about when the measurement was done. It is a subclass of Date Time Description defined by the W3C Time Ontology. See also Date Time Description in http://www.w3.org/TR/owl-time/.

Town – The town or city where the node is located. It can be also a subclass of the entity P.PPL defined by Geonames. http://www.geonames.org/export/codes.html

Trace Route – In computing, traceroute is a computer network diagnostic tool for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. http://en.wikipedia.org/wiki/Traceroute

Trace Server – It is the node information used by PingER that refers to the URL for requesting a traceroute from the described node to another. https://confluence.slac.stanford.edu/display/IEPM/PingER+NODEDETAILS.

Unit – The unit in which the metric is measured.

Unpredictability – The calculation of the distance of each predictability point from the coordinate (1,1). We normalize to a maximum value of 1 by dividing the distance by sqrt(2). It gives a percentage indicator of the unpredictability of the ping performance (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#unpredict). See http://www.slac.stanford.edu/comp/net/wan-mon/unpredict.html for more details.

Unreachability - By looking at the ping data to identify 30 minute periods when no ping responses were received from a given host, one can identify when the host was down. Using this information one can calculate *ping unreachability= (# periods with Node down / total number of periods)*, # Down periods, Mean Time Between Failure (MTBF or Mean Time To Failue MTTF)) and Mean Time To Repair (MTTR). Note that *MTBF = sample_time/ping_unreachability* where for PingER sample time is 30 minutes. The reachability is very dependent on the remote host, for example if the remote host is renamed or removed, the host will appear unreachable yet there may be nothing wrong with the network. Thus before using this data to provide long term network trends the data should be carefully scrubbed for non-network effects (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#availability).

Zero Packet Loss Frequency – also known as Quiescent Network Frequency. When we get a zero packet loss sample, we refer to the network as being quiescent (or non-busy). We can then measure the percentage frequency of how often the network was found to be quiescent. A high percentage is an indication of a good (quiescent or non-heavily loaded) network (http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#quiescent).

APÊNDICE B – MAPEAMENTO DOS PREFIXOS UTILIZADOS NO PROJETO PINGER LINKED OPEN DATA

```
PREFIX : <http://www-iepm.slac.stanford.edu/pinger/lod/resource#>
PREFIX PingER-ont: <http://www-
iepm.slac.stanford.edu/pinger/lod/ontology/PingEROntology.owl#>
PREFIX dbp-owl: <http://dbpedia.org/ontology/>
PREFIX dbp-prop: <http://dbpedia.org/property/>
PREFIX dbp-rsrc: <http://dbpedia.org/resource/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX fb: <http://rdf.freebase.com/ns/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX gn: <http://sws.geonames.org/>
PREFIX gn-ont: <http://www.geonames.org/ontology#>
PREFIX MD: < http://www.fp7-moment.eu/MomentDataV2.owl#>
PREFIX MGC: < http://www.fp7-moment.eu/MomentGeneralConcepts.owl#>
PREFIX MU: < http://www.fp7-moment.eu/MomentUnits.owl#>
PREFIX pos:<http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX Units: < http://www.fp7-moment.eu/Units.owl#>
PREFIX void: <http://rdfs.org/ns/void#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
```

APÊNDICE C – CONSULTA SPARQL PARA GERAR O GRÁFICO DE MÚLTIPLAS MÉTRICAS DE REDE

```
SELECT DISTINCT ?MetricType ?TimeValue (AVG(?value) as ?Average) WHERE {
      ?metric MD:hasSourceNodeInformation ?source .
      ?metric MD:hasDestinationNodeInformation ?destination .
      ?source MD:hasNodeInformation ?SourceName .
      ?SourceName MD:SourceNameValue
'pinger.slac.stanford.edu'^^xsd:string .

      ?destination MGC:isInTown ?DestTown .
      ?DestTown MGC:GeoCountry 'Pakistan'^^xsd:string .
      {
            ?StatisticalAnalysis
MD:measurementsAnalyzed :SimpleMeasurement-
AverageRoundTripDelayMeasurement .
      }
      UNION
      {
            ?StatisticalAnalysis
MD:measurementsAnalyzed :SimpleMeasurement-PacketLossMeasurement .
      }
      UNION
      {
            ?StatisticalAnalysis
MD:measurementsAnalyzed :SimpleMeasurement-TCPThroughputMeasurement .
      }
      ?StatisticalAnalysis MD:measurementsAnalyzed ?MetricType .
      ?StatisticalAnalysis MD:timestamp ?time .
      ?StatisticalAnalysis MD:measuresMetric ?metric .
      ?StatisticalAnalysis MD:StatisticalAnalysisValue ?value .

      ?time time:unitType time:unitMonth .
      ?time MGC:displayValue ?TimeValue .
      ?time MGC:startDate ?start .
      ?time MGC:endDate ?end .
      filter (
            xsd:dateTime(?start) >= '2012-08-01T12:00:00'^^xsd:dateTime &&
            xsd:dateTime(?end) <= '2013-08-31T12:00:00'^^xsd:dateTime
      ) .
}
GROUP BY ?TimeValue ?start ?MetricType
ORDER BY ?start
```

APÊNDICE D – CONSULTA SPARQL PARA GERAR O MAPA DE MÉTRICA DE REDE X MÉTRICA DE UNIVERSIDADE

```
SELECT DISTINCT ?SchoolURI
      ?NodeName ?NodeNickName ?NodeType ?NodeURL ?SchoolURI ?SchoolName
      ?SchoolType ?SchoolMetricValue ?Lat ?Long ?TownName ?CountryName
      ?DBPediaLink
      ?SchoolFacultySize ?SchoolNumberOfGradStudents
      ?SchoolNumberOfUgradStudents ?SchoolEndowment
      (AVG(?value) AS ?Average)
      WHERE {
            ?metric MD:hasSourceNodeInformation ?source .
            ?metric MD:hasDestinationNodeInformation ?destination .

            ?source MD:hasNodeInformation ?SourceName .
            ?SourceName MD:SourceNameValue
'pinger.slac.stanford.edu'^^xsd:string .

            ?destination MD:hasNodeInformation ?dstname .
            ?dstname rdf:type MD:SourceName .
            ?dstname MD:SourceNameValue ?NodeName .

            ?destination MD:hasNodeInformation ?dstnickname .
            ?dstnickname rdf:type MD:SourceNickName .
            ?dstnickname MD:SourceNickNameValue ?NodeNickName .

            ?destination MD:hasNodeInformation ?ntype .
            ?ntype rdf:type MD:ProjectType .
            ?ntype MD:ProjectTypeValue ?NodeType .

            OPTIONAL {
                  ?destination MD:hasNodeInformation ?nUrl .
                  ?nUrl rdf:type MD:NodeURL .
                  ?nUrl MD:NodeURLValue ?NodeURL .
            }

            ?destination MGC:isInSchool ?SchoolURI .
            ?SchoolURI rdf:type MGC:School .
            ?SchoolURI MGC:SchoolName ?SchoolName .
            ?SchoolURI MGC:SchoolNumberOfStudents ?SchoolMetricValue .
            ?SchoolURI pos:lat ?Lat .
            ?SchoolURI pos:long ?Long .
            ?SchoolURI MGC:DBPediaLink ?DBPediaLink .
            OPTIONAL {
                  ?SchoolURI MGC:SchoolType ?SchoolType .
                  ?SchoolURI MGC:isInTown ?schooltown .
                  ?schooltown MGC:GeoCountry ?CountryName .
                  ?schooltown gn-ont:name ?TownName .
            }
            OPTIONAL {
                  ?SchoolURI MGC:SchoolFacultySize ?SchoolFacultySize .
            }
            OPTIONAL {
                  ?SchoolURI
MGC:SchoolNumberOfGradStudents ?SchoolNumberOfGradStudents .
            }
            OPTIONAL {
                  ?SchoolURI
MGC:SchoolNumberOfUgradStudents ?SchoolNumberOfUgradStudents .
            }
            OPTIONAL {
                  ?SchoolURI MGC:SchoolEndowment ?SchoolEndowment .
```

```
            }

            ?StatisticalAnalysis
MD:measurementsAnalyzed :SimpleMeasurement-PacketLossMeasurement .
            ?StatisticalAnalysis MD:timestamp ?time .
            ?StatisticalAnalysis MD:measuresMetric ?metric .
            ?StatisticalAnalysis MD:StatisticalAnalysisValue ?value .

            ?time rdf:type MGC:TimeStamp .
            ?time MGC:startDate ?start .
            ?time MGC:endDate ?end .
            ?time time:unitType time:unitYear .
            filter (
                  xsd:dateTime(?start) >= '1998-01-
01T00:00:00'^^xsd:dateTime &&
                  xsd:dateTime(?end) <= '2013-12-31T23:59:59'^^xsd:dateTime
            ) .
      }
      GROUP BY ?NodeName ?NodeNickName ?NodeType
            ?NodeURL ?SchoolURI ?SchoolName ?SchoolType ?SchoolMetricValue
            ?Lat ?Long ?TownName ?CountryName ?DBPediaLink
            ?SchoolFacultySize ?SchoolNumberOfGradStudents
            ?SchoolNumberOfUgradStudents ?SchoolEndowment
      ORDER BY ?SchoolURI ?SchoolName
```

APÊNDICE E – CONSULTA SPARQL FEDERADA COM MASHUP ENTRE PINGER LOD E WORLD BANK

```
PREFIX property: <http://worldbank.270a.info/property/>
PREFIX indicator: <http://worldbank.270a.info/classification/indicator/>
PREFIX sdmx-dimension: <http://purl.org/linked-data/sdmx/2009/dimension#>
PREFIX sdmx-measure: <http://purl.org/linked-data/sdmx/2009/measure#>

SELECT
?CountryName ?PingERYear
?ResearchDevelopmentExpenditure (AVG(?MeasurementValue) as ?Throughput)
WHERE
{
   SERVICE <http://worldbank.270a.info/sparql> {   #Endpoint Externo
            ?WBAnalysis property:indicator indicator:GB.XPD.RSDV.GD.ZS  .
            ?WBCountry a dbp-owl:Country .
            ?WBAnalysis sdmx-dimension:refArea ?WBCountry .
            FILTER ( REGEX( str(?WBCountryGeonames),
"^http://sws.geonames.org/", "i") ) .
            ?WBAnalysis sdmx-measure:obsValue ?WBValue .
            ?WBAnalysis sdmx-dimension:refPeriod ?WBTime .

            ?WBCountry owl:sameAs ?WBCountryGeonames .
            ?WBAnalysis sdmx-
measure:obsValue ?ResearchDevelopmentExpenditure .
            ?WBAnalysis sdmx-dimension:refPeriod ?WBTime .
    } .
    BIND ( SUBSTR( STR(?WBTime), 38, 4 ) AS ?WBYear ) .

      ?StatisticalAnalysis MD:measurementsAnalyzed :SimpleMeasurement-
TCPThroughputMeasurement .
      ?StatisticalAnalysis MD:timestamp ?time .
      ?StatisticalAnalysis MD:measuresMetric ?SourceDestinationNodes .

      ?SourceDestinationNodes MD:hasSourceNodeInformation :Node-
pinger.slac.stanford.edu .
      ?SourceDestinationNodes MD:hasDestinationNodeInformation ?DestNode .

      ?DestNode MGC:isInTown ?DestTown .
      ?DestTown MGC:isInCountry ?CountryURI .
      ?CountryURI gn-ont:name ?CountryName .
      ?CountryURI MGC:GeonamesLink ?PingERCountryGeonames.

      ?time time:unitType time:unitYear .
      ?time MGC:displayValue ?PingERYear  .

      ?StatisticalAnalysis MD:StatisticalAnalysisValue ?MeasurementValue .

      #JOIN
      FILTER ( xsd:string(?WBYear) = xsd:string(?PingERYear ) &&
             xsd:string(?WBCountryGeonames) =
xsd:string(?PingERCountryGeonames)
    ) .
}
GROUP
BY ?CountryName ?PingERCountryGeonames ?PingERYear ?ResearchDevelopmentExpe
nditure
ORDER BY ?CountryName ?PingERYear DESC(?ResearchDevelopmentExpenditure)
```

APÊNDICE F – CONSULTA SPARQL SOBRE PINGER LOD PARA RECUPERAR MEDIDAS DE REDE AO LONGO DOS ANOS

```
SELECT DISTINCT
?PingERCountryGeonames ?PingERYear (AVG(?MeasurementValue)
as ?NetworkMetricValue) ?CountryName  WHERE {

      ?StatisticalAnalysis MD:measurementsAnalyzed :SimpleMeasurement-
TCPThroughputMeasurement .
      ?StatisticalAnalysis MD:timestamp ?time .
      ?StatisticalAnalysis MD:measuresMetric ?SourceDestinationNodes .

      ?SourceDestinationNodes MD:hasSourceNodeInformation :Node-
pinger.slac.stanford.edu .
      ?SourceDestinationNodes MD:hasDestinationNodeInformation ?DestNode .

      ?DestNode MGC:isInTown ?DestTown .
      ?DestTown MGC:isInCountry ?CountryURI .
      ?CountryURI MGC:GeonamesLink ?PingERCountryGeonames .
      ?CountryURI gn-ont:name ?CountryName .

      ?time time:unitType time:unitYear .
      ?time MGC:displayValue ?PingERYear .

      ?StatisticalAnalysis MD:StatisticalAnalysisValue ?MeasurementValue .


}
GROUP BY ?PingERCountryGeonames ?PingERYear ?CountryName
ORDER BY ?PingERCountryGeonames ?PingERYear
```

APÊNDICE G – CONSULTA SPARQL SOBRE WORLD BANK PARA RECUPERAR % PIB DOS PAÍSES INVESTIDO EM PESQUISA E DESENVOLVIMENTO TECNOLÓGICO

```
PREFIX property: <http://worldbank.270a.info/property/>
PREFIX indicator: <http://worldbank.270a.info/classification/indicator/>
PREFIX sdmx-dimension: <http://purl.org/linked-data/sdmx/2009/dimension#>
PREFIX sdmx-measure: <http://purl.org/linked-data/sdmx/2009/measure#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
select ?WBCountryGeonames ?WBYear ?WBValue where {
      ?WBAnalysis property:indicator indicator:GB.XPD.RSDV.GD.ZS  .
      ?WBAnalysis sdmx-dimension:refArea ?WBCountry .
      ?WBCountry a dbp-owl:Country .
      ?WBCountry skos:prefLabel ?label .
   ?WBCountry owl:sameAs ?WBCountryGeonames .
   FILTER ( REGEX( str(?WBCountryGeonames), '^http://sws.geonames.org/',
'i') ) .
      ?WBAnalysis sdmx-measure:obsValue ?WBValue .
      ?WBAnalysis sdmx-dimension:refPeriod ?WBTime .
      BIND ( SUBSTR( STR(?WBTime), 38, 4 ) AS ?WBYear ) .
}
order by ?WBCountryGeonames ?WBYear
```

ANEXOS

ANEXO 1 – throughput-100-by-site-allyears.txt – Exemplo de arquivo com dados crus para o PingER LOD

```
1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012
2013
202.83.160.42 monalisa.niit.edu.pk . . . . . .
202.83.160.42 monalisa.niit.edu.pk
dns1.arm.gov archive.arm.gov . . . . . dns1.arm.gov
archive.arm.gov
dns1.arm.gov nsa.arm.gov . . . . . dns1.arm.gov
nsa.arm.gov
dns1.arm.gov sgp.arm.gov . . . . . dns1.arm.gov
sgp.arm.gov
dns1.arm.gov xdc.arm.gov . . . . . dns1.arm.gov
xdc.arm.gov
helsinki.cc.gatech.edu 194.67.220.105 . . . . . 207.8967 . . . . . .
helsinki.cc.gatech.edu 194.67.220.105
helsinki.cc.gatech.edu 194.67.220.129 . . . . . 177.5653 . . . . . .
helsinki.cc.gatech.edu 194.67.220.129
helsinki.cc.gatech.edu 194.67.220.161 . . . . . 263.4333 . . . . . .
helsinki.cc.gatech.edu 194.67.220.161
helsinki.cc.gatech.edu 194.67.220.193 . . . . . 430.8690 . . . . . .
helsinki.cc.gatech.edu 194.67.220.193
helsinki.cc.gatech.edu 194.67.220.225 . . . . . 330.1208 . . . . . .
helsinki.cc.gatech.edu 194.67.220.225
helsinki.cc.gatech.edu 194.67.220.73 . . . . . 336.2201 . . . . . .
helsinki.cc.gatech.edu 194.67.220.73
helsinki.cc.gatech.edu 194.67.220.97 . . . . . 330.9624 . . . . . .
helsinki.cc.gatech.edu 194.67.220.97
helsinki.cc.gatech.edu 202.83.160.42 . . . . . 1327.6709 . . . . . .
helsinki.cc.gatech.edu 202.83.160.42
helsinki.cc.gatech.edu ihep.ac.cn . . . . . 723.3356 . . . . . .
helsinki.cc.gatech.edu ihep.ac.cn
helsinki.cc.gatech.edu indix.ncst.ernet.in . . . . . .
204.3099 . . . . . helsinki.cc.gatech.edu indix.ncst.ernet.in
helsinki.cc.gatech.edu pinger.ntc.net.pk . . . . . .
258.8878 . . . . . helsinki.cc.gatech.edu pinger.ntc.net.pk
helsinki.cc.gatech.edu sanchar.ncb.ernet.in . . . . . .
303.4334 . . . . . helsinki.cc.gatech.edu sanchar.ncb.ernet.in
helsinki.cc.gatech.edu uclv12.uclv.edu.cu . . . . . .
helsinki.cc.gatech.edu uclv12.uclv.edu.cu
indix.ncst.ernet.in 194.67.220.105 . . . . . 105.6640 . . . . . .
indix.ncst.ernet.in 194.67.220.105
indix.ncst.ernet.in 194.67.220.129 . . . . . 96.8931 . . . . . .
indix.ncst.ernet.in 194.67.220.129
indix.ncst.ernet.in 194.67.220.161 . . . . . 104.8300 . . . . . .
indix.ncst.ernet.in 194.67.220.161
indix.ncst.ernet.in 194.67.220.193 . . . . . 103.3136 . . . . . .
indix.ncst.ernet.in 194.67.220.193
indix.ncst.ernet.in 194.67.220.225 . . . . . 104.2008 . . . . . .
indix.ncst.ernet.in 194.67.220.225
indix.ncst.ernet.in 194.67.220.73 . . . . . 113.8784 . . . . . .
indix.ncst.ernet.in 194.67.220.73
indix.ncst.ernet.in 194.67.220.97 . . . . . 114.0955 . . . . . .
indix.ncst.ernet.in 194.67.220.97
indix.ncst.ernet.in 202.83.160.42 . . . . . 127.7959 . . . . . .
indix.ncst.ernet.in 202.83.160.42
indix.ncst.ernet.in sanchar.ncb.ernet.in . . . . . .
2112.4778 . . . . . indix.ncst.ernet.in sanchar.ncb.ernet.in
indix.ncst.ernet.in trinetra.ncb.ernet.in . . . . . .
indix.ncst.ernet.in trinetra.ncb.ernet.in
```

```
kinnaird.seecs.edu.pk 194.67.220.129 . . . . . .
kinnaird.seecs.edu.pk 194.67.220.129
kinnaird.seecs.edu.pk 194.67.220.225 . . . . . .
kinnaird.seecs.edu.pk 194.67.220.225
kinnaird.seecs.edu.pk ihep.ac.cn . . . . . 380.1040
299.1848 227.5950 kinnaird.seecs.edu.pk ihep.ac.cn
kinnaird.seecs.edu.pk magic.mn . . . . . 209.1710 274.7011
370.5080 kinnaird.seecs.edu.pk magic.mn
kinnaird.seecs.edu.pk monitor.niit.edu.pk . . . . . .
2100.6740 . . kinnaird.seecs.edu.pk monitor.niit.edu.pk
kinnaird.seecs.edu.pk sanchar.ncb.ernet.in . . . . . .
286.3963 . . kinnaird.seecs.edu.pk sanchar.ncb.ernet.in
kinnaird.seecs.edu.pk web-a.lvdats.lv . . . . . 801.5276
576.6071 kinnaird.seecs.edu.pk web-a.lvdats.lv
monitor.niit.edu.pk 194.67.220.105 . . . . . 73.7641
61.8254 . . . . . monitor.niit.edu.pk 194.67.220.105
monitor.niit.edu.pk 194.67.220.129 . . . . . 65.6598 51.7406 . .
62.2073 . . . monitor.niit.edu.pk 194.67.220.129
monitor.niit.edu.pk 194.67.220.161 . . . . . 67.9539 . . . . . .
monitor.niit.edu.pk 194.67.220.161
monitor.niit.edu.pk 194.67.220.193 . . . . . 77.7956
56.3244 . . . . . monitor.niit.edu.pk 194.67.220.193
monitor.niit.edu.pk 194.67.220.225 . . . . . 64.7034 63.7658 . .
60.9618 . . . monitor.niit.edu.pk 194.67.220.225
monitor.niit.edu.pk 194.67.220.73 . . . . . 83.0839
59.1197 . . . . . monitor.niit.edu.pk 194.67.220.73
monitor.niit.edu.pk 194.67.220.97 . . . . . 68.7177
52.6269 . . . . . monitor.niit.edu.pk 194.67.220.97
monitor.niit.edu.pk 200.37.46.80 . . . . . .
monitor.niit.edu.pk 200.37.46.80
monitor.niit.edu.pk bahria.edu.pk . . . . . 164.3881 . . . . . .
monitor.niit.edu.pk bahria.edu.pk
monitor.niit.edu.pk gu.edu.pk . . . . . 171.5193 . . . . . .
monitor.niit.edu.pk gu.edu.pk
monitor.niit.edu.pk hec.gov.pk . . . . . 191.9806 . . . . . .
monitor.niit.edu.pk hec.gov.pk
monitor.niit.edu.pk ihep.ac.cn . . . . . 186.7626 136.3005 . .
114.9669 89.1363 . . monitor.niit.edu.pk ihep.ac.cn
monitor.niit.edu.pk indix.ncst.ernet.in . . . . . .
70.0635 . . . . . monitor.niit.edu.pk indix.ncst.ernet.in
monitor.niit.edu.pk lhr.comsats.net.pk . . . . . .
monitor.niit.edu.pk lhr.comsats.net.pk
monitor.niit.edu.pk magic.mn . . . . . 106.1450 89.0997 . .
monitor.niit.edu.pk magic.mn
monitor.niit.edu.pk monalisa.niit.edu.pk . . . . . 218045.1989
171345.6760 . . . . . monitor.niit.edu.pk monalisa.niit.edu.pk
monitor.niit.edu.pk monitor.niit.edu.pk . . . . . .
monitor.niit.edu.pk monitor.niit.edu.pk
monitor.niit.edu.pk pinger.ntc.net.pk . . . . . 347.6240 . . . . . .
monitor.niit.edu.pk pinger.ntc.net.pk
monitor.niit.edu.pk pinger2.niit.edu.pk . . . . . 117227.5317 120060.5280
230950.3084 126602.5040 . . . . . monitor.niit.edu.pk pinger2.niit.edu.pk
monitor.niit.edu.pk sanchar.ncb.ernet.in . . . . . 82.1559 96.6733
165.2685 . . 143.3020 131.5143 . . monitor.niit.edu.pk sanchar.ncb.ernet.in
monitor.niit.edu.pk www.lcwu.edu.pk . . . . . 195.4651
85.8212 . . . . . monitor.niit.edu.pk www.lcwu.edu.pk
pinger.ntc.net.pk 194.67.220.105 . . . . . 158.2757 . . . . . .
pinger.ntc.net.pk 194.67.220.105
pinger.ntc.net.pk 194.67.220.129 . . . . . 152.5678 . . . . . .
pinger.ntc.net.pk 194.67.220.129
```

```
pinger.ntc.net.pk 194.67.220.161 . . . . . 189.6274 . . . . . .
pinger.ntc.net.pk 194.67.220.161
pinger.ntc.net.pk 194.67.220.193 . . . . . 262.6626 . . . . . .
pinger.ntc.net.pk 194.67.220.193
pinger.ntc.net.pk 194.67.220.225 . . . . . 190.2706 . . . . . .
pinger.ntc.net.pk 194.67.220.225
pinger.ntc.net.pk 194.67.220.73 . . . . . 322.3212 . . . . . .
pinger.ntc.net.pk 194.67.220.73
pinger.ntc.net.pk 194.67.220.97 . . . . . 228.7071 . . . . . .
pinger.ntc.net.pk 194.67.220.97
pinger.ntc.net.pk 202.83.160.42 . . . . . .
pinger.ntc.net.pk 202.83.160.42
pinger.ntc.net.pk ihep.ac.cn . . . . . 522.1536 . . . . . .
pinger.ntc.net.pk ihep.ac.cn
pinger.ntc.net.pk indix.ncst.ernet.in . . . . . .
pinger.ntc.net.pk indix.ncst.ernet.in
pinger.ntc.net.pk monalisa.niit.edu.pk . . . . . .
pinger.ntc.net.pk monalisa.niit.edu.pk
pinger.ntc.net.pk sanchar.ncb.ernet.in . . . . . .
386.7992 . . . . . pinger.ntc.net.pk sanchar.ncb.ernet.in
pinger2.niit.edu.pk 194.67.220.105 . . . . . 81.3281
57.9047 . . . . . pinger2.niit.edu.pk 194.67.220.105
pinger2.niit.edu.pk 194.67.220.161 . . . . . 83.1297 . . . . . .
pinger2.niit.edu.pk 194.67.220.161
pinger2.niit.edu.pk 194.67.220.65 . . . . . .
pinger2.niit.edu.pk 194.67.220.65
pinger2.niit.edu.pk indix.ncst.ernet.in . . . . . .
626.3499 . . . . . pinger2.niit.edu.pk indix.ncst.ernet.in
pinger2.niit.edu.pk monalisa.niit.edu.pk . . . . . .
pinger2.niit.edu.pk monalisa.niit.edu.pk
pinger2.niit.edu.pk pinger.ntc.net.pk . . . . . .
1390.8517 . . . . . pinger2.niit.edu.pk pinger.ntc.net.pk
pinger2.niit.edu.pk www.lcwu.edu.pk . . . . . 355.2899 . . . . . .
pinger2.niit.edu.pk www.lcwu.edu.pk
sanchar.ncb.ernet.in amp-aarn . . . . . 395.8179 310.0496
195.4417 . . . . . sanchar.ncb.ernet.in amp-aarn
sanchar.ncb.ernet.in amp-apantyo . . . . . 694.2472 359.9392
186.1844 . . . . . sanchar.ncb.ernet.in amp-apantyo
sanchar.ncb.ernet.in amp-capetown . . . . . 251.2510 . . . . . .
sanchar.ncb.ernet.in amp-capetown
sanchar.ncb.ernet.in amp-cnic . . . . . 1797.4839 . . . . . .
sanchar.ncb.ernet.in amp-cnic
sanchar.ncb.ernet.in amp-cnic-hk . . . . . 375.3660
422.0154 . . . . . sanchar.ncb.ernet.in amp-cnic-hk
sanchar.ncb.ernet.in amp-kiwi . . . . . 474.8137 . . . . . .
sanchar.ncb.ernet.in amp-kiwi
sanchar.ncb.ernet.in amp-korea . . . . . 1150.0515 . . . . . .
sanchar.ncb.ernet.in amp-korea
sanchar.ncb.ernet.in amp-singapore . . . . . 409.5800 . . . . . .
sanchar.ncb.ernet.in amp-singapore
sanchar.ncb.ernet.in amp-taiwan . . . . . 420.8825 254.8716
163.5972 . . . . . sanchar.ncb.ernet.in amp-taiwan
sanchar.ncb.ernet.in amp-uhyderabad . . . . . .
sanchar.ncb.ernet.in amp-uhyderabad
sanchar.ncb.ernet.in amp-unin . . . . . .
sanchar.ncb.ernet.in amp-unin
sanchar.ncb.ernet.in indix.ncst.ernet.in . . . . . .
1911.6513 . . . . . sanchar.ncb.ernet.in indix.ncst.ernet.in
sanchar.ncb.ernet.in trinetra.ncb.ernet.in . . . . . 2451793.4493
39245739.3973 . . . . . sanchar.ncb.ernet.in trinetra.ncb.ernet.in
```